



Маликов Рамиль Фарукович, профессор, доктор физико-математических наук. Родился в с.Старосубхангулово Бурзянского района. Закончил Стерлитамакский государственный педагогический институт.

С 1974 г. работает Башкирском государственном педагогическом университете. В его стенах прошел путь от ассистента до профессора. В настоящее время является заведующим кафедрой информационных и полиграфических систем и технологий БашГПУ им.М.Акумлы.

Научные интересы профессора Р.Ф.Маликова сконцентрированы в области нелинейной оптики, математического моделирования реальных процессов, применению информационных и компьютерных

технологий в школьном, профессиональном обучении и в системе дистанционного обучения. Его работы по сверхфлуоресценции, индуцированному сверхизлучению, когерентному усилению импульсов хорошо известны в научном мире. Имеет более 150 научных и учебно-методических работ. Отличник образования Республики Башкортостан. Почетный работник ВПО РФ. E-mail: rfdmalikov@mail.ru



Сулейманов Ринат Рамилевич, кандидат педагогических наук, директор Центра информационных технологий Башкирского института развития образования.

Выпускник физико-математического факультета Башкирского государственного педагогического университета 1984 г.

Область научных интересов методика преподавания информатики, компьютерное решение математических задач, применение компьютерных и информационных технологий в школьном и профессиональном обучении. Имеет более 150 научных и учебно-методических работ и публикации.

Авторы написали ряд книг по информатике и применению компьютерных технологий в обучении, в том числе «Инфор-

матика занимательная и не только..», «Информатика классная и внеклассная», «Компьютерное моделирование физических явлений и объектов».

ИНФОРМАТИКА КЛАССНАЯ И ВНЕКЛАССНАЯ

Р.Ф.Маликов
Р.Р.Сулейманов



**МИНИСТЕРСТВО ОБРАЗОВАНИЯ
РЕСПУБЛИКИ БАШКОРТОСТАН**

**Р.Ф. МАЛИКОВ
Р.Р. СУЛЕЙМАНОВ**

**ИНФОРМАТИКА
КЛАССНАЯ
И ВНЕКЛАСНАЯ**

Рекомендовано Министерством образования
Республики Башкортостан
в качестве учебно-методического пособия

Издание второе, переработанное
и дополненное

Издательство «Гилем»
Уфа – 2004

УДК 518.5
ББК 74.262
М 19

Маликов Р.Ф., Сулейманов Р.Р. Информатика классная и
внеклассная: 2-е изд., перераб. и доп. Уфа: Гилем, 2004. 365 с.

ISBN 5-7501-0415-X

В книге рассмотрены классно-урочные и внеклассные формы организации занятий по информатике, предлагаются занимательные задачи по математике и физике и их решения с использованием компьютеров и новых информационных технологий. Описаны и приведены технологии работы со вспомогательными информационными системами, которые используются при обучении.

Предназначена для учителей, преподавателей информатики средних общеобразовательных школ, профессиональных учебных заведений, вузов.

Рецензенты:

д-р техн. наук, проф. (БИРО) Р.И. Сайтов

канд. пед. наук (МО РБ) М.Д. Назарова

д-р пед. наук, канд. ф.-м. наук, доцент (БГПУ) С.Г. Гильмиярова

ISBN 5-7501-0415-X

© БГПУ, 2001

© Р.Ф.Маликов, Р.Р.Сулейманов, 2004,
перераб. и доп.

© Издательство «Гилем», 2004

ОГЛАВЛЕНИЕ

Предисловие	7
ГЛАВА 1. КЛАССНО-УРОЧНАЯ ФОРМА ОРГАНИЗАЦИИ ЗАНЯТИЙ	
1.1. Задачи и особенности классно-урочных занятий по основам информатики и вычислительной техники в школе	10
1.2. Формы и методы обучения.....	11
1.3. Особенности решения задач по информатике.....	17
1.4. Особенности работы и оформление кабинета информа- тики.....	20
1.5. Урок техники безопасности.....	25
ГЛАВА 2. ПРИМЕРЫ КЛАССНО-УРОЧНОЙ ФОРМЫ ОРГАНИЗАЦИЙ ЗАНЯТИЙ	
2.1. Интегрированный урок «Приближенное вычисление ин- теграла».....	31
2.2. Семинар на тему «Уравнение».....	37
2.3. Составление и решение графических и арифметических ребусов на уроке информатики.....	42
2.4. Практическая работа по теме «Символьные величины» ..	46
2.5. Урок-анализ. Разбор решений ключевых задач по теме «Массивы».....	54
2.6. Урок-проектирование. Расчет на ЭВМ ожидаемой точ- ности изготовления детали.....	60
2.7. Уроки-лекции. Численные методы решения дифферен- циальных уравнений.....	65
ГЛАВА 3. ВНЕКЛАССНАЯ РАБОТА ПО ИНФОРМАТИКЕ В ШКОЛЕ	
3.1. Задачи, особенности и формы внеклассных занятий по информатике, программированию, информационным техно- логиям и архитектуре ЭВМ и сетей.....	82
3.2. Кружки по информатике.....	85

3.2.1. Примерный план работы кружка «Архитектура ЭВМ и сетей».....	85
3.2.2. Примерный план работы кружка «Алгоритмизация и программирование».....	86
3.2.3. Примерный план работы кружка «Информационные технологии».....	87
3.2.4. Роботландский сетевой университет.....	87
3.2.5. Клубы по интересам.....	88
3.3. Школьные олимпиады по информатике.....	88
3.4. Лекторий.....	98
3.5. Подготовка и проведение конференций.....	106
3.6. Проведение конкурсов по информатике.....	114
3.7. Проектная форма обучения	120
3.8. Школьная электронная печать.....	128

ГЛАВА 4. РЕШЕНИЕ ЗАНИМАТЕЛЬНЫХ ЗАДАЧ

4.1. Числовая загадка цифровых клавиш.....	134
4.2. Назойливая разность.....	137
4.3. Симметричная сумма.....	143
4.4. Учебно-исследовательская форма обучения.....	148
4.5. Задачи о кросс-суммах.....	154
4.6. Занимательные этюды.....	159

ГЛАВА 5. РЕШЕНИЕ МАТЕМАТИЧЕСКИХ ЗАДАЧ

5.1. Решение математических задач методом обобщения и аналогии	166
5.2. Решение задач на тему «Разложение натурального числа на натуральные слагаемые».....	170
5.3. Решение задач на тему «Простые числа».....	176
5.4. Решение задач на натуральные числа.....	184
5.5. Задачи для самостоятельного решения.....	202

ГЛАВА 6. РЕШЕНИЕ ФИЗИЧЕСКИХ ЗАДАЧ МЕТОДОМ КОМПЬЮТЕРНОГО МОДЕЛИРОВАНИЯ

6.1. Математическое (компьютерное) моделирование и этапы вычислительного эксперимента.....	208
--	-----

6.2. Моделирование движения тела, брошенного под углом к горизонту	216
6.3. Движение тела с учетом сопротивления среды.....	221
6.4. Движение шарика в вязкой среде.....	224
6.5. Задача двух тел.....	225
6.6. Одноступенчатая ракета.....	229
6.7. Движение заряженных частиц в кулоновском поле.....	232
6.8. Параметрический маятник.....	235
6.9. Выпрямление с фильтрацией.....	237

ГЛАВА 7. ПРИМЕНЕНИЕ ИНФОРМАЦИОННЫХ ТЕХНОЛОГИЙ ПРИ РЕШЕНИИ ФИЗИЧЕСКИХ И МАТЕМАТИЧЕСКИХ ЗАДАЧ

7.1. Использование электронных таблиц для графического представления числовой информации	240
7.2. Электронная таблица EXCEL как инструмент решения физических задач.....	243
7.3. Решение математических и физических задач в системе MathCAD 8.1.....	251
7.4. Разработка базы данных «Планеты солнечной системы»	259
7.5. Моделирование электрических схем из школьного курса физики в интегрированной системе Electronics Workbench ...	266
7.5.1. <i>Технология работы в системе Electronics Workbench 5.12</i>	267
7.5.2. <i>Цепи постоянного тока</i>	277
7.5.3. <i>Источники вторичного питания. Выпрямление с фильтрацией</i>	279
7.6. Изучение основ электроники и вычислительной техники в Electronics Workbench.....	282
7.6.1. <i>Основные понятия булевой алгебры</i>	282
7.6.2. <i>Примеры решения задач на тему «Логические схемы»..</i>	285
7.6.3. <i>Виртуальный логический конвертор (Logic converter)...</i>	288
7.6.4. <i>Цифровой компаратор</i>	291
7.6.5. <i>Устройство контроля четности</i>	293
7.6.6. <i>Мультиплексоры и демультиплексоры</i>	295
7.6.7. <i>Арифметические сумматоры</i>	297
7.6.8. <i>Триггеры</i>	298

7.6.9. Счетчик.....	301
7.6.10. Регистр.....	303
7.6.11. Оперативное запоминающее устройство.....	304

ГЛАВА 8. ПРИМЕНЕНИЕ ИНФОРМАЦИОННЫХ СИСТЕМ В ОБУЧЕНИИ

8.1. Обработка информации в системе MICROCAL ORIGIN 6.0	307
8.2. Система технического конструирования КОМПАС.....	309
8.3. Система автоматизированного перевода PROMT 98.....	319
8.4. Подготовка презентаций в системе POWERPOINT	329
8.5. Работа в системе MICROSOFT FRONTPAGE.....	337
8.6. Образовательные сайты в INTERNET.....	352
<i>Литература</i>	360

ПРЕДИСЛОВИЕ

Интенсивное развитие информатики и вычислительной техники требует необходимости использования компьютеров в различных областях человеческой деятельности. Стало возможным решение многих задач, требующих огромного количества арифметических или иных операций, а использование компьютеров в школе открывает перед учащимися принципиально новые возможности.

Актуальность данного пособия продиктована, с одной стороны, тем, что идет сокращение основных часов математики, физики и других предметов, с другой – идет возрастание роли информатики в обработке потока информации с помощью информационных технологий и систем и применения этих технологий для решения задач разной направленности. Один из способов решения данной проблемы, на наш взгляд, состоит в интеграции уроков информатики с уроками математики, физики и другими учебными дисциплинами. Поэтому возникает необходимость в пропедевтическом знакомстве учащихся с информационными системами и технологиями.

В настоящее время издается очень много книг по различным информационным компьютерным системам и технологиям. Для школьного учителя информатики их освоение сложно по причине недостатка времени. Данная книга вначале была ориентирована как практическое пособие для сельского учителя, однако в связи с внедрением новых информационных и компьютерных технологий во все области образования она может оказаться незаменимым справочным материалом для преподавателя информатики любого уровня, а также использована как дидактическое и научно-практическое пособие.

При подготовке материала мы стремились излагать достаточно доступно и дать преподавателю информатики как можно больше сведений практического характера в виде задач с решениями.

Книга содержит восемь глав. В первой и второй главе рассмотрены задачи, особенности и формы классно-урочных занятий по основам информатики и вычислительной техники. Приведены конкретные примеры проведения классных занятий по информатике.

В третьей главе изложены особенности проведения внеклассных занятий и основные формы внеклассной работы по информатике в школе.

В четвертой и пятой главах приведены материалы, в которых изложены решения занимательных задач, подобраны математические задачи на натуральные числа и задачи для самостоятельного разбора (с решениями и без решений), которые учитель может использовать как в классной, так и во внеклассной работе (факультативной и кружковой работе). Эти главы ориентированы на содержательную линию курса информатики «Алгоритмы и исполнители».

Отметим, что изучение конструкций языков программирования – одна из задач курса информатики. В качестве инструмента для решения математических и физических задач мы специально выбрали язык Бейсик и Паскаль как наиболее распространенные. Решения приведенных задач можно легко перевести на другие языки программирования. Структура урока традиционно состоит из двух частей: первая половина включает изучение нового материала; вторая – отводится формированию умений и отработке навыков составления программ. Ограниченность во времени не позволяет учителю использовать большой объем материала. В связи с этим появляется необходимость изучения лишь программ решения задач, занимающих небольшое количество строк, так называемых ключевых программ, которые позволяют изучить тему урока, закрепить знания и отработать навыки работы с компьютером.

В шестой главе основное внимание уделяется содержательной линии курса информатики «Моделирование и формализация». В ней рассмотрены физические задачи, которые решаются методом математического (компьютерного) моделирования. Вводятся основные понятия компьютерного моделирования и приведены физические задачи, которые можно выполнять в виде лабораторных работ. В структуре каждой темы важное место отведено исследовательской задаче, которая предполагает:

– построение дискретной модели и алгоритма численного решения;

- разработку и отладку программы;
- проведение вычислительного эксперимента и выявление закономерностей физического явления или процесса.

Построение вычислительной установки может быть проведено разными способами:

- прямым программированием на любом языке (Паскаль, Фортан, СИ, Бейсик и др.) и обработкой результатов на графических пакетах Grafer, Origin и др.;
- с использованием пакета Excel – как инструмента для решения физических задач;
- с использованием математических пакетов Maple, MathCAD 8.1 и др.

Содержательная линия курса информатики «Информационные технологии» отражена в седьмой главе. Ее освоение является не только средством подготовки учащихся к жизни в информационном обществе и к будущей профессиональной деятельности, но и средством практического закрепления и развития теоретической подготовки учащихся. Это наиболее простая и доступная большинству учащихся часть курса информатики, составляющая основное содержание профильного курса информатики во многих образовательных учреждениях, оснащенных современными компьютерами. В этой главе изложены технологии решения математических и физических задач с помощью систем Excel, MathCAD, Access. Изложены некоторые приемы и методы моделирования электронных схем и явлений, а также основы алгебры, логики, логических схем и ряд лабораторных работ по основам электроники и вычислительной техники в системе Electronworkbench .

В восьмой главе даны информационные системы, которые необходимы для обработки информации: в графическом пакете Origin; в системе Компас; для автоматизированного перевода из одного языка в другой в системе Promt; для подготовки презентаций в системе PowerPoint; для создания Web-сайтов в системе Front Page; для знания основных сайтов в Internet, необходимых для получения информации в области школьного и профессионального образования.

Авторы выражают признательность д.т.н., с.н.с. Р.И.Саитову, кандидату педагогических наук М.Д.Назаровой и учителю информатики А.Р.Ганеевой за внимательное прочтение книги и ценные замечания.

ГЛАВА 1. КЛАССНО-УРОЧНАЯ ФОРМА ОРГАНИЗАЦИИ ЗАНЯТИЙ

1.1. ЗАДАЧИ И ОСОБЕННОСТИ КЛАССНО-УРОЧНЫХ ЗАНЯТИЙ ПО ОСНОВАМ ИНФОРМАТИКИ И ВЫЧИСЛИТЕЛЬНОЙ ТЕХНИКИ В ШКОЛЕ

В процессе преподавания курса информатики решаются три основные задачи.

Первая связана с усвоением системы понятий, связанных с формированием у учащихся представлений об информатике как о фундаментальной науке.

Вторая задача состоит в том, чтобы учащиеся овладели конкретными навыками использования информационных технологий в различных областях деятельности.

Третья задача связана с формированием у учащихся, кроме общечеловеческих ценностей, таких качеств, как информационная культура, информационная этика, бережное отношение к технике и источникам информации, правовая культура.

Класно-урочная организация обучения сложилась в XVII веке на принципах дидактики, сформулированных Я.Коменским, и до сих пор является преобладающим в школах мира. В постановлении ЦК ВКП (б) от 25 августа 1932 года указывается, что основной формой занятий в школе должен быть урок.

Сформулируем особенности класно-урочной формы организации обучения:

- класно-урочная форма обучения строго регламентирована программой;
- уроки планируются на 45 минут (спаренные на 90 минут);
- урок является основной единицей проведения занятий с учащимися класса;

- классно-урочная форма обучения требует постоянного состава учащихся, объединенных в коллектив по возрастному признаку, с учетом микрорайона жительства;
- класс работает по единому годовому плану и программе согласно расписанию;
- урок посвящен одному учебному предмету, теме, в силу чего учащиеся класса работают над одним и тем же материалом;
- работой учащихся руководит учитель, он же оценивает результаты и фиксирует успеваемость учащихся и прохождение материала в классном журнале.

Атрибуты классно-урочной системы: учебный год, учебный день, расписание уроков, учебные каникулы, перемены, домашнее задание, отметки, классный журнал, дневник успеваемости учащегося, школьные учебники по предметам, школьная программа по предмету, обязательный минимум содержания образования, тематический и календарный планы учителя, санитарно-гигиенические требования к режиму работы в компьютерном классе.

1.2. ФОРМЫ И МЕТОДЫ ОБУЧЕНИЯ

Общеклассные формы: урок, конференция, семинар, лекция, собеседование, консультация, лабораторно-практическая работа, программное обучение, зачетный урок.

Групповые формы: групповая работа на уроке, групповой лабораторный практикум, групповые творческие задания.

Индивидуальные формы: работа с литературой или электронными источниками информации, письменные упражнения, выполнение индивидуальных заданий по программированию или информационным технологиям за компьютером, работа с обучающими программами за компьютером.

Методы обучения: *словесные* – лекция, рассказ, беседа; *наглядные* – иллюстрации, демонстрации как обычные, так и компьютерные; *практические* – выполнение лабораторно-практических работ, самостоятельная работа со справочниками и литературой (обычной и электронной), самостоятельные письменные упражнения, самостоятельная работа за компьютером.

Логический характер применения методов обучения: *индуктивный; дедуктивный; гностический; репродуктивный; поисковый; репродуктивно-поисковый.*

Методы стимулирования учебно-познавательной деятельности: формирование интереса к учению; формирование долга и ответственности в учении.

Методы контроля и самоконтроля: *устный контроль* – фронтальный опрос, индивидуальный опрос, компьютерное тестирование; *письменный контроль* – контрольная работа; выполнение письменных тестовых заданий; письменные отчеты по лабораторно-практическим работам; диктанты по информатике; *лабораторно-практический контроль* – контрольные лабораторно-практические работы; работа с контролирующими программами; *самоконтроль* – устное воспроизведение изученного материала; письменное воспроизведение изученного материала; работа с обучающими программами; компьютерные тесты.

Избираемый темп обучения: быстрый; средний; замедленный.

Большинство форм обучения и методов во взаимодействии педагога с учениками не предстают в так называемом чистом виде. Методы всегда как бы взаимно проникают друг в друга, характеризуя с разных сторон одно и то же взаимодействие педагогов и учащихся. Если говорить о применении в данный момент определенного метода, то это означает, что он детерминирует на данном этапе, внося большой вклад в решение основной дидактической задачи.

Рассмотрим такие традиционно сложившиеся формы классно-урочных учебных занятий, как урок, конференция, семинар, лекция, собеседование, консультация, практическая работа, программированное обучение, зачет.

Урок. Выполняет следующие характерные дидактические функции: сообщение знаний в объеме, определяемом учебными программами; выработка базовых умений, выделенных учебной программой.

Урок является основной формой организации учебных занятий в школе с постоянным составом учащихся и определенным расписанием. Эта форма организации учебных занятий позволяет сочетать работу класса в целом и отдельных групп учащихся с индивидуальной работой каждого ученика. При всем разнообразии форм работы на уроке руководящая роль остается за учителем. Учитель планирует и организует весь учебный процесс по предмету.

Обычно перед уроком учитель ставит не одну, а несколько задач: сообщение учащимся новых знаний, развитие их мышления и

познавательных способностей, формирование научного мировоззрения, привитие практических умений и навыков, повторение ранее пройденного материала, проверка успеваемости (их знаний, навыков, умений) и задачи воспитательного характера.

При всем многообразии решаемых задач в большинстве случаев на каждом уроке можно выделить основную дидактическую, которая обуславливает содержание урока и методы работы учителя с учащимися. В соответствии с этим различают следующие виды уроков: усвоения новых знаний, овладения умениями и навыками, применения знаний, умений и навыков, обобщения и систематизации знаний, проверки и самопроверки знаний, умений и навыков, комбинированный урок по комплексу его основных задач.

Конференция. Характеризуется следующими функциями: расширение и углубление знаний по изученным вопросам; умение работать с источниками информации; выступление с докладом, сообщением; оформление рефератов; воспитание интереса к самостоятельной работе с различными источниками информации (обычной и электронной).

Учебные конференции, как и уроки, проводятся со всем классом в часы, отведенные для предмета по расписанию. Руководящая роль сохраняется за учителем. На конференции, как и на уроке, работа класса в целом сочетается с индивидуальной работой учащихся. Конференции готовят школьников к проведению более сложных форм учебных занятий – лекций и семинаров.

Отличаются конференции от уроков тем, что новые знания школьники приобретают из литературы (обычной и электронной), с которой работали в процессе подготовки к конференции, и из докладов других учащихся. Руководящая роль учителя на конференции заключается в том, что он организует выступление учащихся с докладами и их обсуждение, вносит дополнения и исправления к докладам, если это не сделано во время обсуждения. Он обобщает результаты конференции, оценивает работу класса в целом и отдельных учеников, выступавших с докладами и дополнениями к ним.

Образовательное значение конференций состоит в том, что в процессе подготовки к ним школьники приобретают навыки самостоятельной работы с литературой, электронными источниками информации, могут применять полученные знания для решения конкретных задач, поставленных перед ними.

Проведение конференций способствует выявлению склонностей и способностей учащихся, развитию у них интересов к научным и техническим знаниям.

На конференции можно выносить вопросы, связанные с историей развития информатики, применением изучаемого теоретического материала, обобщением и систематизацией знаний, с принципами устройства и работы компьютеров и др.

При подготовке к конференции учитель обязан:

- определить задачи, круг обсуждаемых вопросов, время проведения;
- подобрать литературу для учащихся;
- распределить темы докладов между учениками, выделить главные этапы работы;
- консультировать учеников по ходу подготовки докладов и проверять их готовность.

План конференции объявляется заранее.

Семинар. Выполняет следующие функции: систематизация и обобщение знаний по изученному вопросу, теме, разделу (в том числе в нескольких учебных курсах); совершенствование умений работать с дополнительными источниками, сопоставлять изложение одних и тех же вопросов в различных источниках информации; умений высказывать свою точку зрения, обосновывать ее; писать рефераты, тезисы и планы докладов и сообщений, конспектировать прочитанное.

Семинары организуют с целью повторения, систематизации и уточнения полученных знаний, развития умения применять знания при решении задач. Руководящая роль учителя в этом случае сводится в основном к разъяснению цели, задач и плана семинара, выдаче индивидуальных заданий и проведению консультации в связи с подготовкой учащимися рефератов, сообщений; всем ученикам указывается минимум литературы и вопросы, на которые они должны ответить. В плане семинара обычно указывают основные вопросы, подлежащие рассмотрению; литературу, рекомендуемую всем и отдельным докладчикам; формы работы на занятии.

При подготовке семинара первостепенное значение приобретает дифференцированный подход к учащимся, а при его проведении – обеспечение активного участия всех в обсуждении вынесенных на семинар вопросов.

По способу проведения различаются следующие семинары: собеседование, обсуждение рефератов и докладов, решение задач; семинары смешанного и комплексного характера, цель последних – обобщение и систематизация знаний учащихся по смежным предметам (математика, физика).

Лекция. Характеризуется следующими функциями: создание представления обзорного характера по какой-то теме или проблеме; систематизация и обобщение знаний по теме или разделу; выработка умения конспектировать лекцию.

Учащиеся, слушая лекции, воспринимают и осмысливают информацию, сообщаемую педагогом. При лекционном изложении материала школьники не имеют возможности проявить инициативу. В этом заключается один из существенных недостатков данной формы обучения. К недостаткам относится и то, что в процессе изложения преподаватель в некоторой мере лишен возможности судить, насколько правильно и хорошо идет усвоение. Только закончив изложение, путем ряда контрольных вопросов учитель может определить, как усвоена тема. Лекционное изложение материала, как правило, длится часть урока и только в некоторых случаях целый урок. Школьная лекция, как правило, всегда заканчивается ответами учителя на возникшие у ребят вопросы.

Собеседование. Выяснение того, что усвоено из основного материала, выявление пробелов в знаниях и внесение коррективов в знания; стимулирование систематической и самостоятельной работы.

Консультация. Устранение пробелов в знаниях и умениях; уточнение усвоенного; ответы на вопросы, возникшие в процессе учебной работы и оказание помощи в овладении разными видами учебной и практической деятельности.

Лабораторно-практическая работа. Формирование у школьников умения обращаться с компьютером и внешними устройствами, пользоваться прикладными программами, составлять программы. Особенностью практической работы является ограничение во времени, определенное СанПиН 2. 2. 2. 542-96.

Приведем примерный план проведения практической работы:

- Определение темы практической работы и целей;
- Определение умений и навыков, которые предполагаются привить учащимся в ходе выполнения практической работы.
- Теоретическая часть, предшествующая практической работе.
- Объяснение хода выполнения работы.

- Непосредственное исполнение работы.
- Составление отчета о практической работе.
- Критерии оценки практической работы.
- Подведение итогов.

Основным в выполнении практических работ является использование полученных знаний и навыков в самостоятельной работе с компьютером, внешними устройствами, прикладными программами, а также ввод, редактирование и отладка программ.

Программированное обучение. Управляемое усвоение программируемого учебного материала с помощью компьютера и обучающих программ. Программируемый учебный материал представляет собой серию небольших порций учебной информации, подаваемых в определенной логической последовательности.

При программированном обучении прежде всего определяют цели и задачи, четко выделяют то, что учащийся должен знать, понимать, уметь: анализируют логическую систему курса, исключают все аналогичное, второстепенное. Затем выделяют основные темы, разделы и подразделы, которые дробят на дозы – кванты информации, уменьшение которых невозможно без ущерба смысловому содержанию. Содержание каждого последующего кванта базируется на информации, содержащейся в предыдущих квантах. Размер кванта информации определяется характером материала, уровнем развития учащихся.

Благодаря немедленной обратной связи удается устранить лишние затраты и более быстрыми темпами добиваться усвоения материала. Информация о правильности ответа после усвоения каждого кванта имеет большое психологическое значение. Это создает у учащихся уверенность в своих силах и повышает интерес к предмету.

Темп подачи информации согласуется с индивидуальными способностями. Каждый учащийся расходует на усвоение материала столько времени, сколько ему необходимо, то есть процесс обучения удастся максимально индивидуализировать.

Однако программированное обучение имеет серьезные недостатки. Дробление учебного материала на кванты и невозможность продвижения вперед при условии, когда какой-то квант не усвоен, лишает ученика видеть перспективу в развитии изучаемого материала, его многочисленные связи и отношения. Весьма затрудни-

тельным является также обеспечение целостности восприятия учащимися всего материала.

Зачет. Зачетный урок предназначен не только для контроля знаний и умений, а прежде всего для обучения, развития и воспитания каждого учащегося посредством индивидуальной работы.

Зачет проводится по целой теме или разделу. Он призван проверить усвоение теоретических основ изучаемой темы, умения и навыки использования теории. В зачет включается тот материал, которым должны владеть все ученики. Существенно, чтобы в ходе зачета можно было установить наличие знаний, умений и навыков, которые необходимы школьникам для изучения последующих тем. Кроме того, целесообразно включать такой материал, который входит в программу выпускных и вступительных экзаменов, так как одна из целей принятия зачета – подготовка школьников к экзаменам.

1.3. ОСОБЕННОСТИ РЕШЕНИЯ ЗАДАЧ ПО ИНФОРМАТИКЕ

Специфика задач по программированию – это взаимосвязь технологии программирования со специальными дисциплинами одного, двух или целого комплекса дисциплин. Обусловлено это содержанием задач, которое включает практическую и прикладную направленность программирования. Многие задачи исторически были решены средствами программирования. Часть задач решает политехническую направленность обучения.

Исходя из вышеизложенного, необходимым условием решения задач по программированию является: наличие у учащихся системы знаний, умений и навыков, приобретенных ими в процессе изучения специальных (предметных) дисциплин; знание общенаучных методов различных наук; умение переноса знаний, сформированных на одном предмете, в другой; комплексное использование фундаментальных знаний различных предметов. Назовем их общеучебными знаниями, умениями и навыками.

Наличие у учащихся общеучебных знаний, умений и навыков не является достаточным условием успешного решения задач по программированию. Без знания, умения и навыков использования основных структур организации программ (алгоритмов), их комбинации нельзя решать задачи по программированию. Необходимым условием является: развитое алгоритмическое мышление; знания, умения и навыки технологии программирования. Только

тесная взаимосвязь двух составляющих является необходимым и достаточным условием успешного обучения решению задач по программированию.

Говоря о методике обучения решению задач по программированию, выделим способы обучения:

- *Обучение принципам* – необходимые отношения заранее сообщают учащемуся в виде общих принципов, правил или алгоритмов;
- *Обучение на примерах* – существенные признаки обнаруживают сами учащиеся в ходе осмысления данных и оперируя ими;
- *Обучение способам мышления* – учащегося обучают приемам находить признаки, с помощью которых обнаруживаются необходимые отношения данных. В этом случае учащийся сам обнаруживает существенные отношения данных, если его вооружают необходимыми способами мыслительной деятельности.

Обучение принципам дает лучшие результаты усвоения понятий. Обучение на примерах несколько эффективнее для запоминания. Обучение способам помогает переносу умственных навыков, т.е. эффективно развивает мышление. Таким образом, обучение должно сочетать в себе все указанные способы, если ставит целью оптимально решать стоящие задачи.

Обучение решению задач должно следовать методам научного познания. Методы индукции и дедукции занимают ключевое место в обучении и являются одним из факторов успешного решения задач.

Индукция – переход от частных случаев к общему выводу.

Полную *индукцию* можно представить следующей схемой:

- A_1, A_2, \dots, A_n , суть B ;
- A_1, A_2, \dots, A_n составляют множество A .
- Следовательно, все элементы A суть B .

В качестве подсобного приема в процессе индуктивного решения большую роль играет аналогия: переход от одного частного случая к другому делается на основании подмечаемого их сходства (анalogии). Заключение по *анalogии* можно представить следующей схемой:

- A обладает признаками c_1, c_2, \dots, c_n .
- B обладает теми же признаками c_1, c_2, \dots, c_n .
- A обладает признаком d . Вероятно, и B обладает признаком d .

Дедукция – это переход от общих положений к частным примерам и конкретным положениям. Отличительные черты дедукции:

- с помощью дедукции устанавливается принадлежность свойств объекту или принадлежность объекта классу с большим объемом;
- делается вывод от общего к подчиненному ему общему или частному;
- делаются достоверные выводы, если посылки верны и умозаключения правильны.

На начальном этапе обучения решению задач по программированию очевидно преобладание индуктивного метода.

Этапы обучения решению задач от частного к общему (индуктивный метод).

1. Решение задач учителем (составления алгоритма).
2. Ручное исполнение учителем алгоритма.
3. Воспроизведение учащимися составления программы (алгоритма) решения задачи и ручное исполнение программы.
4. Выполнение аналогичных упражнений.
5. Модификация решения задач.
6. Составление аналогичных задач по методам решения.
7. Составление аналогичных задач по содержанию.
8. Обобщение условия задачи.
9. Обобщение решения задачи.

По мере накопления технологических знаний, умений и навыков (знание основных конструкций составления алгоритмов, умение составлять алгоритмы их исполнять), освоив основные принципы метода индукции, можно приступить к освоению метода дедукции.

Этапы обучения решению задач от общего к частному (дедуктивный метод).

1. Решение общей задачи. Исполнение алгоритма.
2. Составление задач на развитие данной темы для частных случаев. Решение составленных задач. Исполнение алгоритмов.
3. Поиск общей задачи для исходной задачи, и использование решения общей задачи (модификация решения) для решения исходной задачи. Исполнение алгоритма.

Метод индукции реализован в параграфе «Решение математических задач методами обобщения и аналогии».

Метод дедукции реализован в параграфе «Разложение натурального числа на натуральные слагаемые».

1.4. ОСОБЕННОСТИ РАБОТЫ И ОФОРМЛЕНИЕ КАБИНЕТА ИНФОРМАТИКИ

Кабинеты информатики и вычислительной техники создаются в образовательных учреждениях в соответствии с постановлениями Совета Министров от 12 апреля 1984 г. №313, от 28 марта 1985 г. №271, решением Министерства образования России от 22 февраля 1995 г. №4/1.

Кабинет информатики – учебно-воспитательное подразделение, оснащенное комплексом вычислительной техники, учебно-наглядными пособиями, учебным оборудованием, мебелью, оргтехникой и приспособлениями для проведения теоретических и практических, классных, внеклассных занятий.

Расположение оборудования и оформление кабинета информатики зависит от многих факторов: планировки здания, количества классных комнат, финансирования, наличия стандартных пособий и др. Хотелось бы поделиться некоторыми особенностями оформления кабинета информатики в школе №104 г.Уфы.

Над классной доской расположены портреты видных отечественных и зарубежных ученых, внесших вклад в создание и развитие различных областей информатики (рис.1).



Рис. 1

Портреты ученых – Джорджа Буля, С.А.Лебедева, Клода Шеннона, Джона фон Неймана, А.А. Ляпунова, А.Н. Колмогорова, А.П. Ершова (выполнены учителем рисования). Над портретами высказывание А.П. Ершова: «Программирование – вторая грамотность». Данная часть оформления кабинета выполняет воспитательную функцию.

В кабинете информатики организован уголок-музей истории вычислительной техники в школе (рис. 2). Перечислим экспонаты, выставленные в уголке: счеты, логарифмическая линейка, арифмометр, компьютер «Башкирия-2М», компьютер IBM-286. Выставленные экспонаты функционирующие. В уголке хранится и программное обеспечение для выставленных компьютеров. При прохождении темы «История развития вычислительной техники» изучаются не только экспонаты, но и сравнивается программное обеспечение.



Рис. 2

Стенд «Архитектура ЭВМ» (рис.3). На стенде представлена схема магистрально-модульного принципа архитектуры ЭВМ. Центральное место занимают комплектующие части компьютера: материнская плата, контроллеры, видеокарта, винчестер и другие накопители информации.

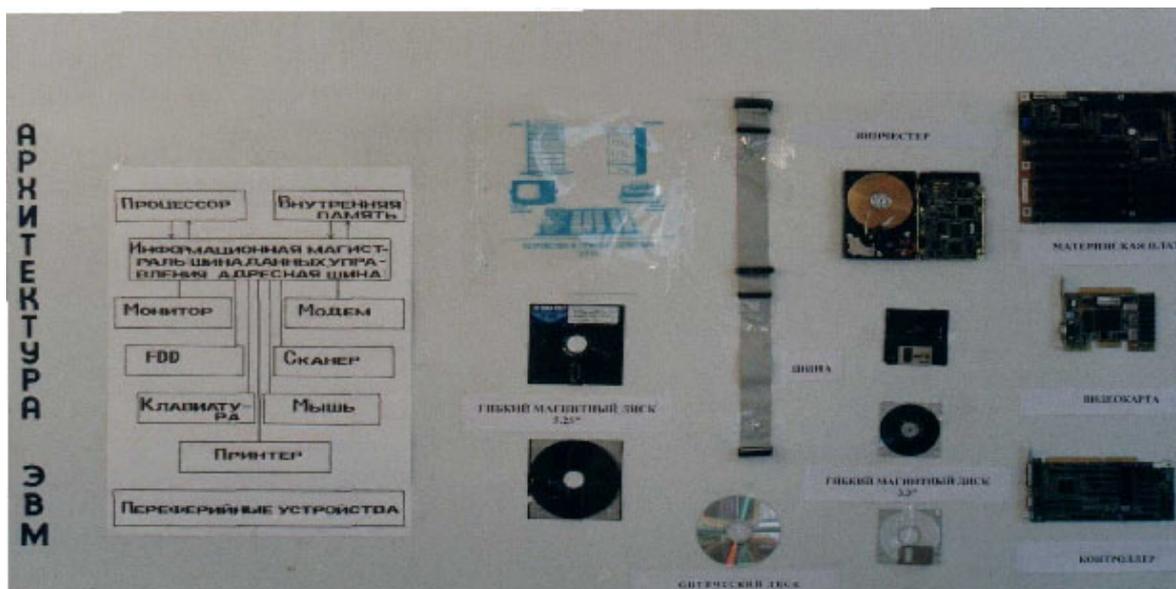


Рис. 3

Стенд «Программирование» (рис.4). На стенде представлены: технология подготовки и решения задач на компьютере, основные алгоритмические конструкции на школьном алгоритмическом языке и языке Бейсик, приведены примеры решения задач.



Рис. 4

На рис. 5 приведена схема расположения компьютеров и рабочих столов. Посреди кабинета расположены столы для ведения

обычных занятий по информатике. Компьютеры расположены вдоль стен по противоположным сторонам или буквой П.



Рис. 5

Библиотечка кабинета информатики. Умение пользоваться научно-технической литературой, пожалуй, одно из приоритетных направлений работы учителя. Как же руководить чтением учащихся? Прежде всего необходимо, чтобы в кабинете информатики и школьной библиотеке была необходимая литература как периодическая, так и непериодическая. Примерами периодических изданий являются «Информатика и образование», «Информатика», «Компьютер в школе», «Компьютерра», «Мир ПК», «Компьютер пресс» и др. Комплектование библиотеки кабинета информатики зависит от тех задач, которые стоят перед учителем. Непериодическая литература должна постоянно пополняться и обновляться.

В кабинете информатики должны быть вывешены списки доступной литературы, рекомендуемой для дополнительного чтения по отдельным темам, для подготовки рефератов, для подготовки к экзаменам.

Интерес к научной, научно-популярной, технической литературе у учащихся повышается, если учитель систематически использует в учебной работе дополнительную литературу, делает на уроках ссылки на отдельные книги и журналы, зачитывает отрывки из них. Следует всячески поощрять учащихся, использующих дополнительный материал.

Передвижной демонстрационный компьютерный комплекс. Для успешного преподавания учебных дисциплин необходимо применение на уроках наглядных пособий. Наглядные пособия создают у учащихся образное представление явлений, процессов, активизируют мышление, оживляют учебный процесс, поддерживают внимание и интерес к изучаемому материалу и тем самым способствуют лучшему его усвоению.

К наглядным пособиям относятся модели, макеты, графические пособия, таблицы, плакаты, диаграммы, графики, рисунки, диа- и эпипроекции, учебные кинофильмы и др. Многообразие технических средств (диапроектор, кодоскоп, эпидиаскоп, кинопроектор и др.) хотя и обогащает использование наглядности, но создает определенные проблемы (к примеру, обслуживание технических средств).

С течением времени наглядные пособия ветшают, традиционный комплекс технических средств обучения требует ремонта, порой дорогостоящего, а иногда и не подлежит ремонту, и тогда накопленный материал может быть безвозвратно утерян. Компьютерная технология позволяет обновить, «вдохнуть» новую жизнь в использование наглядных пособий, а также изготовление новых. Обновление подразумевает создание электронных аналогов – мультимедийные документы. Наличие мультимедийного комплекса (аппаратного и программного) позволяет успешно и качественно решить вопрос обновления и изготовления наглядных пособий. При отсутствии мультимедийного комплекса можно обойтись минимумом.

К минимальным аппаратным средствам изготовления наглядных пособий отнесем: персональный компьютер Pentium, сканер, принтер. К минимальным программным средствам: операционная система Windows-95, стандартные приложения Windows, текстовый процессор Microsoft Word, электронная таблица Microsoft Excel, программа преобразования бумажного документа в электронную

форму, программы сканирования и распознавания текста FineRider, графический редактор CorelDraw, программы фотообработки Adobe Photoshop.

Минимальный демонстрационный компьютерный комплекс может включать системный блок, клавиатуру, мышь, проектор или в отсутствие проектора монитор диагональю 17 дюймов (лучше 19).

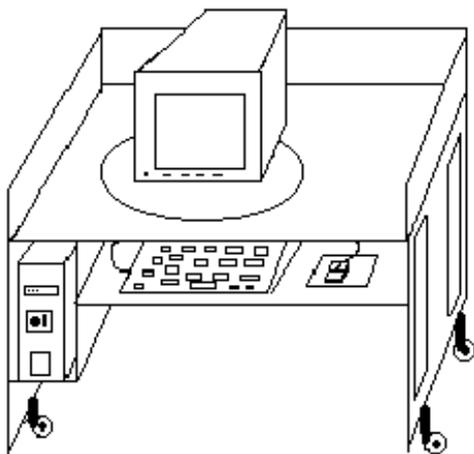


Рис.6

Использование обновленных и изготовленных наглядных пособий наталкивается на существенную трудность – отсутствие в предметных кабинетах компьютеров, невозможность использовать кабинет информатики, ведь во многих случаях демонстрация наглядных пособий является лишь фрагментом учебного процесса, где могут проводиться к примеру, натуральные опыты и демонстрации. Конечно, демонстраци-

онный комплекс можно переносить и по частям, но возникают свои проблемы: время, надежность, проблемы с установкой и др. Выходом из этого положения является использование передвижного демонстрационного компьютерного комплекса. Самым простым и эффективным является доработанный обычный компьютерный стол. Доработка заключается в оснащении компьютерного стола специальными колесиками, которые вращаются и фиксируются. Аппаратные средства закрепляются. При закреплении используются резиновые или поролоновые подушки. Дополнительно можно использовать выдвижные или складные ручки (рис.6) для переноса комплекса (например, с этажа на этаж).

1.5. УРОК ТЕХНИКИ БЕЗОПАСНОСТИ

Цель занятия.

Ознакомить учащихся:

- с вредными воздействиями компьютера на здоровье пользователя;
- с комплексом мероприятий по предупреждению профзаболеваний (организация рабочих мест, комплексы упражнений для глаз, организация упражнений физкультурных минуток и т.д.).

Инструктаж по технике безопасности (роспись учащихся об ознакомлении). Подготовка учащихся к экстремальным случаям: противопожарный инструктаж, медицинский инструктаж.

Тема занятия «Вредные воздействия компьютера на здоровье пользователя».

Исследования последних лет показали, что при работе на персональных компьютерах наблюдается рост профзаболеваний. Например, в США 66% всех профессиональных заболеваний вызвано компьютерами, а 75% всех пользователей страдают от болей в глазах и опорно-двигательном аппарате. Такое положение дел вызывает определенную озабоченность как среди представителей здравоохранения, так и среди самих пользователей. Перечислим некоторые болезни и факторы, влияющие на них.

RSI-синдром – хроническое заболевание связок, проявляющееся в болях запястья от многократных нажатий кнопки мыши. Эта болезнь является следствием использования неправильно оборудованного, с точки зрения эргономики, рабочего места пользователя персонального компьютера. Она возникает при умственном и физическом перенапряжении и представляет собой хроническое заболевание, которое может незаметно развиваться на протяжении нескольких лет.

К сожалению, еще не существует технических, эргономических или медицинских мер, обеспечивающих оптимальные условия работы. Тем не менее создание эргономических устройств и приспособлений является первым шагом в этом направлении.

Для предотвращения развития болезни RSI можно воспользоваться программой Mouse Tool 3.1. Ее действие заключается в том, что после прекращения перемещения мыши она выполняет щелчок вместо пользователя.

Отравляющие вещества. Новый корпус компьютера и монитор источают своеобразный слабый запах, но гораздо большую опасность представляют газы, не имеющие запаха – диоксины и фуран, отравляющие окружающую среду. Они относятся к так называемым *противопожарным добавкам*, которые содержатся в корпусе монитора и платах системного блока компьютера.

Озон возникает в результате действия электрических зарядов, возникающих в лазерных принтерах. Проникающий везде газ раздражает слизистую оболочку носа, глаз и горла. Длительные

опыты над животными, которые проводились в США, выявили устойчивую зависимость между озоном и раковыми заболеваниями.

Существует единственный способ свести до минимума концентрацию вредных веществ – регулярное проветривание комнаты.

Электросмог. Персональный компьютер является источником как высокочастотных, так и низкочастотных полей. Электромагнитные волны низкой частоты излучаются блоком питания и вентилятором, а процессор и некоторые элементы компьютера излучают высокочастотные колебания с частотой в несколько гигагерц. Наибольший электросмог возникает при работе блока питания и вентилятора, то есть источников низкочастотных колебаний.

Чтобы избежать влияния электросмога на организм человека, рекомендуются следующие меры предосторожности:

- правильное заземление всех элементов компьютера (не только системного блока, но и монитора и принтера);
- все устройства должны иметь закрытые разъемы;
- если устройство не работает, то его нужно отключить от электросети;
- кабели от компьютера, мыши, клавиатуры и других устройств, сетевые блоки питания, разъемы должны располагаться как можно дальше от пользователя;
- следует использовать устройства, имеющие энергосберегающий режим работы.

Влияние монитора на зрение. Всем известно, что многочасовая работа за экраном монитора вызывает жжение в глазах, слезы, покраснение, дрожание век. Однако не всем известно, что каждому второму пользователю при длительной работе у экрана монитора грозит опасность испортить зрение. За время восьмичасового рабочего дня пользователь бросает до 30 000 взглядов на экран. Глаз, работающий с перегрузкой, не может адаптироваться к таким условиям. Нерезкое изображение и мерцание на экране повышают угрозу потери зрения. Существует угроза заболевания Сикка: роговица высыхает и мутнеет.

Цветной шрифт увеличивает нагрузку на зрение, поскольку составляющие цветов имеют различные длины волн и видны на разном расстоянии. При этом глаз нуждается в более точной адаптации, чем при черно-белом изображении.

Меры предосторожности:

- пользуйтесь шрифтами размером более 7 пунктов, частота регенерации (развертки) монитора должна быть не менее 70 Гц ;
- в качестве источников освещения используйте лампу дневного света или биологические лампы, свет которых по спектру волн близок к естественному освещению;
- при цветном экране количество цветов должно быть не менее 256;
- графическую карту нужно установить в режиме отображения 800*600 точек; чтобы при этом не было мерцания монитора, частота регенерации (обновления) должна составлять не менее 75 Гц;
- размер зерен монитора – не более 0,28 мм;
- не должно быть бликов на экране монитора; если нет возможности изменить освещение, используйте антибликовые экраны;
- делайте паузы во время работы за монитором;
- при работе с текстом в качестве фона предпочтительнее использовать белый цвет и черные символы для лучшего восприятия текста.

Допустимая продолжительность рабочего времени. Согласно СанПиН 2.2.2.542-96 (Санитарные правила и нормы) для преподавателей высших и средних специальных учебных заведений, учителей общеобразовательных школ устанавливается длительность работы в дисплейных классах и кабинетах информатики и вычислительной техники не более 4 часов в день.

Время непосредственной работы учащихся за компьютерами зависит от их возраста, но не должна превышать:

- для учащихся 1 класса – 10 минут;
- для учащихся 2-5 классов – 15 минут;
- для учащихся 6-7 классов – 20 минут;
- для учащихся 8-9 классов – 25 минут;
- для учащихся 10-11 классов на первом часу учебных занятий – 30 минут, на втором – 20 минут;
- для студентов первого курса оптимальное время составляет 1 час, для студентов старших курсов – 2 часа.

Занятия в кружках с использованием компьютеров должны проводиться не чаще 2 раз в неделю общей продолжительностью:

- для учащихся 2-5 классов (7-10 лет) не более 60 минут;
- для учащихся 6 классов и старше – до 90 минут.

Для предупреждения переутомления обязательным мероприятием являются:

- проведение упражнений для глаз через каждые 20-25 минут работы за компьютером;
- перерыв после каждого академического часа занятий, независимо от учебного процесса, длительностью не менее 15 минут;
- сквозное проветривание с обязательным выходом всех из помещения;
- проведение физкультурных упражнений в течение 3-4 минут.

Примерный вариант упражнений для глаз

Закрывать глаза на несколько секунд, сильно напрягая глазные мышцы, затем раскрыть их, расслабив мышцы глаз. Дыхание ритмичное. Повторить 4-5 раз.

Посмотреть на переносицу и задержать взор в течение трех полных ритмичных дыханий. Затем посмотреть вдаль. Повторить 4-5 раз.

Не поворачивая головы, посмотреть направо и зафиксировать взгляд на несколько секунд, затем посмотреть вдаль прямо. Аналогичным образом проводятся упражнения, но с фиксацией взгляда влево, вверх, вниз. Повторить 4-5 раз.

Часто открывать и закрывать глаза (моргать) в течение 20-30 секунд.

Примерный вариант комплекса упражнений физкультурных минуток.

1. Исходное положение – ноги врозь, руки вперед. Поворот туловища направо, мах левой рукой вправо, правой – назад за спину. То же самое в другую сторону. Упражнения выполняются размашисто, динамично. Повторить 6-8 раз.

2. Согнуть правую ногу, обхватив голень руками, притянуть ногу к животу, ногу опустить, руки вверх – в стороны. То же самое с левой ногой. Повторить 6-8 раз.

Инструктаж по технике безопасности. Правила поведения в кабинете информатики:

- в кабинет информатики входить с разрешения учителя;
- при входе в кабинет не толкаться в дверях, спокойно занимать свое рабочее место, ничего не трогать на столе;
- в кабинет входить без верхней одежды и со сменной обувью;

В кабинете запрещено:

- трогать разъемы соединительных проводов;
- прикасаться к проводам питания;
- прикасаться к экрану и задней стенке монитора;
- работать на клавиатуре при выключенном напряжении;
- работать на ЭВМ во влажной одежде и с мокрыми руками;
- класть вещи на составные части ЭВМ.

Перед началом работы на ЭВМ:

- убедиться в отсутствии видимых неисправностей ЭВМ, в случае их обнаружения сообщить учителю;
- записать в журнале учета машинного времени дату и время начала работы, свою фамилию и класс, замеченные перед началом работы неисправности;
- приступить к работе по указанию учителя.

По окончании работы на ЭВМ:

- привести свое рабочее место в порядок, выключить ЭВМ;
- записать в журнале время окончания работы и все замеченные в процессе работы неисправности;
- сдать выданные для работы дискеты учителю.

На уроке необходимо провести инструктаж по пожарной безопасности и инструктаж по оказанию первой медицинской помощи.

ГЛАВА 2. ПРИМЕРЫ КЛАССНО-УРОЧНОЙ ФОРМЫ ОРГАНИЗАЦИИ ЗАНЯТИЙ

Рассмотрим теперь более подробно ряд уроков, имеющих различную направленность. У учащихся при решении многих математических задач на уроках информатики возникают затруднения. Причина кроется в том, что математические факты, теоремы, определения, свойства быстро забываются. Поэтому перед решением таких задач целесообразно напомнить общепринятые обозначения, математические теоремы, свойства, на конкретных примерах показать общие приемы и методы, используемые при решении математических задач.

2.1. ИНТЕГРИРОВАННЫЙ УРОК «ПРИБЛИЖЕННОЕ ВЫЧИСЛЕНИЕ ИНТЕГРАЛА»

В последнее время идет сокращение основных часов математики и информатики. Один из способов решения данной проблемы, на наш взгляд, состоит в интеграции уроков математики и информатики. Для усиления практического и прикладного направления обучения ниже предлагается интегрированный урок алгебры и информатики. Урок планируется провести в 11 классе после изучения темы «Интеграл».

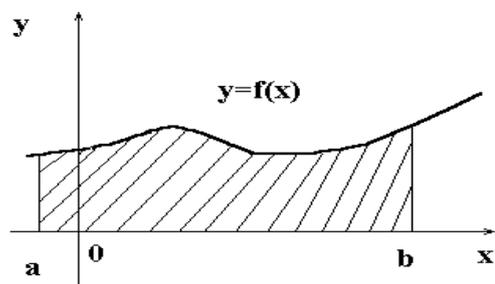


Рис. 7

В начале урока напоминает-ся, что определенный интеграл – это площадь криволинейной трапеции, ограниченной линиями (рис.7). Для приближенного вычисления определенного интеграла используются различные методы.

1. Метод трапеции. Рассмотрим следующую сумму:

$$S=(b-a)/n * (1/2f(x_0)+f(x_1)+.....+f(x_{n-1})+1/2f(x_n)),$$

слагаемые которых равны площадям трапеций, «вписанных» в криволинейную трапецию и ограниченной ломанной, как это изображено на рис.8.

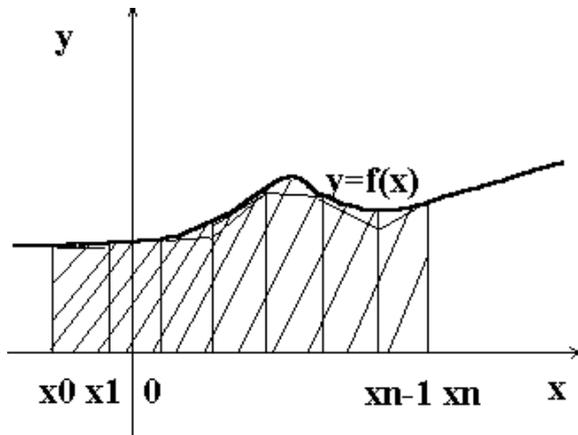


Рис. 8

Вспоминаем материал «Приближенное вычисление интеграла методом трапеций» из курса ОИВТ (учебное пособие «Основы информатики и вычислительной техники» [37].

На доске записываем алгоритм приближенного вычисления площади криволинейной трапеции методом трапеции.

```

алг вещ площадь (арг вещ a,b,цел n)
  нач вещ S,h
  h:=(b-a)/n
  s:=(f(a)+f(b))/2
  x:=a
  нц n-1 раз
    x:=x+h
    s:=s*f(x)
  кц
  знач:=s*h
кон
  
```

По данному алгоритму составляем программу вычисления на языке Бейсик или Паскаль. Ниже приведена программа на этих языках¹.

¹ В дальнейшем не акцентируется внимание на дизайне и оформлении программ. Однако заметим, что последний оператор READKEY или READLN позволяет просмотреть выполнение программы без вылета его в среду Паскаль.

Программа 1.1	Программа 1.2
<pre> 10 REM ПЛОЩАДЬ 20 INPUT «A,B,N»; A,B,N 30 DEF FN F(X)= 40 H=(B-A)/N:S=FN F(A)/2+FN F(B)/2:X=A 50 FOR I=1 TO N-1 60 X=X+H 75 S=S+FN F(X) 80 NEXT I 90 PRINT «ПЛОЩАДЬ S=»;S*H 100 END </pre>	<pre> Program P1_2{Вычисление площади} Uses crt; Var n,i: Integer; a,b,dx,x,s: Real; function f(x:real):real; begin Clrscr; {очистка экрана} f:= ; end; begin write('Введите a,b,n='); readln(a,b,n); dx:=(b-a)/n; x:=a;s:=f(a)/2+f(b); for i:=1 to n-1 do begin x:=x+dx; s:=s+f(x); end; s:=dx*s; writeln('Площадь S=',s:6:2); readkey; end. </pre>

Выбор вида функции в программах 1.1 и 1.2 предоставляется учащемуся.

Уточняем, что $A=a$, $B=b$, N – число делений отрезка $[a,b]$. В строках определения функций справа от знака равенства записываем функцию, которая задает криволинейную трапецию. Отмечаем, что точность приближения зависит от N .

Учитель записывает на доске задание: вычислить интеграл $\int_0^3 \sqrt{x+1} dx$. Приводит решение этого задания на доске, используя формулу Ньютона-Лейбница.

$$\int_0^3 \sqrt{x+1} dx = \int_0^3 (x+1)^{0.5} d(x+1) = [2/3 * (x+1)^{(3/2)}]_0^3 = 2*4^{(3/2)}/3 - 2*1^{(3/2)}/2 = 14/3 = 4. \quad (6)$$

Далее учитель формулирует эту же задачу по-другому: вычислить площадь фигуры, ограниченной линиями: $y = \sqrt{x+1}$, $x=0$, $x=3$, $y=0$. Указывает на равнозначность формулировок. На компьютере вычисляет приближенное значение площади. Ниже приведена программа и результат вычисления.

Программа 2.1	Программа 2.2
<pre> 10 REM ПЛОЩАДЬ 20 INPUT»A,B,N»;A,B,N 30 DEF FN F(x)=SQR(X+1) 40 H=(B-A)/N:S=FN F(A)/2+FN F(B)/2:X=A 50 IFOR I=1 TO N-1 60 X=X+H 75 S=S+FN F(X) 80 NEXT I 90 PRINT «ПЛОЩАДЬ S=»;S*H 100 END RUN A,B,N? 0,3,100 ПЛОЩАДЬ S=4,666648 </pre>	<pre> Program P1_2 uses crt; var n,i: Integer; a,b,dx,x,s: Real; function f(x:real):real; begin clrscr; {очистка экрана} f:=sqrt(x+1); end; begin write('a, b, n ='); readln(a,b,n); dx :=(b-a)/n; x:=a;s:=f(a)/2+f(b); for i:=1 to n-1 do begin x:= x+dx; s:= s+f(x); end; s:= dx*s; writeln('Площадь S=', s:6:2); readkey; end. </pre>

Сравнивают полученные значения.

Учитель предлагает приступить к практической части урока. Класс делится на две группы. Одна группа учащихся вычисляет определенный интеграл по формуле Ньютона-Лейбница. Другая группа, используя и уточняя приведенную программу с помощью компьютера, вычисляет приближенное значение интеграла. Для этого можно использовать номера задач 357, 358, 360, 361 из учебника «Алгебра и начала анализа. Учебник для 10-11 классов» под редакцией А.Н. Колмогорова (М.: Просвещение, 1990). Выполнив часть заданий, группы сравнивают результаты.

Меняются заданиями. После выполнения заданий учитель делает вывод, что для вычисления интеграла можно использовать приближенный метод, рассмотренный на данном уроке. Далее он говорит, что не всегда можно вычислить интеграл, воспользовавшись формулой Ньютона-Лейбница, существуют так называемые «не берущиеся интегралы», тогда можно воспользоваться приведенным выше приближенным методом вычисления определенного интеграла. Даются несколько таких примеров. Ребята находят приближенные значения приведенных примеров.

2. Метод Монте-Карло.

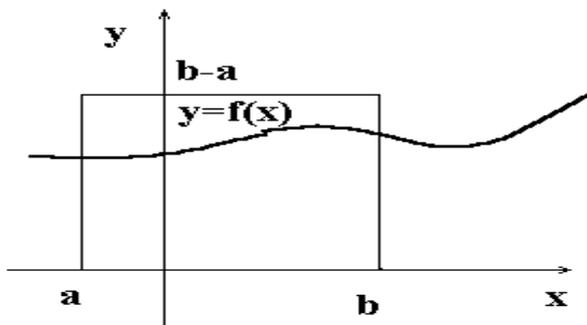


Рис. 9

Впишем криволинейную трапецию в квадрат (рис.9). Если такой чертеж мы некоторое время подержим под дождем, то на его поверхности останутся следы капель. Подсчитаем число следов капель внутри квадрата и внутри криволинейной трапеции.

Очевидно, что их отношение приближенно равно отношению площадей квадрата и криволинейной трапеции, так как попадание капель дождя в различные места чертежа равновероятно.

n – число капель в квадрате, m – число капель в криволинейной трапеции, тогда

$$\text{Скр.тр.} = \text{Скв.} * m/n.$$

Теперь заменим дождь генератором случайных чисел. $RND(1)$ генерирует случайные числа в промежутке от 0 до 1. $RND(1)*(b-a)$ генерирует случайные числа в промежутке от 0 до $b-a$. $RND(1) * (b-a) + a$ генерирует случайные числа в промежутке от a до b .

Тогда координаты случайной точки (координаты «капли дождя») можно написать следующим образом:

$$X=RND(1)*(b-a)+a \quad Y=RND(1)*(b-a).$$

Если выполняется условие $f(x) \geq Y$, тогда случайная точка лежит внутри криволинейной трапеции. Генерируя n случайных точек, выделив m точек, попавших внутрь криволинейной трапеции, можем воспользоваться приведенной выше формулой для приближенного вычисления площади криволинейной трапеции. Приведенный алгоритм реализован в программе 3 для вычисления

$$\int_0^3 \sqrt{x+1} dx$$

Программа 3.1	Программа 3.2
<pre> 10 REM ПЛОЩАДЬ 15 REM МЕТОД МОНТЕ-КАРЛО 20 M=0 30 INPUT «A,B,N»; A,B,N 40 FOR I=1 TO N 50 X(I)=RND(1)*(B-A)+A 60 Y(I)=RND(1)*(B-A) 70 IF SQR(X(I))>=Y(I) THEN M=M+1 80 NEXT I 90 PRINT «ПЛОЩАДЬ S=«;(M/N)* (B-A)^2 100 END </pre>	<pre> Program P3_2 {Вычисление площади} Uses crt; Var n,i,m: Longint; a,b,x,y,s: Real; begin Clrscr; {очистка экрана} writeln('Vvedi: a,b,n', a,b,n); readln(a,b,n); M := 0; For i := 1 to n do begin x:=Random(30000)/30000*(b- a)+a; y:=Random(30000)/30000*(b-a); if sqrt(x+1)>=y then m := m+1; end; s:=m/n*sqr(b-a); writeln('Площадь = ', s:10:5); readkey; end. </pre>

Учитель отмечает, что чем больше N , тем выше точность вычисления.

В качестве домашнего задания можно дать работу учащимся по подготовке программ вычисления определенного интеграла методом Монте-Карло.

2.2. СЕМИНАР НА ТЕМУ «УРАВНЕНИЕ»

Обоснование выбора темы. Школьный курс алгебры содержит разнообразный материал. Однако центральным пунктом является вопрос об уравнениях. Учащиеся приобретают навыки решений, знакомятся с алгоритмами решений линейных и квадратных уравнений. В курсе ОИВТ знакомятся с алгоритмами приближенного решения уравнений. Обобщение и систематизация знаний об уравнениях, использование компьютеров при их решении усилят прикладную и практическую направленность в преподавании математики.

Данное семинарское занятие предполагается провести в 11 классе после изучения темы «Иррациональные уравнения».

Цель занятия: повторение понятий уравнение, решение уравнения, корень уравнения, способы решения уравнений, алгоритм решения уравнений, приближенное решение уравнений; реализация алгоритмов линейного и квадратного уравнений; реализация приближенных решений уравнений на компьютере.

Приборы и принадлежности: таблицы «Линейное уравнение», «Квадратное уравнение», «Графическое решение уравнений», «Алгоритм приближенного решения уравнений»; компьютер.

Учащиеся делятся на четыре группы. Первая группа готовит теорию уравнений. Вторая – алгоритмы решения линейных и квадратных уравнений. Предлагает программы реализации этих алгоритмов на языке Бейсик или Паскаль. Третья – теорию алгебраических уравнений и метод подбора в решении алгебраических уравнений. Четвертая – алгоритм и программу приближенного решения уравнений. Из каждой группы выделяется докладчик. Остальные помогают своему докладчику, готовят наглядные пособия, таблицы.

Примерное содержание и ход занятия.

Определение. Равенство, содержащее переменную, называется уравнением, если необходимо найти значение переменной, при котором оно является верным.

Уравнение с одной переменной в общем виде записывается так: $f_1(x)=f_2(x)$. Значение переменной, при котором уравнение обращается в верное равенство, называется корнем уравнения. Если a – корень уравнения $f_1(x)=f_2(x)$, то равенство $f_1(a)=f_2(a)$ является верным.

Множество значений переменной x , при которых имеет смысл выражение $f_1(x)$ и $f_2(x)$, называется областью определения уравнения.

Решить уравнение – значит найти все множество его корней. Уравнение может и не иметь корней.

Определение. Два уравнения называются равносильными в данном числовом множестве, если всякий корень первого уравнения является корнем второго и, наоборот, всякий корень второго является корнем первого.

Теорема 1. Если к обеим частям уравнения $f_1(x)=f_2(x)$ прибавить одно и то же выражение $f_3(x)$, имеющее смысл при всех допустимых значениях переменной x , то новое уравнение равносильно данному.

Следствие. Равенство не нарушится, если к обеим частям прибавить (отнять) одно и то же число.

Следствие. Любое слагаемое можно переносить из одной части уравнения в другую, поменяв знак слагаемого на противоположный.

Теорема 2. Если обе части уравнения $f_1(x)=f_2(x)$ умножить или разделить на одно и то же число m , не равное нулю, то новое уравнение $m \cdot f_1(x)=m \cdot f_2(x)$ равносильно данному.

Следствие. Если знаки всех членов изменить на противоположные, то полученное уравнение равносильно данному.

Следствие. Если обе части уравнения привести к общему знаменателю, не содержащему переменную, а затем обе части умножить на этот знаменатель, то полученное уравнение равносильно данному.

Определение. Если $f_1(x)$ и $f_2(x)$ являются многочленами, то уравнение $f_1(x)=f_2(x)$ называется алгебраическим.

Вид алгебраического уравнения n степени:

$$a_n x^n + a_{n-1} x^{n-1} + \dots + a_1 x + a_0 = 0.$$

Алгебраическими являются линейные и квадратные уравнения. Для этих уравнений существует алгоритм решения, т.е. перечень указаний, следуя которым можно найти решение.

На доске вывешиваются таблицы: «Линейное уравнение», «Квадратное уравнение». Докладчик комментирует таблицы.

Если есть алгоритм решения уравнения, то можно написать программу решения на компьютере. На доске записывает программы решения линейного и квадратного уравнений.

Программа 1.1	Программа 1.2
<pre> 10 REM РЕШЕНИЕ ЛИНЕЙНОГО УРАВНЕНИЯ 20 REM A1*X + B1 = A2*X + B2 30 INPUT A1,B1,A2,B2 40 PRINT "КОРЕНЬ УРАВНЕНИЯ X=";(B2-B1)/(A1-A2) 50 STOP </pre>	<pre> Program P1_2 Uses crt; Var a1,b1,a2,b2,x:real; begin Clrscr; {очистка экрана} writeln('vvedi a1,b1,a2,b2 '); readln(a1,b1,a2,b2); writeln('Корень x=',(b2-b1)/(a1- a2):6:3); readkey; end. </pre>

Программа 2.1	Программа 2.1
<pre> 10 REM РЕШЕНИЕ КВАДРАТ- НОГО УРАВНЕНИЯ 20 REM A*X^2+B*X+C=0 30 INPUT A,B,C 40 D=B^2-4*A*C 50 IF D<0 THEN PRINT "КОРНЕЙ НЕТ":GOTO 90 60 IF D=0 THEN PRINT "ОДИН КОРЕНЬ X=";-B/2*A:GOTO 90 70 PRINT "КОРНИ:X1=";(-B- SQR(D))/2*A 80 PRINT " X2=";(-B+SQR(D))/2*A 90 STOP </pre>	<pre> Program P2_1 Uses crt; var a,b,c,x1,x2,d:real; begin Clrscr; {очистка экрана} read(a,b,c); d:=sqr(b)-4*a*c; if d<0 then writeln('Корней нет'); if d=0 then writeln('Один ко- рень x=',-b/(2*a)); if d>0 then begin writeln('Корни'); writeln('x1=',(-b- sqr(d))/(2*a):6:3); writeln('x2=',(-b+ sqr(d))/(2*a):6:3); end; readkey; end. </pre>

Теорема Безу. Многочлен $P(x)$ делится без остатка на двучлен $x-a$ в том и только в том случае, когда a – корень многочлена.

Уравнение вида $x^n+a_1x^{n-1}+...+a_n=0$, $a_n \in Z$, где старший коэффициент равен единице, называется приведенным.

Все целые корни приведенного алгебраического уравнения с целыми коэффициентами являются делителями его свободного члена. Приведенное уравнение с целыми коэффициентами не имеет других корней, кроме целых. Поэтому в качестве целых корней надлежит испытывать не какие-либо произвольные числа, а лишь делители свободного члена. Этот метод подбора дает все рациональные решения уравнения.

Пример. Решить уравнение $x^3 - 4x^2 - 27x + 90 = 0$.

Решение. Выписываем делители свободного члена: $\pm 1, \pm 2, \pm 3, \pm 5, \pm 6, \dots$. Подставляя эти числа в уравнение, находим, что корнем данного уравнения является число $x=3$, поскольку $3^3 + 4 \cdot 3^2 - 27 \cdot 3 + 90 = 0$.

Многочлен, расположенный в левой части уравнения, по теореме Безу должен без остатка делиться на двучлен $x-a$. Проведя деление, найдем в частном квадратный трехчлен $x^2 - x - 30$. Его два корня присоединим к ранее найденному корню, и тогда корни уравнения: $3, -5, 6$.

Если уравнение $a_n x^n + a_{n-1} x^{n-1} + \dots + a_0 = 0$ имеет целые коэффициенты и имеет рациональный корень $x = p/q$, то число p является делителем a_0 , а число q является делителем a_n .

Пример. Решить уравнение $2x^3 - x^2 - 6x + 3 = 0$.

Решение. Если данное уравнение имеет корень p/q , то число p должно быть делителем числа 3, а число q должно быть делителем числа 2.

Тогда, если уравнение имеет рациональные корни, то это могут быть числа: $\pm 1, \pm 1/2, \pm 3, \pm 3/2$. Подставляя, убеждаемся, что уравнение имеет корень $x = 1/2$. Преобразуем левую часть уравнения: $2x^3 - x^2 - 6x + 3 = (x - 1/2)(2x^2 - 6)$. Следовательно, корни уравнения: $x_1 = 1/2$, $x_2 = \sqrt{3}$, $x_3 = -\sqrt{3}$.

А если не знаем решение уравнения или уравнение не имеет аналитического решения? Тогда можно воспользоваться графическим способом решения уравнений. Этот способ дает приближенное решение уравнений.

На доске вывешивается таблица «Графическое решение уравнения». Докладчик комментирует таблицу.

Для приближенного решения уравнения $f(x) = 0$ воспользуемся методом уточнения корня (методом половинного деления).

На доске вывешивается таблица «Алгоритм приближенного решения уравнений». Докладчик делает соответствующие поясне-

ния. Приводит программу приближенного решения уравнений. Замечает, что в строчку 30 нужно вписать за знаком равенство, справа, решаемое уравнение.

Программа 3.1	Программа 3.2
<pre> 10 REM УТОЧНЕНИЕ КОРНЯ 11 REM E – ТОЧНОСТЬ ПРИБЛИ- ЖЕНИЯ 12 REM A,B – ОБЛАСТЬ ОПРЕДЕ- ЛЕНИЯ 20 INPUT "A,B,E";A,B,E 30 DEF FN I(X)= 40 IF B-A<=2*E THEN 80 50 C=(A+B)/2 60 IF FN I(A)*FN I(C)<=0 THEN B=C: GOTO 40 70 A=C:GOTO 40 80 PRINT "X=";(A+B)/2: END </pre>	<pre> Program P3_2 {Уточнение корня} Uses crt; var a,b,e,c:real; function f(x:real):real; begin Clrscr; {очистка экрана} f:=x*x-2; end; begin writeln('Введи a='); readln(a); writeln('Введи b='); readln(b); writeln('Введи точность e'); readln(e); while b-a>2*e do begin c:=(a+b)/2; if f(a)*f(c)<=0 then b:=c else a:=c; end; c:=(a+b)/2; writeln('koren=',c:6:3); readkey; end. </pre>

По окончании докладов выслушиваются отзывы, оценки, дополнения. В оставшееся время учащиеся приступают к компьютерному решению уравнений. Задания учитель готовит заранее.

2.3. СОСТАВЛЕНИЕ И РЕШЕНИЕ ГРАФИЧЕСКИХ И АРИФМЕТИЧЕСКИХ РЕБУСОВ НА УРОКЕ ИНФОРМАТИКИ

Ребус – это загадка, в которой разгадываемые слова даны в виде рисунков в сочетании с буквами и некоторыми другими знаками.

При разгадывании ребусов надо помнить о некоторых условиях. Если запятая (или две) стоит перед знаком или нарисованным предметом, то отбрасываем одну или две первые цифры соответственно знака или нарисованного предмета. Если запятые стоят после знака или нарисованного предмета, то в соответствующем слове отбросить последние буквы, одну или две. В отдельных случаях указано, какую букву надо отбросить или заменить другой буквой (рис.10).

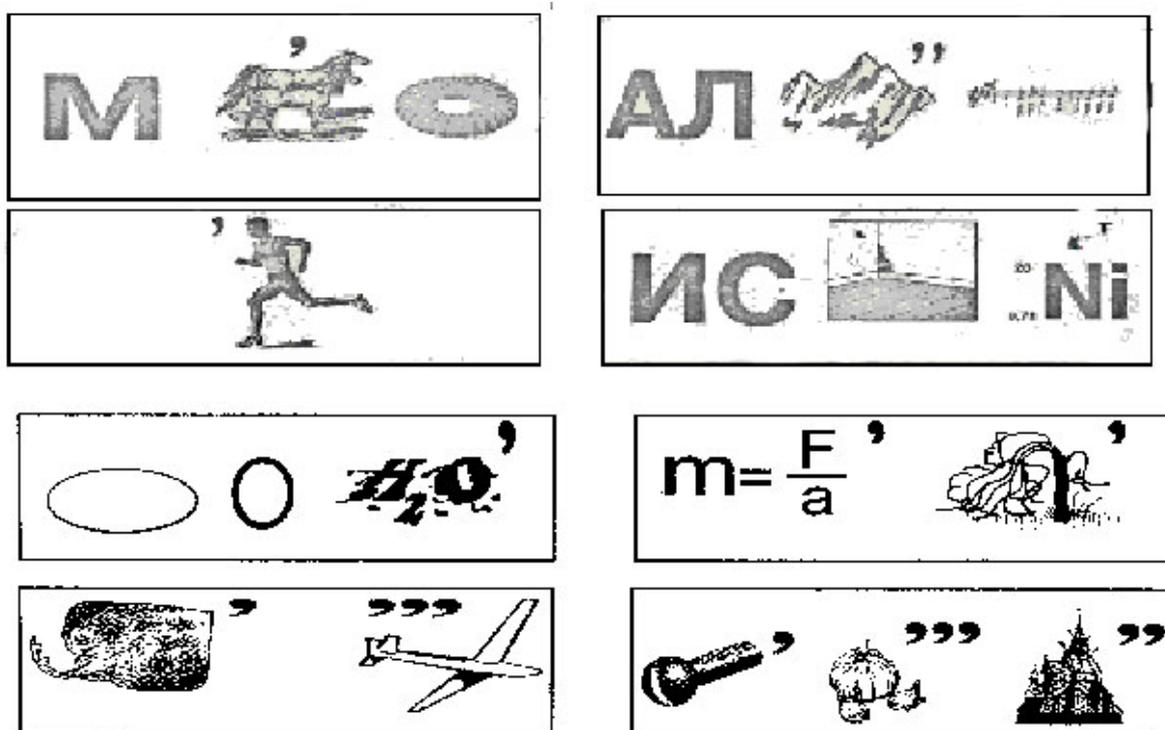


Рис. 10

Арифметические ребусы. Кроме графических ребусов, для развития логического мышления учащимся можно предложить арифметические ребусы. В них каждая буква соответствует какой-либо цифре. Предлагается компьютерное решение математических ребусов. Ниже приведена программа решения арифметического ребуса (комментарии ориентированы на язык Бейсик).

Задача 1. Решить арифметический ребус, т.е. найти, какую цифру обозначает каждая буква в следующем равенстве:

$$AX^A = BAX.$$

Решение. Для начала вспомним, что при записи натуральных чисел в десятичной системе счисления цифры, участвующие в записи числа, при чтении их справа налево указывают последовательно, сколько в данном числе содержится единиц, затем десятков, сотен, тысяч и т. д.

Например: $18 = 1 \cdot 10 + 8,$
 $348 = 3 \cdot 100 + 4 \cdot 10 + 8,$
 $4762 = 4 \cdot 1000 + 7 \cdot 100 + 6 \cdot 10 + 2.$

Для решения непосредственно нашей задачи нужно составить числа AX и BAX . Обычно в математике переменные обозначают одной буквой x, y, a, b и т.д., но в Бейсике переменная может включать до 8 латинских букв (или символов, но первой должна быть латинская буква).

AX обозначим как $AX,$
 BAX как $BAX.$

Составим числа $AX = A \cdot 10 + X$
 $BAX = B \cdot 100 + A \cdot 10 + X.$

Теперь нам остается организовать перебор по B, A, X (три вложенных цикла) и проверить условие $AX^A = BAX,$ в случае выполнения данного условия вывести на печать $BAX.$ Решение данного арифметического ребуса приведено в программе 1. Строчки 70, 80, 90 исключают совпадение $A, B, X.$ Ответ: 625.

Программа 1.1	Программа 1.2
<pre> 10 FOR B=1 TO 9 20 FOR A=1 TO 9 30 FOR X=0 TO 9 40 AX= 10*A + X 50 BAX= 100*B + 10*A + X 60 IF AX^A=BAX THEN PRINT BAX 70 IF X=B THEN 100 80 IF X=A THEN 100 </pre>	<pre> Program P1_2 uses crt; label 100; var b,a,x:byte; ax,bax,k,i:integer; begin Clrscr; {очистка экрана} for b:=1 to 9 do for a:=1 to 9 do </pre>

<pre> 90 IF A=B THEN 110 100 NEXT X 110 NEXT A 120 NEXT B 130 END </pre>	<pre> for x:=0 to 9 do begin ax:=10*a+x; bax:=100*b+10*a+x;k:=1; for i:=1 to a do k:=k*ax; if k=bax then writeln (bax:10:5); if x=b then goto 100; if x=a then goto 100; if a=b then goto 100; 100: end; readkey; end. </pre>
--	---

Задача 2. В десятичной записи числа $42*4*$ две цифры пропущены. Восстановите их, если известно, что число кратно 72.

Решение.

Программа 2.1	Программа 2.2
<pre> 10 FOR C3=0 TO 9 ' ПЕРЕБИРАЕМ ВОЗМОЖНЫЕ ЗНАЧЕНИЯ ТРЕТЬЕЙ ЦИФРЫ 20 FOR C5=0 TO 9 ' ПЕРЕБИРАЕМ ВОЗМОЖНЫЕ ЗНАЧЕНИЯ ПЯТОЙ ЦИФРЫ 30 CH=42000 + C3*100 + 40 + C5 40 IF CH/72=CH\72 THEN PRINT CH 50 NEXT C5 60 NEXT C3 70 END </pre>	<pre> Program P2_2; Uses crt; var c3,c5,ch:longint; begin Clrscr; {очистка экрана} for c3:=0 to 9 do {перебираем значения третьей цифры } for c5:=0 to 9 do{перебираем значения пятой цифры} begin ch:=42000+c3*100+40+c5; if ch/72=ch div 72 then writeln(ch); end; readkey; end. </pre>

Задача 3. В написанном выражении $(((((1?2)?3)?4)?5)?6$ вместо каждого вопросительного знака вставить знак одного из четырех арифметических действий: +, -, *, / так, чтобы результат вычисления равнялся 35 (при делении дробная часть в частном отбрасывается). Достаточно найти одно решение.

Решение

Программа 3.1	Программа 3.2
<pre> 10 'АРИФМЕТИЧЕСКИЕ ДЕЙСТ- ВИЯ 20 M=35: N=6 30 DIM A(N), B(N) 40 B(1)=1: K=0: A(2)=0 50 FOR I=3 TO N: A(I)=4: NEXT 60 FOR I=N TO 2 STEP -1 70 IF A(I)=4 THEN A(I)=1: GOTO 300 80 A(I)=A(I)+1:Y=B(I-1) 90 ON A(I) GOTO 100,110,120,130 100 Z=Y+I:GOTO 140 110 Z=Y-I:GOTO 140 120 Z=Y*I:GOTO 140 130 Z=Y\I 140 B(I)=Z:IF I=N GOTO 160 150 FOR J=I+1 TO N:B(J)=B(J- 1)+J:NEXT 160 IF B(N)<>M GOTO 60 170 K=K+1 180 FOR J=2 TO N-1:PRINT «(«;:NEXT 190 PRINT 1; 200 FOR J=2 TO N 210 IF J>2 THEN PRINT «)»; 220 ON A(J) GOTO 230,240,250,260 230 PRINT «+»;:GOTO 270 240 PRINT «-»;:GOTO 270 250 PRINT «*»;:GOTO 270 260 PRINT «\»; 270 PRINT J; 280 NEXT J:PRINT»=«;M 290 GOTO 60 300 NEXT I:PRINT K 310 END </pre>	<pre> Program P3_2; Uses crt; const M=35; N=9; label R; var i,j,k,y:integer; A,B :array[1..N] of integer; BEGIN Clrscr; {очистка экрана} B[1]:=1; k:=0; A[2]:=0; for i:=N downto 2 do if (A[i]=4) then A[i]:=4; R: for i:=N downto 2 do if (A[i]=4) then A[i]:=1 else begin A[i]:=A[i]+1; y:=B[i-1]; case (A[i]) of 1: B[i]:=y+i; 2: B[i]:=y-i; 3: B[i]:=y*i; 4: B[i]:=y div i end; for j:=i+1 to N do B[j]:=B[j-1]+j; if (B[N]<>M) then goto R; k:=k+1; for j:=2 to N-1 do write('('); write ('1'); for j:=2 to N do begin if (j>2) then write ('); case A[j] of 1:write('+'); 2:write ('-'); 3:write('*'); 4:write('/'); end; write(j); end; writeln('= ',M);goto R; end; </pre>

writeln(k); readkey; end.

Задачи для самостоятельного решения:

1. Решите арифметические ребусы:

$$\begin{array}{r} \text{а) } \begin{array}{r} \text{А} \\ + \text{АБ} \\ \hline \text{АБВ} \\ \text{----} \end{array} \qquad \begin{array}{r} \text{б) } \text{АБВ} \\ - \text{ВБА} \\ \hline \text{---} \end{array} \\ \text{ГБА:А=АА} \\ \text{БВБ} \end{array}$$

в) $\overline{\text{ЛОБ}} + \overline{\text{ТРИ}} = \overline{\text{САМ}}$

г) $(\overline{\text{Ж-1}})^5 = \overline{\text{ЖЖЖ(Ж-1)}}$

д) $\overline{\text{ИКС}}^2 = \overline{\text{***ИКС}}$

е) $\overline{\text{АВВА}} = \overline{\text{АА}}^2 + \overline{\text{ВВ}}^2$

ж) $\overline{\text{КИО}} * \overline{\text{ИО}} = \overline{\text{ТОКИО}}$

2. В трехзначном числе зачеркнули первую цифру слева; когда полученное число умножили на 7, получилось исходное число. Найдите его.

3. Припишите к 523*** три такие цифры справа, чтобы полученное шестизначное число делилось на 7, 8 и 9 без остатка.

4. Припишите к ***999 три такие цифры слева, чтобы полученное число делилось на 13, 17 и 19 без остатка.

2.4. ПРАКТИЧЕСКАЯ РАБОТА ПО ТЕМЕ «СИМВОЛЬНЫЕ ВЕЛИЧИНЫ»

Одним из основных направлений использования ЭВМ является накопление и обработка таблиц, справочников, словарей, где кроме числовых данных используется и текст. В языках программирования, кроме числовых значений и переменных, используются символьные (литерные) величины и переменные.

Определимся в терминологии. Значением символьной величины является один символ: русская или латинская большая или маленькая буква, цифра, знак препинания или специальный знак. Пробел также является символом. Всего символьная величина может принимать 256 различных значений, т.е. существует 256 различных символов. Их принято нумеровать от 0 до 255. Номер сим-

вола называется его кодом. Символьные значения в Бейсике заключаются в кавычки, например: «А», «\$», «4», в Паскале описываются через тип переменной string. Для обозначения символьных переменных используются до 8 символов, первый символ обязательно должен быть латинской буквой, другие — латинскими буквами или цифрами.

Для отличия символьных переменных от других типов в конце имени приписывается специальный символ \$. Пример присваивания символьной переменной символьных величин в языке Бейсик: ABC\$=«К», XY\$= «+», X\$=«4», в Паскале ABC='К', XY='+', X='4'.

Линейные таблицы, элементами которых являются символы, называются литерными величинами. Значением литерной величины является строка символов. Число символов в строке называется ее длиной. Имена литерных переменных обозначаются так же, как и у символьных переменных.

В дальнейшем символьные и литерные переменные будем считать равносильными.

Пример присваивания символьной переменной A\$ величины ИНФОРМАТИКА: A\$= «ИНФОРМАТИКА».

Основные операции над символьными величинами: соединение, вычисление длины, вырезки символьной величины, поиск подстроки.

Ниже даны примеры работы с символьными величинами и их решения на языках Бейсик и Паскаль.

Пример 1. Склеивание или соединение двух литерных величин «МА» и «ША». Результатом является литерная величина «МАША».

Программа 1.1	Программа 1.2
<pre>10 A\$="МА" 20 B\$="ША" 30 C\$=A\$+B\$ 40 PRINT C\$</pre>	<pre>Program P1_2; uses crt; var a,b,c:string; begin clrscr; {очистка экрана} a:='ма'; b:='ша'; c:=a+b; write(c); end.</pre>

Пример 2. Вычисление длины литерной величины «ИНФОРМАТИКА». Результатом является целое число, равное 11.

Программа 2.1	Программа 2.2
<pre>10 A\$="ИНФОРМАТИКА" 20 X=LEN (A\$) 30 PRINT X</pre>	<pre>Program P2_2; uses crt; var a:string; x:word; begin clrscr; {очистка экрана} a:='информатика'; x:=length(a); write(x:6:3) readkey; end.</pre>

Пример 3. Функция LEFT\$ (X\$,Y) позволяет вывести на экран строку длиной Y, начиная с крайнего левого символа литерной переменной X\$.

Программа 3.1	
<pre>10 C\$="ПАРОХОД" 20 A\$=LEFT\$ (C\$,3) 30 PRINT X</pre>	

Результатом исполнения этой программы является литерная величина A\$, значение которой «ПАР».

Пример 4. Функция RIGHT (Y\$,X) позволяет вывести на экран строку длиной X, начиная с крайнего правого символа литерной переменной Y\$.

Программа 4.1	
<pre>10 C\$="ПАРОХОД" 20 A\$=RIGHT\$ (C\$,3) 30 PRINT A\$</pre>	

Результатом исполнения этой программы является литерная величина A\$, значение которой «ХОД».

Пример 5. Функция MID\$ (C\$,X,Y) в Бейсике позволяет сделать вырезку символов длиной Y, начиная с X позиции литерной переменной C\$. На языке Паскаль вырезка делается с помощью функции COPY.

Программа 5.1	Программа 5.2
<pre>10 A\$="ИНФОРМАТИКА" 20 B\$=MID\$(A\$,3,5) 30 PRINT B\$</pre>	<pre>Program P5_2; uses crt; var a,b:string; begin clrscr; {очистка экрана} a:='информатика'; b:=copy(a,3,5); write(b); readkey; end.</pre>

Результатом исполнения этой программы является литерная величина, значение которой «ФОРМА».

Пример 6. Функция CHR\$(X) позволяет вывести на экран символ, код которого равен X. X принимает значения от 0 до 255.

```
10 INPUT X
20 PRINT "Y$="; CHR$(X)
```

Пример 7. Функция ASC(X\$) позволяет вывести на экран код первого символа литерной величины X\$. Результат функции – числовое значение от 0 до 255.

```
10 X$= «TEST»
20 Y=ASC(X$)
30 PRINT Y
```

Результатом является число 84.

Пример 8. Функция str\$(x) преобразует число x в литерную величину.

Программа 8.1	Программа 8.2
<pre>10 INPUT "ВЕДИТЕ ЧИСЛО";X 20 A\$=STR\$(X) 30 PRINT"СТРОКА СИМВОЛОВ"; A\$</pre>	<pre>Program P8_2 uses crt; var a:string; x:integer; begin clrscr; {очистка экрана} writeln('vvedi chiclo'); read(x);</pre>

	<pre>str(x,a); writeln('stroka simvolov=',a); readkey; end.</pre>
--	---

Пример 9. Функция val(x\$) в Бейсике преобразует литерное выражение как цепочку цифр в число.

Программа 9.1	Программа 9.2
<pre>10 INPUT"ВВЕДИТЕ СТРОКУ СИМВОЛОВ";X\$ 20 Y=VAL(X\$) 30 PRINT"ЧИСЛО";Y</pre>	<pre>Program P9_2 uses crt; var x:string; b:word; y:real; begin clrscr; {очистка экрана} writeln('vvedi stroku simvolov'); Read(x); val(x,y,b); write('chiclo=',y) readkey; end.</pre>

Задачи для классного и самостоятельного решения

Задача 1. Написать программу подсчета числа вхождений буквы «А» в данный текст.

Программа 1.1	Программа 1.2.
<pre>10 INPUT A\$ 20 X=LEN(A\$) 25 K=0 30 FOR I=1 TO X 40 IF MID\$ (A\$,I,1)="A" THEN K=K+1 50 NEXT I 60 PRINT K</pre>	<pre>Program P1_2; Uses crt; var i,k:integer; y:string; begin clrscr; {очистка экрана} writeln('vvedi stroky');readln(y); k:=0; for i:=1 to length(y) do if y[i]='a' then k:=k+1; write(k); readkey; end.</pre>

Задача 2. Составить программу замены в слове всех букв «А» на букву «В».

Программа 2.1	Программа 2.2
<pre> 10 INPUT A\$ 20 X=LEN (A\$) 25 C\$=" _" 30 FOR I=1 TO X 40 B\$=MID\$(A\$,I,1) 50 IF B\$="A" THEN B\$="B" 60 C\$=C\$+B\$ 70 NEXT I 80 PRINT C\$ </pre>	<pre> Program P2_2; uses crt; var i:integer; y,c:string; begin clrscr; {очистка экрана} writeln('vvedi stroky');readln(y); for i:=1 to length(y) do begin if y[i]='a' then y[i]:='b'; c:=c+y[i]; end; write(c); readkey; end. </pre>

Задача 3. Составить алгоритм замены в слове всех букв “А” на “В” и наоборот.

Программа 3.1	Программа 3.2
<pre> 10 INPUT A\$ 20 X=LEN (A\$) 25 C\$=" _" 30 FOR I=1 TO X 40 B\$=MID\$(A\$,I,1) 50 IF B\$="A" THEN B\$="B": GOTO 70 60 IF B\$="B" THEN B\$="A" 70 C\$=C\$+B\$ 80 NEXT I 90 PRINT C\$ </pre>	<pre> Program P3_2; uses crt; label aa; var i:integer; y,c:string; begin clrscr; {очистка экрана} writeln('vvedi stroky');readln(y); for i:=1 to length(y) do begin if y[i]='a' then y[i]:='b' else if y[i]='b' then y[i]:='a'; c:=c+y[i]; end; write(c); readkey; end. </pre>

Задача 4. Написать программу подсчета числа вхождений слова X в слово Y (например: КОЛОКОЛ, КОЛ).

Программа 4.1	Программа 4.2
<pre> 10 INPUT X\$ 20 INPUT Y\$ 30 X=LEN (x\$) 40 Y=LEN (Y\$) 45 K=0 50 FOR I=1 TO Y-x 60 MID\$(Y\$,I,X)=X\$ THEN K=K+X 70 NEXT I 80 PRINT K </pre>	<pre> Program P4_2; uses crt; var i,xd,yd,k:integer; x,y:string; begin clrscr; {очистка экрана} writeln('vvedi slovo 1'); readln(x); writeln('vvedi slovo 2'); readln(y); xd:=length(x); yd:=length(y); k:=0; for i:=1 to xd do if copy(x,i,yd)=y then k:=k+1; write('k=',k); readkey; end. </pre>

Задача 5 Написать программу обращения слова.

Программа 5.1	Программа 5.2
<pre> 10 INPUT A\$ 20 L=LEN A\$ 30 C\$="" 40 FOR I=L TO 1 STEP-1 50 B\$=MID\$(A\$,I,1) 60 C\$=C\$+B\$ 70 NEXT I 80 PRINT C\$ </pre>	<pre> Program P5_2; uses crt; var i:integer; y,c:string; begin clrscr; {очистка экрана} writeln('vvedi slovo');readln(y); c:=""; for i:=length(y) downto 1 do c:=c+y[i]; write(c); readkey; end. </pre>

Задача 6. Составить алгоритм проверки, является ли данное слово перевертышем .

Программа 6.1	Программа 6.2
<pre> 10 INPUT A\$ 20 X=LEN(A\$) 30 Y=INT(X/2) 40 M\$=LEFT\$(A\$,Y) 50 N\$=RIGHT\$(A\$,Y) 60 IF M\$=N\$ THEN PRINT "Слово перевертыш"</pre>	<pre> Program P6_2; uses crt; var i:integer; y,c:string; begin clrscr; {очистка экрана} writeln('введи слово');readln(y); c:=""; for i:=length(y) downto 1 do c:=c+y[i]; if c=y then write('perivertch'); readkey; end.</pre>

Задача 7. Предложение состоит из слов, между словами – пробел, а после последнего слова – точка. Определить число слов в предложении.

Программа 7.1	Программа 7.2
<pre> 10 INPUT A\$ 20 X=LEN (A\$) 25 K=0 30 IF MID\$ (A\$,I,1)=" " THEN K=K+1 50 NEXT I60 PRINT "КОЛИЧЕСТВО СЛОВ";K+1</pre>	<pre> Program P7_2; uses crt; var i,k:integer; y:string; begin clrscr; {очистка экрана} writeln('введи текст');readln(y); k:=0; for i:=1 to length(y) do if y[i]=' ' then k:=k+1; write(k); readkey; end.</pre>

Задачи для самостоятельного решения

1). Напишите программу, составляющую с помощью вырезки и склеивания из слова «ШЛЯПА» все возможные слова (сущест-

вующие в ед. числе). Каждую букву можно использовать многократно. Сколько получится таких слов. Результат: 6

2). Даны два слова. Сколько раз во втором слове встречается первая буква первого слова?

Контрольный пример.

Данные: «БУКВА», «БАБОЧКА»

Результат: 2

Данные: «БУКВА», «ЗВУК»

Результат: 0

3). Напишите программу, составляющую с помощью вырезки и склеивания из слова «БАЛКОН» слова «ЛОБ», «КЛАН», «КОЛБА».

4). Даны два слова одинаковой длины. Присвоить переменной k число, равное количеству попарно одинаковых букв.

Контрольный пример:

Данные: «ЧЕРЕПАШКА», «БЕГЕМОШКА».

Результат: 5

2.5. УРОК-АНАЛИЗ. РАЗБОР РЕШЕНИЙ КЛЮЧЕВЫХ ЗАДАЧ ПО ТЕМЕ «МАССИВЫ»

Ниже приведены ключевые задачи с решениями для коллективного разбора. В примечаниях выделены вопросы и задания, на которые следует ответить или выполнить. Это может сделать ученик или учитель. Предлагаемые задачи с решениями раздают ученикам в распечатанном виде.

Задача 1. Написать программу ввода одномерного массива с клавиатуры.

Программа 1.1	Программа 1.2
<pre>10 DIMA(6) 20 FOR I=1 TO 6 30 INPUT A(I) 40 NEXT I 50 FOR I=1 TO 6 60 PRINT"A("I")=";A(I) 70 NEXT I</pre>	<pre>Program P1_2; uses crt; type massiv=array[1..6] of integer; var a:massiv; i:word; begin clrscr; {очистка экрана} for i:=1 to 6 do readln(a[i]);</pre>

	<pre>for i:=1 to 6 do writeln('a[' ,i ,']=',a[i]); readkey; end.</pre>
--	--

Примечание. Пояснить организацию ввода элементов массива с клавиатуры. Пояснить организацию вывода.

Задача 2. Написать программу ввода двумерного массива с клавиатуры.

Программа 2. 1	Программа 2.2
<pre>10 DIMX(3,4) 20 FOR I=1 TO 3 30 FOR J=1 TO 4 40 INPUT X(I,J) 50 NEXT J 60 NEXT I 70 FOR I=1 TO 3 80 FOR J=1 TO 4 90 PRINT "X("I","J")="; X(I,J) 100 NEXT J 110 NEXT I</pre>	<pre>Program P2_2; uses crt; var x:array[1..3,1..4] of integer; i,j:word; begin clrscr; {очистка экрана} for i:=1 to 3 do for j:=1 to 4 do readln(x[i,j]); for i:=1 to 3 do for j:=1 to 4 do writeln('x[' ,i ,',' ,j ,']=',x[i,j]); readkey; end.</pre>

Примечание. Обратить внимание на особенности ввода, вывода двумерного массива. Особенности вывода на экран.

Задача 3. Написать программу ввода символьного массива с клавиатуры.

Программа 3.1	Программа 3.2
<pre>10 DIMB\$(4) 20 FOR I=1 TO 4 30 INPUT B\$(I) 40 NEXT I 50 FOR I=1 TO 4 60 PRINT B\$(I) 70 NEXT I</pre>	<pre>Program P3_2; uses crt; var b:array[1..4] of char; i:word; begin clrscr; {очистка экрана} for i:=1 to 4 do readln(a[i]); for i:=1 to 4 do writeln('b[' ,i ,']=',b[i]); readkey; end.</pre>

Примечание. Разъяснить отличие ввода, вывода символьного массива от числового.

Задача 4. Написать программу ввода элементов массива с использованием генератора случайных чисел.

Программа 4.1	Программа 4.2
<pre> 10 DIMA(2,3) 20 FOR I=1 TO 2 30 FOR J=1 TO 3 40 A(I,J)=INT(RND(1) *10) 50 PRINT A(I,J) 60 NEXT J 70 NEXT I </pre>	<pre> Program P4_2; uses crt; var x:array[1..3,1..4] of integer; i,j:word; begin clrscr; {очистка экрана} for i:=1 to 3 do for j:=1 to 4 do x[i,j]:=trunc(random(100))-50; for i:=1 to 3 do for j:=1 to 4 do writeln('x[' ,i ,',',j,']=',x[i,j]); readkey; end. </pre>

Примечание: Пояснить, как генерируются случайные числа. Обратить внимание на то, что ввод и вывод организованы в одном цикле. Объяснить, как изменится программа, если изменить интервал генерации случайных чисел.

Задача 5. Написать программу ввода значений массива, записанных в тексте программы с помощью операторов DATA, READ.

Программа 5.1	Программа 5.2
<pre> 20 DIMA\$(4) 30 FOR I=1 TO 4 40 READ A\$(I) 50 NEXT I 60 DATA "ЛЕ- НА", "ОЛЯ", "КАТЯ", "КОЛЯ" 80 FOR I=1 TO 4 90 PRINT A\$(I) 100 NEXT I </pre>	<pre> Program P5_2; uses crt; var a:array[1..4] of string; i:intrger; begin clrscr; {очистка экрана} a[1]:='koli'; a[2]:='oli'; a[3]:='kati'; a[4]:='lena'; for i:=1 to 4 do writeln('a[' ,i,']=',a[i]); readkey; end. </pre>

Примечание. Разъяснить включение в программу значений элементов массива.

Задача 6. Написать программу расчета элементов массива по формуле.

Программа 6.1	Программа 6.2
<pre> 20 DIM T%(9,9) 30 FOR I=1 TO 9 40 FOR J=1 TO 9 50 T%(I,J)=I*J: REM ФОРМУЛА 60 PRINT T%(I,J); 70 NEXT J 100 NEXT I </pre>	<pre> Program P6_2; uses crt; var x:array[1..9,1..9] of integer; i,j:word; begin clrscr; {очистка экрана} for i:=1 to 9 do for j:=1 to 9 do begin x[i,j]:=i*j;{formula} writeln('x[' ,i ,',' ,j ,']=' ,x[i,j]); end; readkey; end. </pre>

Примечание. Показать особенность ввода элементов массива по формуле.

Задача 7. Вычислить сумму элементов двумерного массива, элементы которого задаются генератором случайных чисел.

Программа 7.1	Программа 7.2
<pre> 20 DIM A(2,3) 30 FOR I=1 TO 2 40 FOR J=1 TO 3 50 A(I,J)=INT(RND(1) *10) 60 PRINT A(I,J) 70 S=S+A(I,J) 80 NEXT J 90 NEXT I 100 PRINT "S=";S </pre>	<pre> Program P7_2; Uses crt; var x:array[1..3,1..4] of integer; i,j:word; s:integer; begin clrscr; {очистка экрана} s:=0; for i:=1 to 3 do for j:=1 to 4 do begin x[i,j]:=trunc(random(10)); writeln('x[' ,i ,',' ,j ,']=' ,x[i,j]); s:=s+x[i,j]; end; end; </pre>

	writeln('s=',s); end.
--	--------------------------

Примечание. Обратить внимание на организацию копилки суммы элементов массива.

Задача 8. Вычислить произведение элементов двумерного массива, элементы которого задаются генератором случайных чисел.

Программа 8.1	Программа 8.2
<pre> 20 DIMA(2,3):P=1 30 FOR I=1 TO 2 40 FOR J=1 TO 3 50 A(I,J)=INT(RND(1)*10) 60 PRINT A(I,J) 70 P=P*A(I,J) 80 NEXT J 90 NEXT I 100 PRINT "P=";P </pre>	<pre> Program P8_2; Uses crt; var x:array[1..3,1..4] of integer; i,j,:word; p:integer; begin s:=1; clrscr; {очистка экрана} for i:=1 to 3 do for j:=1 to 4 do begin x[i,j]:=trunc(random(10)); writeln('x[';i;',';j;']=';x[i,j]); p:=p*x[i,j]; end; writeln('p=',p); readkey; end. </pre>

Примечание. Обратить внимание на организацию копилки произведения элементов массива.

Задача 9. Подсчет количества элементов, превосходящих число p .
($-7 \leq p \leq 7$)

Программа 9.1	Программа 9.2
<pre> 20 DIMA(2,3) 30 INPUT "ВВЕДИ P";P 40 FOR I=1 TO 2 50 FOR J=1 TO 3 60 F(I,J)=INT(RND(1)*15-8) 70 PRINT A(I,J); 80 IF A(I,J)>P THEN K=K+1 90 NEXT J 100 NEXT I </pre>	<pre> Program P9_2; uses crt; var a:array[1..2,1..3] of integer; i,j,p,k:integer; begin k:=0; clrscr; {очистка экрана} writeln('vvedi p'); readln(p); </pre>

<pre>110 PRINT "ЧИСЕЛ БОЛЬШИХ, ЧЕМ P"; K</pre>	<pre>for i:=1 to 2 do for j:=1 to 3 do begin a[i,j]:=trunc(random(15)-8); if a[i,j]>=p then k:=k+1; writeln('a[';i;',';j;']=';a[i,j]); end; writeln('k=';k); readkey; end.</pre>
--	---

Примечание. Обратить внимание на организацию проверки условия задачи. Разобрать, какие случайные числа генерируются генератором случайных чисел. Рассказать, что изменится в программе, если условия будут следующие: а) подсчитать количество четных элементов; б) вывести положительные элементы массива; в) найти максимальный элемент массива.

Задача 10. Задан одномерный массив из 9 элементов. Сформировать из него двумерный массив 3x3.

Программа 10.1	Программа 10.2
<pre>10 DIM B(3,3) 20 DIM A(9): 30 FOR I=1 TO 9 40 INPUT(I) 45 PRINT "A("I")=";A(I) 50 NEXT I 60 FOR I=1 TO 3 70 FOR J=1 TO 3 80 K=K+1 90 B(I,J)=A(K) 100 PRINT B(I,J) 110 NEXT J 120 NEXT I</pre>	<pre>Program P10_2; uses crt; var a:array[1..9] of integer; b:array[1..3,1..3] of integer; i,j,k:word; begin for i:=1 to 9 do begin read(a[i]); writeln('a[';i;']=';a[i]); end; k:=0; for i:=1 to 3 do for j:=1 to 3 do begin k:=k+1; b[i,j]:=a[k]; writeln('b[';i;',';j;']=';b[i,j]); end; end; readkey; end.</pre>

Примечание. Разобрать исполнение программы.

Задача 11. Упорядочивание одномерного массива, заданного генератором случайных чисел, по возрастанию.

Программа 11.1	Программа 11.2
<pre> 10 DIM A(10) 20 FOR I = 1 TO 10 30 A(I) = INT(RND(1) * 100) 40 PRINT A(I); 50 NEXT I; PRINT 60 FOR J = 1 TO 10 70 FOR I = 1 TO 10 - J 80 IF A(I)<=A(I+1) THEN 120 90 C = A(I) 100 A(I)=A(I + 1) 110 A(I + 1)=C 120 NEXT I 130 NEXT J: 140 FOR I = 1 TO 10 150 PRINT A(I); 160 NEXT I </pre>	<pre> Program P3_2; uses crt; var i,j,c:integer; a:array[1..10] of integer; begin for i:=1 to 10 do begin a[i]:=trunc(random(100)); write(a[i],' '); end;writeln; for j:=1 to 10 do for i:=1 to 10-j do if a[i]>a[i+1] then begin c:=a[i]; a[i]:=a[i+1]; a[i+1]:=c; end; for i:=1 to 10 do write(a[i],' '); readkey; end. </pre>

Примечание. Разобрать исполнение программы и обратить внимание на организацию сортировки элементов массива. Вручную исполнить выполнение программы.

2.6. УРОК-ПРОЕКТИРОВАНИЕ. РАСЧЕТ НА ЭВМ ОЖИДАЕМОЙ ТОЧНОСТИ ИЗГОТОВЛЕНИЯ ДЕТАЛИ

Данный урок-проект предлагается для усиления прикладной и практической направленности обучения. При проведении урока-проекта можно привлечь учителя технологии производства или заранее проконсультироваться с ним: взять конкретные значения для проведения расчетов. Расчет ожидаемой точности изготовления детали при выполнении операций процесса является составной

частью технологического проектирования. По результатам расчетов выбираются схемы установки деталей в приспособлениях, способы настройки технологической системы (станок, приспособление, инструмент, деталь и др.). Реализацию сформулированной задачи на ЭВМ можно рассматривать как решение одного из вопросов при разработке системы автоматизированного проектирования технологических процессов.

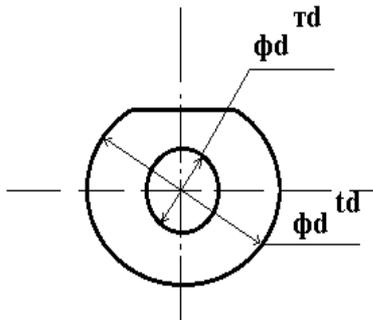


Рис. 11

Определение задачи. Введем следующие ограничения. Рассмотрим операцию фрезерования плоскости у деталей типа «втулка». Операционный эскиз представлен на рис.11, без размера до обрабатываемой поверхности.

На точность положения обрабатываемой поверхности оказывают влияние следующие факторы:

- установка детали в приспособлении;
- погрешности изготовления;
- настройка технологической системы;
- процесс обработки (погрешности, возникающие в результате упругих деформаций системы, станок, приспособление, инструмент, деталь; износ режущего инструмента и т. д.).

Для упрощения задачи исключим из рассмотрения погрешности, связанные с процессом обработки. Для расчета ожидаемой точности обработки необходимы сведения: о заготовке, приспособлении, принятом способе настройки технологической системы, координации обрабатываемой поверхности. В результате расчета должно быть получено ожидаемое рассеяние размера, координирующего обрабатываемую поверхность.

Решение задачи. Смысл расчета ожидаемой точности обработки сводится к определению величины погрешностей, влияющих на рассеяние координирующего размера, и суммированию этих погрешностей. В качестве подзадач решаемой задачи можно выделить следующие:

- определение величины погрешности, связанной с настройкой технологической системы;
- определение величины погрешности, связанной с изготовлением приспособления;

- расчет величины погрешности, связанной с установкой детали;
- суммирование погрешностей.

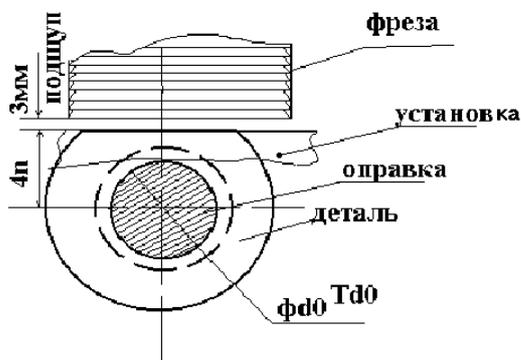


Рис. 12

Погрешность принятого способа настройки технологической системы (δH) зависит от квалификации наладчика и биения фрез. По данным производства и технологическим рекомендациям $\delta H = 0,030 \dots 0,060$ мм. По условию поставленной задачи величина погрешности настройки должна быть задана в исходной информации. Погрешность, связанная с изготовлением приспособления, для рассматриваемого примера – это отклонение положения установки относительно оси поправки. Максимальная величина погрешности равна допуску на размер TLn (см. рис. 12).

По данным производства и технологическим рекомендациям, $TLn = (1/3 \dots 1/5) TLg$, где TLg – допуск на координирующий размер обрабатываемой поверхности.

По условию поставленной задачи, допуск на размер приспособления (TLn) задается в исходной информации. Положение обрабатываемой поверхности у различных деталей может быть закоординировано одним из размеров $L1, L2, L3 \dots L6$, проставленных на расчетной схеме (рис. 13).

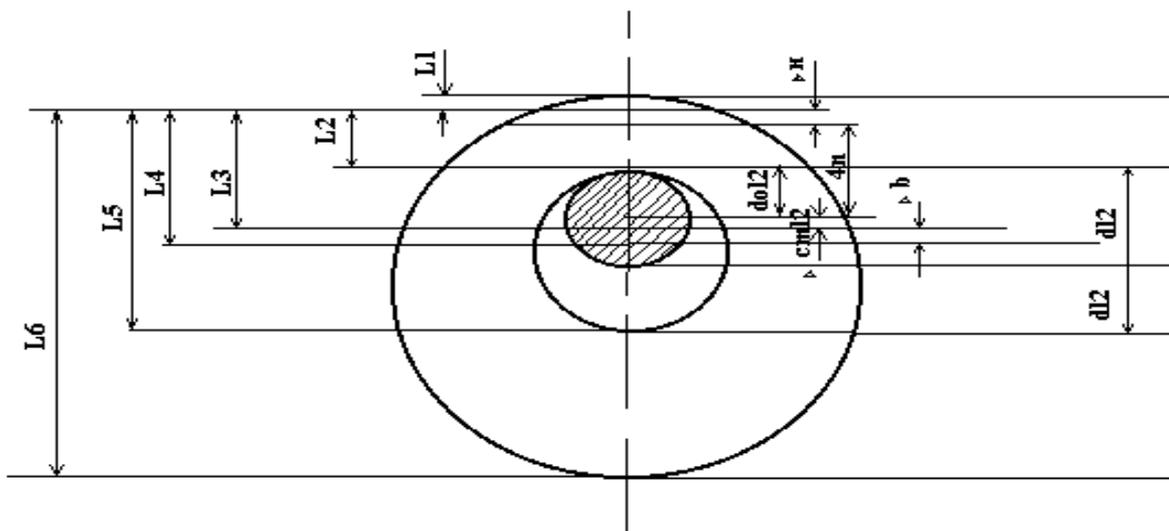


Рис.13

Для наглядности смещение заготовки относительно оправки, погрешность настройки технологической системы, а также возможное смещение осей D и d изображены в одну сторону. Суть расчета возможного рассеяния размеров $L1, L2, L3, \dots L6$ заключается в выполнении и решении соответствующей размерной цепи. Для размера $L1$ уравнение размерной цепи имеет вид

$$L1 + \delta H + L_n + \delta_{cm} + \delta B + d/2 = 0,$$

где $\delta H, \delta_{cm}, \delta B$ рассматриваются как звенья размерной цепи, имеющие номинальное значение размера, равное нулю. Выявленные уравнения размерных цепей могут быть использованы для расчета значений L_n, d_0 , распределения допусков между составляющими звеньями размерной цепи. В поставленной задаче необходимо определить только рассеяние исходных (чертежных) размеров $L1, L2, \dots L6$.

На основании рекомендаций, принятых в литературе, суммирование погрешностей произведем алгебраически. На основании анализа уравнений размерных цепей рассеяние размеров определяются выражениями:

$$wL1=wL6= \sqrt{(T_H^2 + T^2 L_n + T^2 \delta_{cm} + T^2 \delta B + T^2 d/2)} ;$$

$$wL2=wL5= \sqrt{(T_H^2 + T^2 L_n + T^2 \delta_{cm} + T^2 d/2)} ;$$

$$wL3= \sqrt{(T_H^2 + T^2 L_n + T^2 \delta_{cm})} ;$$

$$wL4= \sqrt{(T_H^2 + T^2 L_n + T^2 \delta_{cm} + T^2 B)} .$$

Величина погрешности смещения равна максимальному зазору между оправкой и заготовкой $\delta_{cm}=S_{max}$.

При условии равенства номинальных значений диаметров D и d , задания допуска на D в тело детали величину S_{max} можно рассчитать как

$$S_{max}=HOTd+BOTD,$$

где HOT и BOT – соответственно нижнее и верхнее отклонения допусков диаметров. Ниже приведена программа на языке Бейсик.

Некоторые пояснения к используемым переменным в программе:

HODOM – нижнее отклонение размера заготовки;
 BOD – верхнее отклонение размера заготовки;
 TH – допуск на настройку технологической системы;
 TLD – допуск на размер приспособления;
 TB – допуск биения;
 TDM, TD – допуски на размеры заготовки.

Программа 1.1	Программа 1.2
<pre> 10 INPUT «НИЖНЕЕ ОТ- КЛОНЕНИЕ РАЗМЕРА ОП- РАВКИ»;HODOM 20 INPUT «ВЕРХНЕЕ ОТ- КЛОНЕНИЕ РАЗМЕРА ЗА- ГОТОВКИ»;BOD 30 INPUT «ДОПУСК НА НАСТРОЙКУ ТЕХНОЛОГИ- ЧЕСКОЙ СИСТЕ- МЫ»;TH 40 INPUT «ДОПУСК НА РАЗМЕР ПРИСПОСОБЛЕ- НИЯ»;TLD 50 INPUT «ДОПУСК БИЕ- НИЯ»;TB 60 INPUT «ДОПУСК НА РАЗМЕРЫ ЗАГОТОВ- КИ»;TDM,TD 65 INPUT «НОМЕР ВАРИ- АНТА»;N 70 IF N=1 THEN GOTO 100 71 IF N=2 THEN GOTO 120 72 IF N=3 THEN GOTO 140 73 IF N=4 THEN GOTO 160 80 Smax=HODOM+BOD:TDcm= Smax 90 TLP=TLD/4 95 REM WL - РАССЕЙЯНИЕ РАЗМЕРА 100 WL=SQR(TH^2+TLP^2 +TDCM^2+ TB^2+TDM^2/2) 110 PRINT «РАССЕЙЯНИЕ </pre>	<pre> Program P1_2; uses crt; label 100,120,140,160; var hodom, bod,th,tld,tb,tdm,td,tlp, smax, tdcм, wl :real; n:integer; begin clrscr; {очистка экрана} writeln('Нижнее отклонение разме- ра оправки'); readln(hodom); writeIn('Верхнее отклонение раз- мера заготовки'); readln(bod); writeln('Допуск на настройку тех- нологической системы'); readln(th); writeln('Допуск на размер при- способления '); readln(tld); writeln('Допуск биения'); readln(tb); writeln('Допуск на размеры заготовки'); readln(tdm,td); writeln('Номер варианта'); readln(n); if n=1 then goto 100; if n=2 then goto 120; if n=3 then goto 140; if n=4 then goto 160; smax:=hodom+bod;tdcm:=smax; tlp:=tld/4; {wl – рассеяние размера} 100:wl:=sqrt(th*th+tlp*tlp+tdcm*tdcm+ </pre>

<pre> РАЗМЕРА WL1= WL6=«;WL:STOP 120 WL=SQR(TH^2+TLP^2 +TDCM^2+ TD^2/2) 130 PRINT «РАССЕЯНИЕ РАЗМЕРА WL2=WL5=«;WL:STOP 140 WL=SQR(TH^2+TLP^2+ TDCM^2) 150 PRINT «РАССЕЯНИЕ РАЗМЕРА WL3=«;WL:STOP 160 WL=SQR(TH^2+TLP^2 +TDCM^2+ TB^2) 170 PRINT «РАССЕЯНИЕ РАЗМЕРА WL4=«;WL:STOP </pre>	<pre> tb*tb+tdm*tdm/2); write('Рассеяние размера wl1=wl6=',wl); exit; 120:wl:=sqrt(th*th+tlp*tlp+tdcm*tdcm+ td*td/2); write('Рассеяние размера wl2=wl5=',wl); exit; 140:wl:=sqrt(th*th+tlp*tlp+tdcm*tdcm); write('Рассеяние размера wl3=',wl); exit; 160:wl:=sqrt(th*th+tlp*tlp+tdcm*tdcm+ tb*tb); write('Рассеяние размера wl4=',wl); exit; readkey; end. </pre>
---	---

Данная программа позволяет выполнить четыре варианта расчета. Первый вариант – простановка исходного размера от наружной цилиндрической поверхности, второй – простановка исходного размера от внутренней цилиндрической поверхности. Вариант простановки исходного размера от оси OD – третий, а от оси Od – четвертый.

2.7. УРОКИ-ЛЕКЦИИ. ЧИСЛЕННЫЕ МЕТОДЫ РЕШЕНИЯ ДИФФЕРЕНЦИАЛЬНЫХ УРАВНЕНИЙ

Цель лекций: ознакомление учащихся с численными методами решения дифференциальных уравнений. Материал этих лекций в некоторой степени сложен для школьников, однако, для раздела «Моделирование физических задач» он необходим, поэтому мы излагаем его в упрощенном виде. Учитель по своему усмотрению может сократить объем лекций, в частности, понятия сходимости и устойчивости разностных схем, а также выбрать один или два численных метода решения разностной задачи. Изложение материала этой лекции дается после ознакомления с этапами вычислительного эксперимента (см. главу 6).

Лекция 1

Моделирование физических задач можно проводить разными методами:

- методом прямого программирования на языках Паскаль, Фортан, СИ, Бейсик и других и обработкой результатов на графических пакетах Grafer, Origin и др.;
- с использованием математических пакетов Maple, Mathcad и др.
- с использованием пакета Excel как инструмента для решения математических и физических задач;

Для метода прямого программирования необходимо построение дискретной модели физической задачи, которая предполагает создание вычислительного алгоритма¹, поэтому рассмотрим подробнее второй этап вычислительного эксперимента, т.е. процесс создания алгоритма решения дифференциальных уравнений (математической модели физической задачи).

Процесс создания вычислительного алгоритма можно подразделить на четыре стадии.

1. На первой стадии строится расчетная сетка, или, как говорят, разностная сетка.
2. На второй стадии проводится построение разностных уравнений, иначе говоря, строится дискретная модель исходной модели.
3. На третьей стадии изучаются свойства дискретных моделей.
4. На четвертой стадии осуществляется решение системы алгебраических уравнений каким-нибудь известным методом.

Построение разностной схемы. Продемонстрируем эти стадии на примере решения задачи Коши. Рассмотрим простую задачу для уравнения второго порядка на отрезке.

$$\frac{d}{dx} \left(\frac{du}{dx} \right) = f(x), \quad 0 < x < l, \quad (1)$$

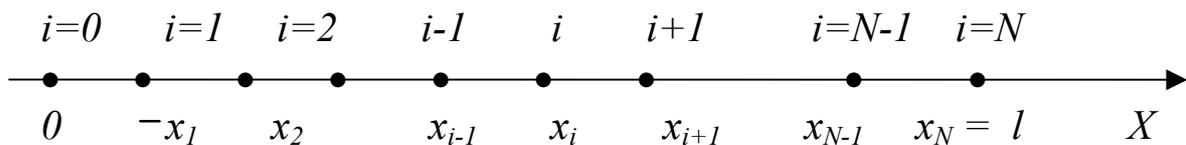
$$\frac{du}{dx} \Big|_0 = C_1,$$

$$u(0) = C_0.$$

¹ Методы численного решения дифференциальных уравнений см., например, в работах [8, 13, 28, 39, 44, 48, 76].

Будем считать, что решение существует и оно единственно.

Область непрерывного изменения аргумента заменяются конечным дискретным набором точек, называемых сеткой. Для нашей задачи заменяем непрерывную область $0 < X < l$ на дискретную совокупность конечного числа точек N . Самый простой способ замены – это *равномерное* деление отрезка $[0, l]$ по правилу $x_i = ih$, $h = l/N$, $0 \leq i \leq N$. Множество $\omega_h = \{x_i\}$, $i = 0, \dots, N$ этих точек представляет собой (равномерную) разностную сетку с шагом h , точки x_i называются узлами разностной сетки.



В разностной сетке имеются внешние и внутренние узлы. Решение $U(x_i)$ ищется только во внутренних узлах, на левом узле значение искомой функции задается.

Функция U рассматривается как функция не непрерывного аргумента, а дискретного аргумента x_i . Функции $U(x_i) = U_i$ называются сеточными функциями: $0 \leq x \leq l$ $0 \leq i \leq N$

Построение разностных (дискретных) уравнений. На этой стадии строится дискретная аналогия дифференциальных уравнений (1) и входных данных по определенным правилам.

Проведем дискретизацию дифференциальных уравнений (1). Это замена производных конечными разностями. Наиболее часто используются односторонние разностные производные $U_{\bar{x}}, U_x$ по аргументу.

$$\frac{du}{dx} \begin{array}{l} \text{Правосторонняя} \\ \text{Левосторонняя} \end{array} \begin{array}{l} U_x = \frac{U(x+h) - U(x)}{h} = \frac{U_{i+1} - U_i}{h} \\ U_{\bar{x}} = \frac{U(x) - U(x-h)}{h} = \frac{U_i - U_{i-1}}{h} \end{array} \quad (2)$$

Первые две выражения из (2) – дискретные аппроксимации производной $\frac{du}{dx}$, для получения которой достаточно использовать значения функций $U(x)$ лишь в двух точках (двухточечный шаб-

лон). Эти выражения получены в определении производной в предположении малого шага сетки. Отметим, что под шаблоном аппроксимации в узле понимают набор узлов, значения в которых входят в данную аппроксимацию.

По определению производной,

$$\frac{du}{dx} = \lim_{h \rightarrow 0} \frac{U(x+h) - U(x)}{h}.$$

Если освобождаемся от знака предела, то возникает погрешность, называемая погрешностью аппроксимации. Проведем оценку этой погрешности. Аппроксимация производных может быть представлена в виде

$$\begin{cases} \frac{du(x)}{dx} = \frac{U(x+h) - U(x)}{x+h-x} + o(h) = U_x + o(h), \\ \frac{du(x)}{dx} = \frac{U(x) - U(x-h)}{x-(x-h)} + o(h) = U_{\bar{x}} + o(h). \end{cases} \quad (3)$$

Разложение функции $f(x)$ в окрестности точки x_0 в ряд Тейлора в общем случае имеет вид:

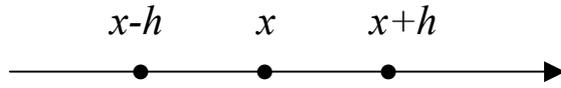
$$f(x) = f(x_0) + f'(x_0)(x - x_0) + \frac{1}{2!} f''(x_0)(x - x_0)^2 + \dots \quad (4)$$

Разложим теперь функцию $U(x)$ в ряд Тейлора:

$$\begin{aligned} u(x+h) &= U(x) + U'(x)h + U''(x)\frac{h^2}{2} + U'''(x)\frac{h^3}{6} + O(h^4), \\ u(x-h) &= U(x) - U'(x)h + U''(x)\frac{h^2}{2} - U'''(x)\frac{h^3}{6} + O(h^4). \end{aligned} \quad (5)$$

Сравнивая эти разложения с (3), получим, что погрешность аппроксимации $U_x - u'(x)$ является величиной порядка $o(h)$ при $h \rightarrow 0$. В этом случае говорят, что имеет место аппроксимация первого порядка. Использование односторонней разности во многих случаях дает удовлетворительные результаты.

Рассмотрим теперь несколько другую, так называемую центральную разностную схему (трехточечный шаблон):



Найдем разность функций правого и левого разложения в ряд Тейлора:

$$U(x+h) - U(x-h) = 2U'(x)h + U'''(x)\frac{h^3}{6} + O(h^5). \quad (6)$$

Производная может быть представлена в виде

$$\frac{du}{dx} = \frac{U(x+h) - U(x-h)}{2h} + O(h^2) = U'(x) + U'''(x)\frac{h^2}{12} + O(h^4) = U'(x) + O(h^2) \quad (7)$$

Сравнение с (3) показывает, что определение по центральной разностной схеме точнее аппроксимирует производную, чем правая или левая разностная схема, и имеет второй порядок аппроксимации.

Вторую производную функции U можно представить в виде

$$\frac{d}{dx} \left(\frac{du}{dx} \right) = U_{xx} + O(h^2), \quad (8)$$

где

$$U_{xx} = \frac{U_x - U_{\bar{x}}}{h} = \frac{U(x+h) - U(x) - U(x) + U(x-h)}{h^2} = \frac{U(x+h) - 2U(x) + U(x-h)}{h^2}.$$

Подставляя разложение в ряд Тейлора, можно показать, что

$$U_{xx} - U'' = \frac{h^2}{12} U^{(4)} + O(h^4), \quad (9)$$

т.е. вторая разностная производная аппроксимирует $u''(x)$ со вторым порядком.

Таким образом, мы левую часть уравнения (1) аппроксимировали.

Правую часть можно представить в виде:

$$f(x) \rightarrow f(x_i), \quad i = 0, \dots, N.$$

Тогда вместо уравнения (1) имеем дискретное уравнение

$$U_{x\bar{x}} = f(x_{i-1}).$$

С граничными условиями

$$U_{\bar{x}}(x_1) = C_1,$$

$$U(x_0) = C_0.$$

В другой форме записи

$$\begin{cases} \frac{U_{i+1} - 2U_i + U_{i-1}}{h^2} = f_{i-1}, & i = 1, \dots, N-1, \\ \frac{U_i - U_{i-1}}{h} = C_1, & U_0 = C_0. \end{cases} \quad (10)$$

Система уравнений (10) называется разностной системой уравнений, или дискретным аналогом уравнения (1). В краткой форме систему уравнений (10) записывают в виде

$$L_h U^{(h)} = f^{(h)}, \quad (11)$$

где $L_h U = \frac{U_{i+1} - 2U_i + U_{i-1}}{h^2}$ называют отображением или разностным оператором, здесь индекс по i опущен. Предполагается, что начальные и граничные условия содержатся L_h и в правой части.

Таким образом, для построения дискретного аналога математической модели мы должны решить следующие задачи:

1. Построить сетку, на которой ищется решение $U^{(h)}$ разностного уравнения.

2. Построить разностный оператор (отображение) L_h , действующий на сеточную функцию $f^{(h)}$.

3. Определить множество сеточных функций $f^{(h)}$, также являющееся областью значений оператора L_h (причем они могут быть определены на сетке, не совпадающей с сеткой для $U^{(h)}$).

Лекция 2

Стадия исследования вычислительного алгоритма. Для получения физически правильного решения необходимо иметь два условия: хорошую аппроксимацию уравнений; устойчивость и сходимость алгоритма.

Прежде чем рассмотреть эти условия, введем некоторые понятия теорий разностных уравнений. Запишем дискретную модель – аналог уравнения (1) в виде

$$U_i'' = \varphi_i \quad \text{или} \quad (U_{x\bar{x}})_i = \varphi_i. \quad (11)$$

Пусть U_i – приближенное решение дискретной модели, u_i – точное решение уравнения (1), тогда $u_i = U_i + \delta u$. Разность $\delta u = u_i - U_i$ называется погрешностью решения.

Определение 1. Если

$$\delta u_i = O(h^\alpha), \quad i = 1 \dots N \quad \alpha > 0, \quad (12)$$

то говорят, что разностная схема (11) сходится (с порядком α) и $\delta u \rightarrow 0$ при $h \rightarrow 0$ для всех i .

При уменьшении шага h (измельчении сетки) U_i хорошо аппроксимирует точное решение $u(x)$ в узлах x_i . Промежуточное значение определяется простой интерполяцией.

Определение 2. Если u_i'' – вторая производная $u_i = u(x_i)$ и $U_{x\bar{x}}$ – аппроксимация второй производной, то

$$L_h \delta u = u_i'' - U_{x\bar{x}} = f_i - L_h U_i = f_i - \varphi_i, \quad i = 1, \dots, N-1 \quad (13)$$

называется погрешностью аппроксимации дифференциального оператора, или невязкой.

Определение 3. Если

$$L_h \delta u = u'' - U_{x\bar{x}} = O(h^\beta), \quad i = 1, \dots, N-1 \quad \beta > 0, \quad (14)$$

то говорят, что имеет место аппроксимация. При этом непрерывный оператор приближается к дискретному оператору с порядком β .

Для оценки величины скаляра используется его модуль, для вектора – его *норма*. Существует несколько определений нормы

вектора. Мы будем определять его как максимум модуля компонент, т.е. если задан вектор U_i , то

$$\|U\| = \max_i |U_i|.$$

Определение 4. Разностная схема называется устойчивой, если для любых входных φ, U_1, U_2 выполняется неравенство

$$\|U\| = \max |U_i| \leq C \|\varphi\| = C \max |\varphi_i|, \quad (15)$$

где $C > 0$ и не зависящая от h постоянная, $\|\varphi\|$ – норма вектора правой части уравнения (1) равна максимальному значению компонент вектора φ .

Сходимость разностной схемы к исходной задаче, т.е. стремление δu к нулю, может быть обеспечена двумя условиями:

1) по условию аппроксимации,

$$f - L_h U \rightarrow 0 \quad (\text{по норме!}) \text{ при } h \rightarrow 0. \quad (16)$$

2) Второе условие зависит от свойств разностной задачи. Разностный оператор L_h должен быть таким, что при любых h решение задачи (13) имело тот же порядок, что и правая часть, т.е.

$$\delta u \approx f - LU. \quad (17)$$

Это условие называется условием устойчивости. Отметим, что таким свойством обладает не всякий разностный оператор.

Уравнения (11) и (14) отличаются только обозначениями, поэтому условие (15) можно переписать в виде

$$u \approx f. \quad (18)$$

Для придания точности условиям (16) и (18) необходимо записать эти условия через их нормы, т.е. $\|u\| \approx \|f\|$.

Вывод. Таким образом, сходимость $\|U - u\| \rightarrow 0$ при $h \rightarrow 0$ любой линейной разностной задачи вытекает из аппроксимации $\|LU - f\| \rightarrow 0$ при $h \rightarrow 0$ и устойчивости $\|u\| \approx \|f\|$ или, более точнее, $\|u\| \leq C \|f\|$, где $C > 0$ и не зависящая от h постоянная.

Методы решения разностных уравнений. Систему уравнений (10) можно записать в виде

$$\begin{cases} U_{i+1} = 2U_i - U_{i-1} + h^2 f_{i-1} & i = 1, \dots, N-1, \\ U_1 = U_0 + hC_1 \\ U_0 = C_0. \end{cases} \quad (19)$$

При решении уравнений (1) по первой производной использовалась аппроксимация (3). Обычно при решении уравнений второго порядка первая производная считается известной, часто задается равной постоянной величине.

Система уравнений (19) представляет собой систему алгебраических уравнений, которую можно решить методом последовательных итераций по шагу h , и поэтому говорят, что найдено численное решение системы (1).

Иначе говоря, построен алгоритм решения уравнения (1), т.к. по известному граничному значению U_0 и сеточным функциям f_i можем определить все значения сеточной функции U_i методом итерации. Согласно этому алгоритму разрабатывается программа решения на одном из языков программирования.

Мы нашли решение уравнения (1), однако отметим, существует различные аппроксимации, которые являются также решениями уравнение (1), т.е. существуют различные методы численного решения дифференциальных уравнений. Рассмотрим эти методы, на примере задачи Коши для дифференциального уравнения первого порядка

$$\begin{aligned} \frac{du}{dx} &= f(x, u), & 0 < x < l, \\ u(0) &= C_0. \end{aligned} \quad (20)$$

Метод Эйлера. Воспользовавшись двухточечным шаблоном, построим для уравнения (18) его разностный аналог:

$$\begin{cases} \frac{U_{i+1} - U_i}{h} = f_i, & i = 0, 1, \dots, N, \\ U_0 = C_0. \end{cases} \quad (21)$$

Тогда приближенное значение $U(x_{i+1})$ находится по формуле

$$\begin{cases} U_{i+1} = U_i + hf_i(x_i, U_i) & i = 0, 1, \dots, N, \\ U_0 = C_0, \end{cases} \quad (22)$$

т.е. все последующие значения U_1, U_2, \dots, U_N определяются из уравнения (11) методом итераций.

Формула (21) является основной формулой метода Эйлера, или метода ломаных. Из формул (5) и (7) следует, что формула Эйлера обладает вторым порядком точности по шагу и первого порядка аппроксимации на интервале.

На рис.14 показано геометрическое представление метода Эйлера, откуда видно, что точка U_{i+1} не лежит точно на интегральной кривой $u = u(x)$ и $U_{i+1} - u_i$ определяет погрешность аппроксимации $O(h^2)$.

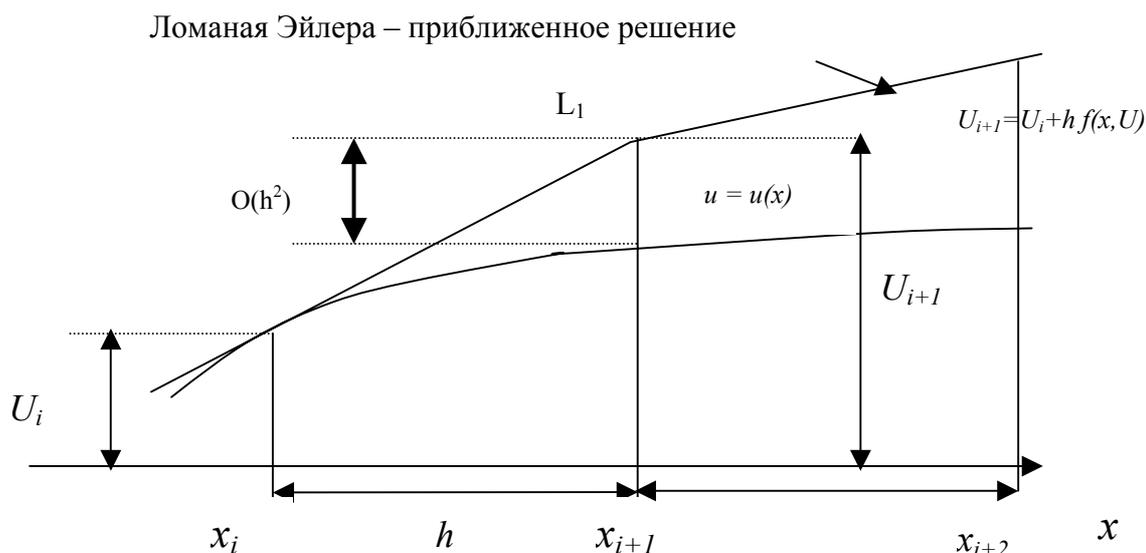


Рис. 14. Геометрическое представление метода Эйлера

Практическую оценку погрешности решения, найденного на сетке с шагом $h/2$, в точке x_i производят с помощью приближенного равенства – **правила Рунге**:

$$|u(x_i) - U_i(h/2)| \approx \frac{|U_i(h) - U_i(h/2)|}{2^k - 1}, \quad (23)$$

где k – порядок точности численного метода.

Для метода Эйлера $k=2$, поэтому

$$O(h^2) = |u(x_i) - U_i(h/2)| \approx \frac{|U_i(h) - U_i(h/2)|}{3}.$$

Из формулы (21) следует, что для оценки результата необходимо проводить вычисления дважды: один раз с шагом h , другой – с шагом $h/2$ на одном и том же отрезке.

Лекция 3

Метод Эйлера-Коши, или исправленный метод Эйлера. В этом методе значение U_{i+1} находится по формуле

$$\left\{ U_{i+1} = U_i + \frac{h}{2} [f(x_i, U_i) + f(x_{i+1}, U_{i+1})] \right. , \quad (24)$$

т.е. вместо тангенса угла наклона касательной L_1 к интегральной кривой в точке x_i, U_i , который применяется в методе Эйлера, используется полусумма значений тангенсов углов наклона касательных L_1 и L_2 в известной x_i, U_i и искомой точке x_{i+1}, U_{i+1} (рис.15). Для нахождения функции $\{f(x_{i+1}, U_{i+1})\}$ разложим $f(x, U)$ в ряд Тейлора:

$$f(x, U) = f(x_i, U_i) + (x - x_i) \frac{\partial f}{\partial x} + (U - U_i) \frac{\partial f}{\partial y} + \dots$$

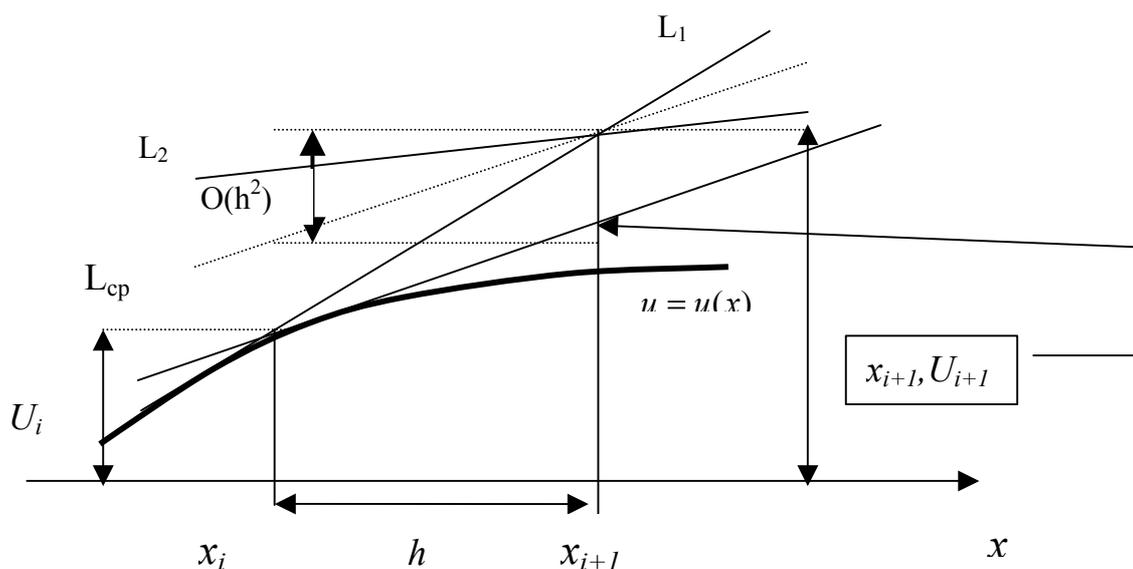


Рис.15. Геометрическое представление исправленного метода Эйлера

Подставляя в эту формулу $x = x_i + h$, $U = U_i + h U'$ и учитывая, что $U'_i = f(x_i, U_i)$, получаем

$$f(x_i + h, U_i + hU'_i) = f + hf'_x + hf \cdot f'_y + O(h^2),$$

где снова функция и ее производные вычисляются в точке x_i, U_i .

Подставляя результат в (23), получаем

$$U_{i+1} = U_i + hf(x_i, U_i) + \frac{h^2}{2}(f'_x + f(x_i, U_i) \cdot f'_y) + O(h^3) \quad (25)$$

Как видим, исправленный метод Эйлера согласуется с разложением в ряд Тейлора вплоть до членов степени h^2 , т.е. является методом третьего порядка точности по шагу интегрирования h или методом второго порядка аппроксимации на интервале $[x_i, x_{i+1}]$.

Оценку погрешности по шагу можно произвести по правилу Рунге:

$$O(h^3) = |u(x_i) - U_i(h/2)| \approx \frac{|U_i(h) - U_i(h/2)|}{5} \quad (26)$$

Модифицированный метод Эйлера, или метод Рунге-Кутты второго порядка. В этом методе проведем усреднение точки в отличие от исправленного метода Эйлера, где усреднялись наклоны касательных.

Для этого берем точку P , лежащую на пересечении прямой L_1 и ординаты $x_{i+1/2} = x_i + h/2$. Проведем мысленно касательную к кривой $u = u(x)$ и перенесем эту касательную в точку P и O . На рис.16 – это прямые L_0 и L^* . Пересечение прямой L_0 с ординатой $x_{i+1} = x_i + h$ дает искомую точку x_{i+1}, U_{i+1} .

Тогда алгоритм решения можно записать в виде

$$\begin{cases} U_{i+1} = U_i + hf(x_i + \frac{h}{2}, U_i + \frac{h}{2} f(x_i, U_i)), & i = 0, 1, \dots, N, \\ U_0 = C_0. \end{cases} \quad (27)$$

Запишем следующее уравнение :

$$U_{i+1} = U_i + h[(1-b)f(x_i, U_i) + bf(x_i + \frac{h}{2b}, U_i + \frac{h}{2b} f(x_i, U_i))] \quad (28)$$

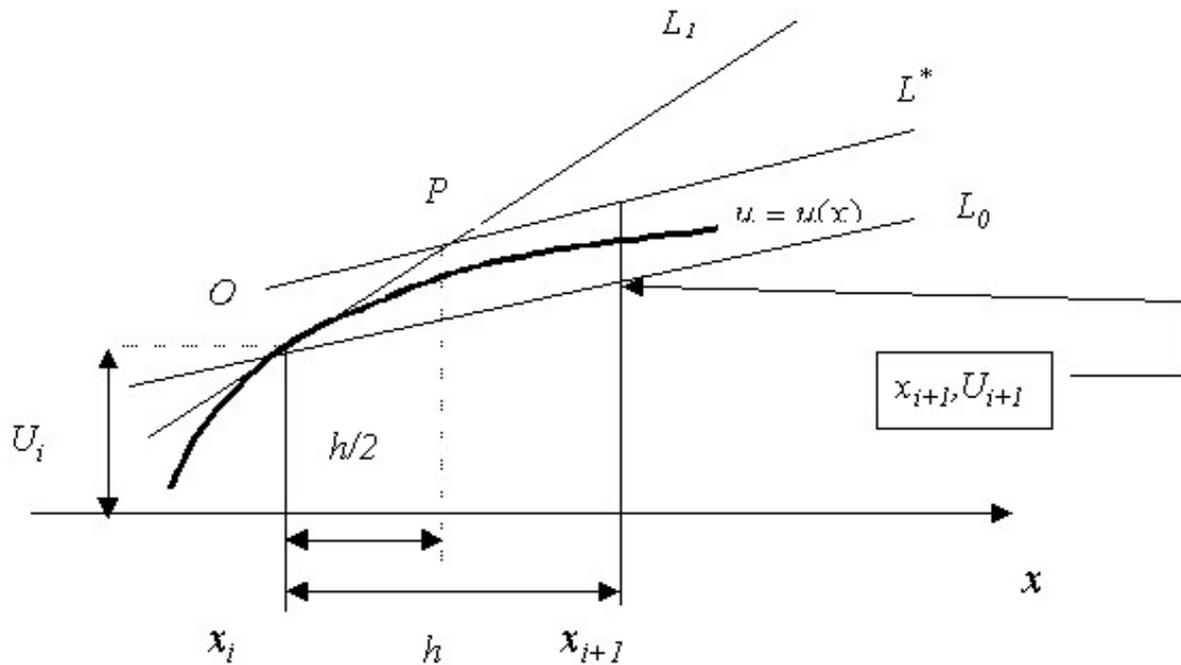


Рис.16. Геометрическое представление модифицированного метода Эйлера

Из данной схемы можно получить все разностные схемы, рассмотренные выше:

- при $b = 0$ получим метод Эйлера;
- при $b = 1/2$ имеем метод Эйлера-Коши, или исправленный метод Эйлера;
- при $b = 1$ имеем модифицированный метод Эйлера.

Уравнение (26) является общей формой записи метода Рунге-Кутты второго порядка.

Метод прогноза-коррекции. Нелинейное уравнение (21) можно также решить методом многократной итерации. Запишем уравнение (21) в виде

$$\left\{ U_{i+1}^{(k+1)} = U_i + \frac{h}{2} [f(x_i, U_i) + f(x_{i+1}, U_{i+1}^{(k)})] \right. \quad , (29)$$

здесь индекс k означает номер итерации.

В качестве нулевой итерации используется значение $U_{i+1}^{(0)}$, вычисленное по методу Эйлера. Тогда вычислительный алгоритм первой итерации имеет вид:

$$\begin{cases} U_{i+1}^{(0)} = U_i + hf_i(x_i, U_i), & i = 0, 1, \dots, N, \\ U_{i+1}^{(1)} = U_i + \frac{h}{2}[f(x_i, U_i) + f(x_{i+1}, U_{i+1}^{(0)})] \\ U_0 = C_0. \end{cases} \quad (30)$$

По методу Эйлера находим приближенное (прогнозируемое) решение, затем это решение уточняется (корректируется).

На втором шаге итерации можно для прогнозирования использовать центральную разностную схему (7). Это связано с тем, что погрешность аппроксимации для такой схемы равна $O(h^2)$.

$$\begin{cases} U_{i+1}^{(1)} = U_{i-1} + 2hf_i(x_i, U_i), & i = 0, 1, \dots, N, \\ U_{i+1}^{(2)} = U_i + \frac{h}{2}[f(x_i, U_i) + f(x_{i+1}, U_{i+1}^{(1)})]. \end{cases} \quad (31)$$

Метод прогноза-коррекции является методом третьего порядка по шагу и методом второго порядка на интервале, так же как и исправленный метод Эйлера. Реально метод решения напоминает метод стрельбы (недолет, перелет), т.е. дает двухстороннее приближение к решению. Итерационный процесс (30) или (31) можно продолжать до достижения заданной точности.

Обычно на практике используют одну или две итерации, т.к. дальнейшее уточнение не повышает порядок точности.

Геометрическое предсказание сводится к тому, что находится угол наклона касательной в точке x_i, U_i (прямая L_1 на рис.17). После этого через точку (x_{i-1}, U_{i-1}) проводится прямая L , параллельная L_1 . Предсказанное значение $U_{i+1}^{(0)}$ будет расположено там, где прямая L_1 пересечется с ординатой $x=x_{i+1}$.

Рассмотрим геометрическое представление корректирующей формулы (29). Для этого найдем наклон касательной в точке x_{i+1} к интегральной кривой. Эта касательная показана на рис. 17 как L_2 . Усредним тангенсы углов L_1 и L_2 , получаем линию \bar{L} . Проведем через точку x_i, U_i линию L , параллельную \bar{L} . Тогда точка пересечения этой линии с ординатой $x=x_{i+1}$ дает новое приближение к U_{i+1} .

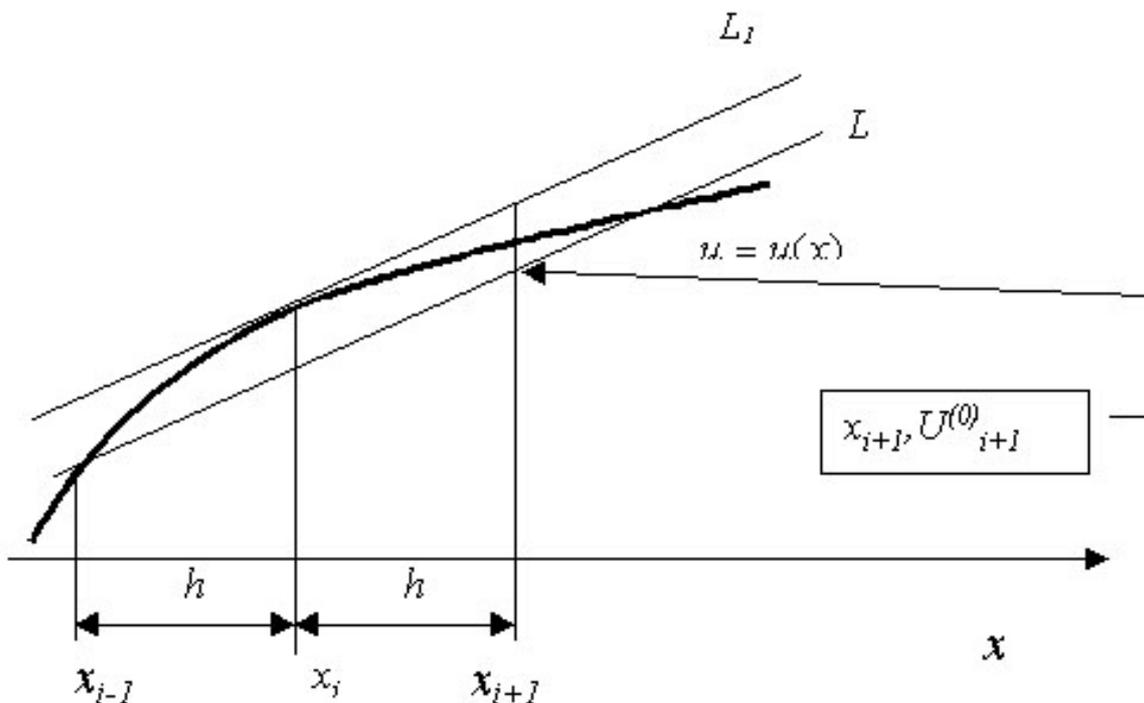


Рис.17. Геометрическое представление прогнозирующей формулы (31)

В методе прогноза-коррекции получается хорошая оценка погрешности по шагу интегрирования. Для разностной схемы (27) погрешность вычислений оценивается по формуле

$$O(h^3) = -\frac{h^3}{12} u''' = \frac{1}{5} |U_{i+1}^{(k-1)} - U_{i+1}^{(k)}|. \quad (32)$$

Здесь k – порядок итерации. Величины, которые используются для этой оценки, берутся из результатов вычислений.

Таким образом, можно выделить одноступенчатые методы (методы Рунге-Кутты) и многоступенчатые методы (методы прогноза коррекции). Анализ достоинств и недостатков этих методов показывает:

1. В методах Рунге-Кутты используется информация только об очередной точке и не используется информация о ранее найденных точках в отличие от методов прогноза-коррекции, поэтому для нахождения решения, максимально приближенного к истинному решению, вначале можно использовать методы Рунге-Кутты, затем применять методы прогноза и коррекции.

2. При использовании методов Рунге-Кутты приходится многократно вычислять функцию $f(x, u)$. Это приводит к большим затра-

там машинного времени, в отличие от методов прогноза коррекции, где используется информация о ранее найденных точках. Отметим, что если используется многократная итерация, то экономия машинного времени может быть незначительной.

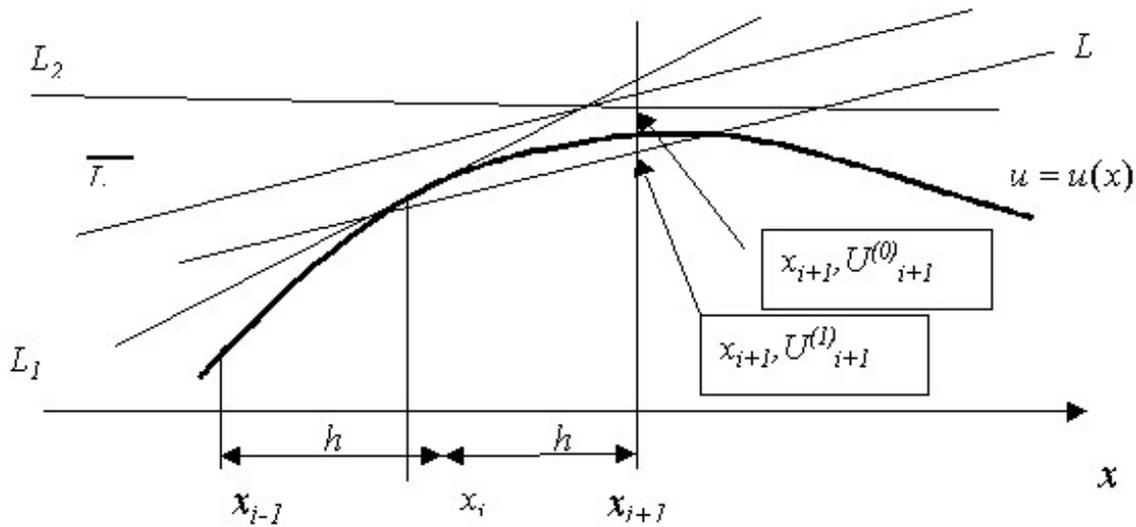


Рис.18. Геометрическое представление корректирующей формулы (31)

Отметим, что в отличие от методов Рунге-Кутты, которые являются частично устойчивыми, метод прогноза коррекции является абсолютно устойчивым, т.е. абсолютная ошибка не возрастает.

Выбор шага интегрирования. Выбор метода решения зависит от характера физической задачи. Для первой оценки решения часто используют метод Эйлера. Уменьшение шага интегрирования для данного метода в 10 раз уменьшает погрешность примерно в 10 раз. Этот метод имеет большую погрешность аппроксимации; кроме того, очень часто оказывается неустойчивым – малая ошибка (происходящая от погрешности аппроксимации, округления или заложенная в исходных данных) увеличивается с ростом x .

Использование методов третьего порядка точности по шагу интегрирования уменьшает погрешность вычислений. Уменьшение шага интегрирования для этих методов в 10 раз уменьшает погрешность примерно в 100 раз. Поэтому при моделировании физического явления необходимо начинать с более точных методов.

В начальной стадии решения задачи всегда стоит вопрос: какой шаг интегрирования выбрать для решения дискретной модели?

Рассмотрим последовательность определения шага интегрирования, при котором решение дискретной модели будет прибли-

жаться к истинному решению. На первом этапе находим при шаге интегрирования $h < 1$. Уменьшим шаг интегрирования, тогда получим другое решение. Уменьшая шаг интегрирования все время, получим семейство решений. Начиная с некоторого малого шага, изменения искомых величин становятся малыми и в некотором смысле контролируруемыми.

Определим количественные критерии сравнения решений при разных шагах интегрирования.

Для выбранного значения шага h находится «грубое» решение $U_{i+1}(h)$. Затем шаг уменьшается в два раза и заново находится уточненное значение $U_{i+1}(h/2)$ на том же отрезке. Если процесс устойчивый, то решения разностных уравнений для шагов интегрирования h и $h/2$ будут сближаться. Относительную погрешность можно оценить как

$$\frac{|U_{i+1}(h/2) - U_{i+1}(h)|}{|U_{i+1}(h/2)|} < \varepsilon_0,$$

где ε_0 – наперед заданное положительное число, определяющее точность вычислений. Если левая часть будет меньше чем ε_0 , то будем считать, что шаг интегрирования удовлетворяет нас и можно проводить вычислительный эксперимент.

Оценку насколько полученное решение достоверно, помогают также интегралы движения (законы сохранения) для данной физической задачи. Выполнение законов сохранения в процессе вычисления искомых величин является одним из эмпирических методов проверки (тестирования) алгоритма решения физической задачи.

После этих лекций можно приступить к практической реализации вычислительного эксперимента на примере физических задач, изложенных в главе 6. Технология обработки, полученная в результате численного решения информации в графическом пакете Microsoft Origin, изложена в главе 7.

ГЛАВА 3. ВНЕКЛАССНАЯ РАБОТА ПО ИНФОРМАТИКЕ В ШКОЛЕ

3.1. ЗАДАЧИ, ОСОБЕННОСТИ И ФОРМЫ ВНЕКЛАССНЫХ ЗАНЯТИЙ ПО ИНФОРМАТИКЕ, ПРОГРАММИРОВАНИЮ, ИНФОРМАЦИОННЫМ ТЕХНОЛОГИЯМ И АРХИТЕКТУРЕ ЭВМ И СЕТЕЙ

Воспитание ответственного отношения к учебе, интереса к занятиям, увлеченности наукой проводятся в основном на уроке. Но учитель ограничен рамками школьной программы. Удовлетворение запросов и интересов школьников, развитие их склонностей и дарований, поддержание устойчивого интереса к предмету осуществляются в значительной степени через внеклассную работу.

В задачи внеклассной работы по информатике входят:

1. Углубление знаний теоретических основ информатики, программирования, изучение архитектуры ЭВМ и сетей, знакомство и работа с программным обеспечением.
2. Популяризация и изучение достижений в области информационных технологий.
3. Привитие учащимся навыков работы с компьютером и программным обеспечением, интереса к исследовательской работе.
4. Воспитание интереса к чтению как обычной, так и электронной научно-популярной литературы, формирование умений и навыков в работе с ними.
5. Популяризация знаний среди остальных учащихся школы.
6. Работа в кабинете информатики.
7. Профессиональная ориентация учащихся.

Внеклассные занятия оказывают положительное влияние и на классные занятия, так как учащиеся, члены кружков более тщательно, углубленно изучают учебный материал, читают дополнительную литературу, осваивают работу с компьютером. Внеклассные занятия провоцируют и самостоятельное изучение основ информатики и вычислительной техники.

Особенности внеклассной работы по информатике:

1. По своему содержанию она строго не регламентирована государственной программой. На внеклассных занятиях материал предлагается в соответствии со знаниями и умениями учащихся: при подборе заданий и материала желательно ориентироваться на определенный уровень подготовки учащихся, но это не обязательно. Надо исходить только от общего уровня развития учащихся.

2. Если обычные уроки планируются по 45 минут, то внеклассные занятия в зависимости от содержания и формы проведения могут быть рассчитаны и на 5-10 минут (занимательные переменки), и на 1,5-2 часа. При проведении уроков необходимо придерживаться санитарных норм², пункты 9.4.8–9.4.20.

3. Если классно-урочная форма требует постоянного состава учащихся, объединенных в коллектив по возрастному признаку, с учетом микрорайона жительства, то для внеклассной работы учащиеся могут объединяться в группы, обучаясь либо в одном и том же классе, либо в разных классах; при этом группы создаются на добровольных началах. Состав учащихся, даже при наличии одной и той же формы внеклассной работы, может меняться.

4. Внеклассная работа характеризуется многообразием форм и видов: групповые занятия, викторины, вечера, олимпиады, кружки, заочные и дистанционные формы обучения.

5. Особенностью внеклассной работы по информатике является занимательность предлагаемого материала либо по содержанию, либо по форме, более свободное выражение своих чувств во время работы, более широкое использование игровых форм проведения занятий и элементов соревнования.

6. Другой особенностью внеклассной работы по информатике является оснащённость кабинета информатики соответствующим техническим оборудованием: наличие достаточного количества компьютеров, соответствующих современному уровню развития вычислительной техники; наличие периферийных устройств (принтеров, сканеров, систем мультимедиа); наличие локальной и глобальной сети; научной и научно-популярной периодической и обычной литературы.

² Санитарные правила и нормы СанПиН 2.2.2.542-96 «Гигиенические требования к видеодисплейным терминалам, персональным электронно-вычислительным машинам и организации работы» (утв. постановлением Госкомсанэпиднадзора РФ от 14 июля 1996 г. № 14).

7. Немаловажным фактором внеклассной работы по информатике является перспектива развития информатики и информационных технологий, которая предполагает использование компьютеров в профессиональной деятельности как в будущем, так и в обычной, повседневной деятельности уже сегодня.

Однако внеклассная работа имеет и общие черты с классно-урочной.

1. В обоих видах работы в процессе обучения школьников соблюдаются одни и те же дидактические принципы: научность, сознательность и активность учащихся, наглядность, индивидуальный подход, связь обучения с практикой.

2. Оба вида работы как две части единого учебно-воспитательного процесса не только содействуют формированию знаний, умений, навыков, но и подготовке будущего гражданина, способного свободно ориентироваться в обществе с высокоразвитыми информационными технологиями.

Формы внеклассной работы по предметам в школе разнообразны. Накоплен огромный опыт внеклассной работы по различным дисциплинам. Умелое сочетание опыта внеклассной работы со спецификой предмета информатики обеспечат успех проведения внеклассных занятий по информатике. Формы внеклассной работы классифицируются по разным признакам: охвату учащихся, времени проведения, систематичности, дидактической цели и т.д. По систематичности можно выделить эпизодические внеклассные мероприятия и постоянно действующие внеклассные организации (работающие, по крайней мере, в течение учебного года).

К первому виду относятся:

1. Подготовка и проведение школьных олимпиад по информатике; участие в районных, городских олимпиадах.
2. Летние компьютерные лагеря.
3. Выпуск стенной газеты.
4. Проведение викторин, вечеров, КВН по информатике.
5. Проведение тематических конференций и семинаров по информатике.

Ко второму виду внеклассных занятий относятся:

1. Разнообразные по формам, задачам кружки и факультативные занятия по информатике.
2. Школьные научные общества.

3. Организация различных форм заочного и дистанционного обучения учащихся.

Приведенное деление внеклассных мероприятий, как свидетельствует практика школ, имеет условный характер: в живой внеклассной работе нередко одни формы порождают другие, разветвляются на несколько новых форм. Например, хорошо работающий кружок может выступить организатором вечера или инициатором выпуска стенной газеты по информатике.

Индивидуальная работа есть во всех видах внеклассных занятий, она может выражаться в чтении литературы, в подготовке материала для стенгазеты, вечера, конференции, викторине, в обучении в заочных или дистанционных школах и т. д.

Массовая работа выражается в проведении вечеров, конференций, конкурсов и олимпиад. Разнообразие форм внеклассной работы делает ее интересной.

3.2. КРУЖКИ ПО ИНФОРМАТИКЕ

Работа кружков по информатике является основной формой внеклассных занятий по информатике, по содержанию связана с программой школьного курса информатики.

3.2.1. Примерный план работы кружка «Архитектура ЭВМ и сетей»

1. Архитектура ЭВМ. Назначение и функции базовых аппаратных средств.
2. Системы счисления.
3. Алгебра логики. Физические основы алгебры логики.
4. Процессор. Регистр. Стек. Память.
5. Система команд процессора. Примеры записи команд процессора.
6. Некоторые понятия языка ассемблер.
7. Архитектура IBM PC.
8. Логическое распределение оперативной памяти.
9. Элементы памяти и их особенности.
10. Процессоры и их характеристики.
11. Контроллеры. Аппаратные прерывания.
12. Шины.

13. Внешние запоминающие устройства.
14. Видеокарта.
15. Мониторы.
16. Клавиатура. Мышь.
17. Локальные сети.
18. Глобальные вычислительные сети.

3.2.2. Примерный план работы кружка «Алгоритмизация и программирование»

1. Понятие алгоритма, свойства, способы описания.
2. Алгоритмизация как базовая составляющая технологического процесса создания программного продукта.
3. Принципы разработки схем алгоритмов, программ, данных и систем.
4. Арифметические и логические основы программирования, формы представления и кодирования информации.
5. Эволюция языков программирования, их классификация.
6. Исходный, объектный и загрузочный модули, связь этих понятий с компонентами программирования.
7. Виды программирования.
8. Области применения, принципы и методы построения программ.
9. Программирование на языке высокого уровня.
10. Основные элементы языка. Числа. Литерные величины. Имена. Переменные. Выражения. Операции. Комментарии.
11. Общая структура программ. Заголовок. Описания. Тело программы. Ввод и вывод.
12. Ветвление, выбор, циклы, безусловный переход.
13. Массивы. Описание. Ввод-вывод массивов. Операции над массивами.
14. Вспомогательные алгоритмы. Процедуры и функции.
15. Множества и операции над ними.
16. Перечисляемые типы программиста.
17. Записи.
18. Работа с файлами.
19. Комбинированный тип данных.
20. Динамические структуры данных.

3.2.3. Примерный план работы кружка «Информационные технологии»

1. Информационные технологии.
2. Решение задач с помощью компьютера.
3. Постановка задачи.
4. Построение модели.
5. Разработка алгоритма.
6. Анализ результатов.
7. Текстовый редактор.
8. Хранение текстов.
9. Печать текстов.
10. Формат печати.
11. Графический редактор.
12. База данных.
13. Операции над данными.
14. Электронная таблица.
15. Пакет прикладных программ.
16. Телекоммуникация.
17. Телекоммуникационная сеть.
18. Электронная почта.
19. Телеконференция.
20. Мультимедиа технология.

3.2.4. Роботландский сетевой университет

Роботландский сетевой университет планирует следующие курсы (Информатика. 2000. №15):

1. Введение в информатику.
2. Компьютерное конструирование.
3. Азы программирования.
4. Буки программирования.
5. Введение в Интернет.
6. Интернет-конструирование.
7. Интернет-программирование.
8. Английский с компьютером.

В университет принимаются индивидуальные и коллективные ученики. Учитель и курируемый им детский коллектив образуют в

рамках курсов активный школьный сетевой узел – команду, которая связана с куратором университета и другими командами в едином информационном пространстве. Эта связь крепится активной совместной деятельностью. Основной канал связи – электронная почта.

3.2.5. Клубы по интересам

Любительским объединением, клубом по интересам является организованная форма общественной самодеятельности учащихся, созданная на основе добровольности, общих творческих интересов и индивидуального членства участников в целях удовлетворения многообразных духовных запросов и интересов учащихся в свободное время. (Положение о любительском объединении, клубе по интересам // Информатика и образование. 1987. №2.)

Деятельность объединения осуществляется по следующим основным видам: познавательная, пропагандистская, учебная, поисково-исследовательская, художественно-творческая, развлекательная. При наличии материально-технической базы школы можно организовать разнообразные формы работы клубов как по содержанию, так и по организации.

3.3. ШКОЛЬНЫЕ ОЛИМПИАДЫ ПО ИНФОРМАТИКЕ

Олимпиады возникли в Древней Греции как состязания в ловкости, силе, красоте. Первая олимпиада состоялась в 776 г. до н.э. Различного рода состязания проводились не только в спорте. Хорошо известна любовь к состязаниям в решении задач как на Руси, так и во многих других странах мира.

В России конкурсы по решению задач начали проводиться с 1886 г. Первая олимпиада по программированию была проведена в 1980 г. С тех пор олимпиады по информатике проводятся ежегодно. Предметные олимпиады проводятся в несколько туров: школьные, районные, городские, республиканские, общероссийские. Завершающий тур – международный.

Одной из целей проведения олимпиад является развитие интереса к предмету, привлечение учащихся к занятиям в кружках. Другая цель – выявление у учащихся склонностей к данному предмету и развитие их способностей. Несмотря на то, что олимпиады по информатике аналогичны по своей сути всем другим предметным

олимпиадам, тем не менее организация и проведение их с технической точки зрения значительно сложнее. Во-первых, требуется достаточно большое количество персональных компьютеров; во-вторых, необходимо соответствующее программное обеспечение; в-третьих, вся техника и программное обеспечение должны бесперебойно функционировать; и последнее, составление задач, проведение олимпиады и проверка работ участников намного сложнее и могут быть осуществлены только квалифицированными специалистами в области информатики и вычислительной техники.

Проведение олимпиад предполагает соответствующую подготовку учащихся. Поэтому в каждой школе должны работать предметные кружки. Также систематически должна проводиться индивидуальная работа с наиболее сильными, одаренными учащимися. Для проведения олимпиад создаются оргкомитет и жюри. Они обеспечивают всю подготовительную работу, предшествующую непосредственно проведению олимпиады, обеспечивают подбор заданий, проверку работ участников, присуждают призы. При подборе заданий целесообразно придерживаться такого принципа, при котором часть заданий должна быть посильна для большинства участников. Такие задания вселяют уверенность в свои силы, не отпугивают учащихся от занятий информатикой.

На сегодняшний день школьные олимпиады проводятся по программированию и информационным технологиям. При подборе заданий можно использовать материалы, опубликованные в журнале «Информатика и образование», газете «Информатика» и других изданиях.

После проведения школьной олимпиады по информатике следует провести вечер по подведению итогов.

Ниже приведены примерные задания к школьной олимпиаде и алгоритмы решения.

Задание 1. Две точки на плоскости, не лежащие на координатных осях, заданы своими координатами: $A(x_1, y_1)$ и $B(x_2, y_2)$. Проверить, лежат ли эти точки в координатной четверти.

Идея решения: последовательно проверить, совпадают ли знаки первых координат точек, затем – знаки вторых координат точек; для проверки воспользоваться соображением, что знаки двух чисел совпадают тогда и только тогда, когда их произведение положительно.

```

Алг квадрант (вещ x1,y1,x2,y2, лит ответ)
  Арг x1,y1,x2,y2
  Рез ответ
Нач
  Ответ:=«нет»
  Если x1*x2>0
    То если y1*y2>0
      То ответ:=«да»
    Все
  Все
Кон

```

Задание 2. Три точки на плоскости заданы своими координатами: $A(x_1, y_1)$, $B(x_2, y_2)$, $C(x_3, y_3)$. Определить, лежат ли они на одной прямой.

Идея решения: вычислить длину трех отрезков, используя вспомогательный алгоритм, и применить неравенство треугольника.

```

Алг Прямая (вещ x1,y1,x2,y2,x3,y3, лит ответ)
  Арг x1,y1,x2,y2,x3,y3
  Рез ответ
Нач вещ d1,d2,d3
  Ответ:=«нет»
  Отрезок (x1,y1,x2,y2,d1)
  Отрезок (x2,y2,x3,y3,d2)
  Отрезок (x1,y1,x3,y3,d3)
  Если d1=d2+d3 или d2=d1+d3 или d3=d1+d2
    То ответ:=«да»
  Все
Кон

```

```

Алг Отрезок (вещ a1,b1,a2,b2, d)
  Арг a1,b1,a2,b2
  Рез d
Нач
  d:=sqr ((a2-a1)^2+(b2-b1)^2)
кон

```

Задание 3. Выяснить, в какой координатной четверти расположен треугольник, образованный прямой, заданный уравнением $Y=A*X+B$, и осями координат.

Возможный алгоритм:

0. Начать.
1. Ввод А и В.
2. Проверить условие ($A=0$ или $B=0$), если верно, то идти к пункту 9.
3. Проверить условие $B>0$, если верно, то идти к пункту 6.
4. Проверить условие $A>0$, если верно, вывести; 2-я четверть, идти к пункту 10.
5. Вывести 3-я четверть, идти к пункту 10.
6. Проверить условие $A>0$, если верно, то вывести 2-я четверть, идти к пункту 10.
7. Вывести 1-я четверть.
8. Идти к пункту 10.
9. Вывести – треугольника нет.
10. Закончить.

Задание 4. Можно ли разменять 25 рублей десятью монетами достоинством в 1, 3 или 5 рублей?

Идея решения: попытаемся набрать нужную сумму прямым перебором монет (три вложенных цикла).

Алг Размен (лит ответ)

Арг

Рез ответ

Нач цел i, j, k

$I:=0$

Ответ:=«нет»

Пока $i \leq 10$ и ответ=«нет»

Нц $j:=0$

Пока $j \leq 10$ и ответ=«нет»

Нц $k:=0$

Пока $k < 10$ и ответ=«нет»

Нц если $i+3*j+5*k=25$ и $ш+о+л=10$

То ответ:=«да»

Все

$K:=k+1$

Кц

$J:=j+1$

Кц

$I:=i+1$

Кц

Кон

Задание 5. Существует ли четырехзначное натуральное число, куб суммы цифр которого равен самому? Разработать алгоритм построения наименьшего такого числа.

Идея решения: четыре вспомогательных переменных принимают значения цифр так, чтобы из них формировались возрастающие четырехзначные числа. Для каждого из них проверяется выполнение условия задачи. Алгоритм заканчивает работу либо при окончании полного перебора, либо при получении положительного ответа.

```
Алг Числа (лит ответ, цел p)
  Арг
  Рез ответ, p
  Нач цел j1,j2,j3,j4
    ответ:=»не существует«
    j1:=1
    пока j1<=9 и ответ=«не существует»
      нц j2:=0
        пока j2<=9 и ответ=«не существует»
          нц j3:=0
            пока j3<=9 и ответ=«не существует»
              нц j4:=0
                пока j4<=9 и ответ=«не существует»
                  нц p:=j1*1000+j2*100+j3*10+j4
                    если P=(j1+j2+j3+j4)^3
                      то ответ:=«существует»
                    все
                  j4:=j4+1
                кц
              j3:=j3+1
            кц
          j2:=j2+1
        кц
      j1:=j1+1
    кц
  КОН
```

Возможен и другой подход к решению: перебирать числа, начиная с 10, пока их кубы будут оставаться меньше 10000, в их кубах выделять цифры и, найдя их сумму, проверять условие задачи.

Задание 6. Найти все трехзначные числа, равные среднему арифметическому чисел, полученных из каждого такого числа всеми перестановками его цифр.

Первый вариант решения. Решение получается простым и коротким, если провести некоторый анализ задачи. Пусть $a*100 + b*10 + c$ – искомое число, abc – его десятичная запись. Тогда $abc, acb, bca, bac, cab, cba$ – десятичные записи всех чисел, получающихся перестановками цифр искомого. Сложив их все, получим $(2*100 + 2*10 + 2)*(a+b+c)$, что по условию равно $6*(a*100 + b*10 + c)$, откуда $7a = 3b + 4*c$ и $a = (3*b + 4*c)/7$, а поскольку b и c могут принимать значения 0, 1, ..9, то остается только проследить, чтобы a получалось натуральным числом.

```

Алг Задача (нат n)
  рез n
  нач нат a,b,c
  для b от 0 до 9
    нц для c от 0 до 9
      нц a:=(3*b+4*c)/7
        если целчасть(a)=a и a>0
          то n:=100*a+b*10+c
        все
      кц
    кц
  кон

```

Второй вариант решения получается прямым перебором с проверкой. Цикл для нахождения всех перестановок из трех цифр заводить все же не имеет смысла.

```

Алг Задача (нат n)
  рез n
  нач нат a,b,c,m1,m2,m3,m4,m5,m6,s
  n:=0
  для a от 1 до 9
    нц для b от 0 до 9
      нц для c от 0 до 9
        нц m1:=100*a+b*10+c

```

```

        m2:=100*a+c*10+b
        m3:=100*b+c*10+a
        m4:=100*b+a*10+c
        m5:=100*c+b*10+a
        m6:=100*c+a*10+b
        s:=m1+m2+m3+m4+m5+m6
        если s=6*m1
            то n:=m1
        все
    кц
кц
кц
кОН

```

Задание 7. Из двух упорядоченных (в порядке возрастания) линейных таблиц размерности $1:n$ сформировать одну линейную таблицу, также упорядоченную по возрастанию, используя упорядоченность исходных таблиц.

Идея решения: попарно сравниваем элементы двух массивов, начиная с наименьших, заполняем очередной элемент создаваемого массива меньшим из этих элементов; оставшийся элемент одного из исходных массивов сравниваем со следующим элементом второго исходного массива. В случае, если один из массивов окажется исчерпанным, приписать к создаваемому массиву оставшиеся элементы второго (уже упорядоченного по условию) массива.

Предложенный алгоритм не является оптимальным, возможны лучшие решения.

```

Алг Слияние (цел n, вещ таб a[1..n],b[1..n],c[1..2*n])
    арг n,a,b
    рез с
    нач цел i,j,k
        i:=1;j:=1;k:=0
        пока i<=n и j<=n
            нц k:=k+1
                если a[i]<b[j]
                    то с[k]:=a[i]
                    i:=i+1
                иначе с[k]:=b[j]
                    j:=j+1
            кц
    кц

```

```

    все
    кц
если  $i > n$ 
    то  $i := j$ 
        пока  $i \leq n$ 
            нц  $k := k + 1$ 
                 $c[k] := b[i]$ 
            кц
        иначе  $j := i$ 
            пока  $j \leq n$ 
                нц  $k := k + 1$ 
                     $c[k] := a[j]$ 
                     $j := j + 1$ 
                кц
        все
кц

```

кц

Задание 8. Произвольный выпуклый многоугольник на плоскости задан координатами своих вершин (вершины указаны по часовой стрелке). Определить количество прямых углов многоугольника.

Идея решения: вычислить для каждой вершины координаты вектора, выходящего из нее к следующей (по часовой стрелке) вершине, и координаты вектора, выходящего из следующей вершины, найти их скалярное произведение и если оно равно нулю, увеличить количество найденных прямых углов.

Чтобы организовать перебор вершин K -угольника в цикле, добавим координаты вершин $K+1$ и $K+2$, совпадающие с координатами вершин 1 и 2.

Алг Многоугольник (цел k , прям-углы, вещ таб $x[1:k+2]$, $y[1:k+2]$)

```

    арг  $k, x, y$ 
    рез прям-углы
    нач цел  $j$ , вещ  $sx1, sy1, sx2, sy2$ , скалпр
        прям-углы := 0
        для  $j$  от 1 до  $k$ 
            нц  $sx1 := x[j+1] - x[j]$ 
                 $sy1 := y[j+1] - y[j]$ 
                 $sx2 := x[j+2] - x[j+1]$ 
                 $sy2 := y[j+2] - y[j+1]$ 
            кц
        кц

```

```

    скалпр:=cx1*cx2+cy1*cy2
    если скалпр=0
        то прям-углы:=прям-углы+1
    все
кц
кон

```

Задание 9. Произвольный выпуклый многоугольник на плоскости задан координатами своих вершин. Найти самую длинную диагональ данного многоугольника, если порядок вершин не указан, но известно, что существует только один выпуклый многоугольник с указанным набором вершин.

Возможный алгоритм:

1. Ввести N вершин многоугольника.
 2. Ввести массивы координат X(1:N), Y(1:N).
 3. Обнулить массив L(1:N,1:nn).
 4. I:=0.
 5. I:=i+1.
 6. J:=1.
 7. J:=J+1.
 8. Переход на подпрограмму ДИАГОНАЛЬ (X(I),Y(I),X(J),Y(J)) (она выдает значение ключа P=2, если X(I),Y(I), X(J),Y(J) является диагональю, и P=1 – в противном случае).
 9. Проверить условие P=1, если верно, то идти к пункту 6.
 10. $L(I,J)=\text{SQR}((X(I)-X(J))^2+Y(I)-Y(J))^2$.
 11. Проверить условие J<N, если верно, то идти к пункту 6.
 12. Проверить условие I<N-1, если верно, то идти к пункту 4.
 13. Найти элемент LMAX в массиве L.
 14. Вывести LMAX.
 15. Вывести (X(I),Y(I),X(J),Y(J)).
- Подпрограмма ДИАГОНАЛЬ (X1,Y1,X2,Y2,P).
1. P1:=0, P2:=0.
 2. K:=0.
 - 2.1. Проверить условие X1=X2, если верно, то идти к пункту 5.
 3. $C(X1,Y1,X2,Y2):=(Y2*X1-X2*Y1)/(X1-X2)/$
 4. $A(X1,Y1,X2,Y2):=(Y1-Y2)/(X1-X2)$ (A и C – процедуры вычисления коэффициентов прямой $y=a*x+c$, проходящей через точки (x1,y1) и (x2,y2)).
 5. K:=K+1.

6. Проверить условие $X(K)=X1$ В $Y(K)=Y1$) или $(X(K)=X2$ и $Y(K)=Y2$, если верно, то идти к пункту 5.

6.1 . Проверить условие $X1 <> X2$, если верно, то идти к пункту 7.

6.2 . Проверить условие $X(K)=X1$, если верно, то ($P1:=1$, идти к пункту 9).

6.3 . $P2:=1$, идти к пункту 9.

7. Проверить условие $Y(K)>A*X(K)+C$, если верно, то ($P1:=1$, идти к пункту 9).

8. $P2:=1$.

9. $P:=P1+P2$.

10. Проверить условие $P=2$, если верно, то идти к пункту 12.

11. Проверить условие $K < N$, если верно, то идти к пункту 5.

12. Возврат в основную программу.

Конец алгоритма.

Задание 10. Составить алгоритм, получающий фразу и подсчитывающий число слов в этой фразе. Слова во фразе разделяются одним или несколькими пробелами.

Алг Число слов

нач цел i, n , лит s , пред

вывод нс «Введите фразу»; ввод s

$n:=0$; пред:=« »

нц для i от 1 до $\text{длин}(s)$

если пред=« » и $s [i] <> \text{« »}$ то $n:=n+1$

все

пред:= $s [i]$

кц

вывод нс «Число слов во фразе:», s, n

кон

В основном цикле алгоритма сравниваем каждый символ фразы с предыдущим. Если символ отличен от пробела, а предыдущий символ равен пробелу, то этот символ начинает новое слово и мы увеличиваем счетчик n на 1. Очередной символ запоминается в переменной $пред$. Чтобы не пропустить при подсчете первое слово фразы, перед началом цикла этой переменной присваивается пробел.

Задание 11. Составить алгоритм вычисления функции $F(X)$ на отрезке $[a, b]$ методом деления пополам с точностью $\epsilon=0,01$. Функ-

ция непрерывна и монотонна на отрезке и принимает значение разных знаков на его концах. (Функция задана в виде $y=x^2-2$).

Алг Вычисление корня

нач вещ A, B, e, C

у:=0.01

ввод A, ввод B

нц пока $B-A > 2 * e$

С:=(A+B)/2

если $F(A) * F(C) \leq 0$

то B:=C

иначе A:=c

все

кц

С:=(A+B)/2

вывод «корень», C

кон

Алг вещ F(вещ X)

нач

знач:= X^2-2

кон

3.4. ЛЕКТОРИЙ

В лекторий объединяются ребята, интересующиеся лекторской работой и хорошо знающие информатику. В их задачу входит разработка плана лектория, подготовка лекций на различные темы с привлечением компьютера и других технических средств.

Члены лекторской группы начинают готовить сообщения с подбора и изучения литературы, составления плана лекций, затем записывают текст, заучивают его, репетируют свое выступление перед учителем, членами кружка. Лекторская группа выступает на занятиях кружка, конференциях и других внутришкольных мероприятиях. В качестве лекторов можно пригласить преподавателей вузов, специалистов.

Примерный текст лекции на тему

«Признаки делимости в двоичной системе счисления»

Сумма, разность и произведение двух целых чисел – всегда целые числа. Этот факт принято называть замкнутостью множества

целых чисел по отношению к действиям сложения, вычитания и умножения.

По отношению к действию деления множество всех целых чисел не является замкнутым: частное от деления одного целого числа на другое может и не быть целым. Поэтому при изучении обстоятельств, связанных с делением целых чисел, одним из первых встает вопрос о выполнимости этого действия на множестве целых чисел, т.е. о делимости этих чисел. Под «числом» далее будем понимать целое число, если не оговорено противное.

Для выяснения делимости числа a на число b имеется довольно много разнообразных способов. Один из них состоит в непосредственном делении числа a на число b . Однако такое деление часто оказывается долгим и утомительным занятием и, естественно, появляется желание установить истинность интересующей нас делимости, не производя фактического деления.

Способы, которые позволяют установить факт делимости чисел, не производя грубого деления a на число b , называются признаками делимости. По существу, эти способы должны установить признак делимости более коротким, экономным путем.

Некоторые признаки делимости целых чисел общеизвестны, например на 2, 3, 5, 9 и т. п.

Число делится на 2, если на 2 делится число единиц его последнего разряда.

Число делится на 3, если сумма его цифр делится на 3.

Число делится на 5, если его последняя цифра есть 5 или 0.

Число делится на 9, если сумма его цифр делится на 9.

Эти признаки связаны с представлением чисел в десятичной системе счисления. Эти признаки не применимы, если пользоваться системой счисления с каким-либо другим основанием, отличным от 10. Например, 86 в восьмеричной системе записывается в виде $(126)_8$ (так как $86 = 1 \cdot 8^2 + 2 \cdot 8 + 6$). Сумма цифр равна 9, но 86 не делится ни на 9, ни на 3.

Однако для каждой позиционной системы счисления можно сформулировать свои признаки делимости на то или иное число. Так, в двенадцатеричной системе можно сформулировать следующие признаки делимости:

а) число $A = (a_n a_{n-1} \dots a_1 a_0)_{12}$ делится на 8, если на 8 делится число $(a_1 a_0)_{12}$, образованное его двумя последними цифрами;

б) число $A=(a_n a_{n-1} \dots a_1 a_0)_{12}$ делится на 9, если на 9 делится число $(a_1 a_0)_{12}$, образованное его двумя последними цифрами;

в) число $A=(a_n a_{n-1} \dots a_1 a_0)_{12}$ делится на 11, если на 11 делится сумма его цифр, т. е. число $a_n + a_{n-1} + \dots + a_1 + a_0$.

Но нас в данном случае будут интересовать не признаки делимости в каждой системе счисления, а признаки делимости, существующие в десятичной системе счисления.

Системы счисления с различными основаниями инвариантны, значит, можно полагать, что признаки делимости в системе счисления с одним основанием существуют в системах счисления с другим основанием, но примут скорее всего другой вид. Выведем признаки делимости на 3, 5, 7 в двоичной системе счисления. Признаки делимости на 2, 4 (на 2^n) общеизвестны.

1. Двоичное число делится на два, если оно оканчивается на 0.

2. Двоичное число делится на 4, если оно оканчивается двумя нулями.

3. Признак делимости на 3 в двоичной системе счисления.

Теорема 1. *Запись числа в двоичной системе счисления разделим на группы по 2 цифры в каждой, начиная справа. Складываем образованные группы. Если получившаяся сумма делится на 3, то исходное двоичное число делится на 3.*

Доказательство. Запишем двоичное число в виде суммы ряда, сгруппируем по два справа и вынесем в образовавшихся группах общий множитель:

$$a_n * 2^n + a_{n-1} * 2^{n-1} + \dots + a_7 * 2^7 + a_6 * 2^6 + a_5 * 2^5 + a_4 * 2^4 + a_3 * 2^3 + a_2 * 2^2 + a_1 * 2^1 + a_0 * 2^0 = \dots + (a_7 * 2^7 + a_6 * 2^6) + (a_5 * 2^5 + a_4 * 2^4) + (a_3 * 2^3 + a_2 * 2^2) + (a_1 * 2^1 + a_0 * 2^0) = \dots + 2^6 * (a_7 * 2 + a_6) + 2^4 * (a_5 * 2 + a_4) + 2^2 * (a_3 * 2 + a_2) + 2^0 * (a_1 * 2 + a_0).$$

Общий вид слагаемых $2^{2^k} * (a_{k+1} * 2 + a_k)$. Докажем, что множитель всегда равен $2^{2^k} = 3 * N + 1$, где N – натуральное число.

$$2^{2^k} = 3 * N + 1,$$

$$2^{2^k} - 1 = 3 * N,$$

$$(2^k - 1) * (2^k + 1) = 3 * N.$$

На 3 делится либо $2^k - 1$, либо $2^k + 1$, так как 2^k не делится на 3 (три последовательных натуральных числа). Запишем остатки при делении нашего числа на 3:

$$(a_1 * 2 + a_0) + (a_3 * 2 + a_2) + (a_5 * 2 + a_4) + \dots = (a_1 a_0)_2 + (a_3 a_2)_2 + (a_5 a_4)_2 + \dots$$

Слева – сумма образовавшихся групп. Если эта сумма делится на 3, то само двоичное число делится на 3. Теорема доказана.

Следствие 1. Алгоритм определения делимости на 3 двоичного числа.

- а) Запись числа в двоичной системе счисления разделить на группы по 2 цифры в каждой, начиная справа. Сложить образованные группы.
- б) Если количество цифр числа суммы больше 2, то перейти к пункту а.
- в) Если сумма равна 11_2 , то двоичное число делится на 3.

4. Признак делимости на 5 в двоичной системе счисления.

Теорема 2. Запись числа в двоичной системе счисления разделим на группы по 4 цифры в каждой, начиная справа. Складываем образованные группы. Если получившаяся сумма делится на 5, то исходное двоичное число делится на 5.

Доказательство. Аналогично доказательству теоремы 1, записав двоичное число в виде суммы ряда, сгруппировав справа на группы по 4 и вынеся в образовавшихся группах общий множитель, получим, что множитель имеет общий вид: 2^{4*k} . Докажем, что $2^{4*k} = 5*N + 1$, где N – натуральное число.

Для доказательства используем метод математической индукции:

$$2^4 = 16 = 5*3 + 1, \quad 2^8 = 256 = 5*51 + 1.$$

Допустим справедливость $2^{4*k} = 5*N + 1$, где k и N принадлежат множеству натуральных чисел. Докажем справедливость для $2^{4*(k+1)} = 5*N + 1$, где k и N принадлежат множеству натуральных чисел.

$$\begin{aligned} 2^{4*(k+1)} &= 2^{4*k+4} = 2^{4*k} * 2^4 = 2^{4*k} * 16 = 2^{4*k} * (15+1) = 15*2^{4*k} + 2^{4*k} = \\ &= 15*2^{4*k} + 5*N + 1 = \text{по допущению } 2^{4*k} = 5*N + 1 = 5*(3*2^{4*k} + N) + 1. \end{aligned}$$

$3*2^{4*k} + N$ принадлежит множеству натуральных чисел, так как k и N принадлежат множеству натуральных чисел.

Тогда остатки при делении числа на 5

$$\begin{aligned} &(2^3*a_3 + 2^2*a_2 + 2^1*a_1 + 2^0*a_0) + (2^3*a_7 + 2^2*a_6 + 2^1*a_5 + 2^0*a_4) + \dots \\ &= (a_3a_2a_1a_0)_2 + (a_7a_6a_5a_4)_2 + \dots \end{aligned}$$

Слева сумма образовавшихся групп. Если эта сумма делится на 5, то и само число делится на 5. Теорема доказана.

Следствие 2. Алгоритм определения делимости на 5 двоичного числа.

- а) Запись числа в двоичной системе счисления разделить на группы по 4 цифры в каждой, начиная справа. Сложить образованные группы.
- б) Если количество цифр суммы больше 4, то перейти к пункту а.
- в) Если сумма равна 1111_2 , 1010_2 или 101_2 , то исходное двоичное число делится на 5.

5. Признак делимости на 7 в двоичной системе счисления.

Теорема 3. *Запись числа в двоичной системе счисления разделить на группы по три цифры в каждой, начиная справа. Складываем образованные группы. Если получившаяся сумма делится на 7, то исходное двоичное число делится на 7.*

Данная теорема доказывается аналогично доказательствам теорем 1 и 2.

Следствие 3. *Алгоритм определения делимости на 7 двоичного числа.*

- а) Запись числа в двоичной системе счисления разделить на группы по 3 цифры, начиная справа. Сложить образованные группы.
- б) Если количество цифр суммы больше 3, то перейти к пункту а.
- в) Если сумма равна 111_2 , то исходное двоичное число делится на 7.

В дальнейшем используем следующее свойство: все четные числа в двоичной записи оканчиваются на 0.

Следствие 4. Признак делимости на 6.

В записи числа в двоичной системе счисления зачеркиваем справа ноль. Вновь образованное число разделим на группы по 2 цифры в каждой, начиная справа. Складываем образованные группы. Если получившаяся сумма делится на 3, то исходное двоичное число делится на 6.

Другая формулировка следствия 4.

В записи числа зачеркиваем справа 0. Если вновь образованное число делится на 3, то исходное двоичное число делится на 6.

Следствие 5. Признак делимости на 10.

В записи числа в двоичной системе счисления зачеркиваем справа ноль. Если вновь образованное число делится на 5, то исходное двоичное число делится на 10.

Следствие 6. Признак делимости на 12.

В записи числа зачеркиваем справа два нуля. Если вновь образованное число делится на 3, то исходное двоичное число делится на 12.

Следствие 7. Признак делимости на 14.

В записи числа в двоичной системе счисления зачеркиваем справа ноль. Вновь образованное число разделим на группы по 3 цифры в каждой, начиная справа. Складываем образованные группы. Если получившаяся сумма делится на 7, то исходное двоичное число делится на 14.

Другая формулировка следствия 7.

В записи числа зачеркиваем справа 0. Если образованное число делится на 7, то исходное двоичное число делится на 14.

Следствие 8. Признак делимости на 15.

Запись числа в двоичной системе счисления разделим на группы по 4 цифры в каждой, начиная справа. Складываем образованные группы. Если получившаяся сумма делится на 15, то исходное двоичное число делится на 15. (Доказывается аналогично теореме 2).

Алгоритм определения делимости на 15 двоичного числа.

- а) Запись числа в двоичной системе счисления разделить на группы по 4 цифры справа. Сложить образованные группы.
- б) Если количество цифр числа суммы больше 4, то перейти к пункту а.
- в) Если сумма равна 1111_2 , то исходное двоичное число делится на 15.

Ниже приведены программы на Turbo Pascal и Qbasic для экспериментальной проверки некоторых (на 3, 5, 7) приведенных выше признаков делимости чисел в двоичной системе счисления.

Программа 1.1	Программа 1.2
'ПРОВЕРКА ПРИЗНАКОВ ДЕЛИМОСТИ ЧИСЛА N НА M В ДВОИЧНОЙ СИСТЕМЕ СЧИСЛЕНИЯ DIM A(100): DIM C(100): DIM B(50, 50) PRINT «ПРОВЕРКА ПРИЗНАКА ДЕЛИМОСТИ ЧИСЛА N В ДВОИЧНОЙ СИСТЕМЕ СЧИСЛЕНИЯ» INPUT «ВВЕДИ N»; N	Program P2_2; {Проверка признаков делимости числа N на M в двоичной системе счисления} uses crt; var a:array[1..100] of integer; C:array[1..100] of integer; B: array [1..50,1..50] of integer; M,Nби,p,j,k,t,h,x,y,s,d,r,z: inte-

<pre> PRINT «НА ЧИСЛО M» INPUT «ВВЕДИ M»; M I = 0 'ПЕРЕВОД ДЕСЯТИЧНОГО ЧИСЛА В ДВОИЧНОЕ WHILE N > 0 A(I) = N MOD 2 P = N \ 2 N = P: I = I + 1 WEND PRINT «ДВОИЧНОЕ ПРЕДСТАВ- ЛЕНИЕ ЧИСЛА» FOR J = I - 1 TO 0 STEP -1 'ВЫВОД ДВОИЧНОГО ЧИСЛА PRINT A(J); NEXT J: PRINT 'ГРУППИРУЕМ ДВОИЧНОЕ ЧИСЛО В ГРУППЫ ПО K' PRINT «ГРУППИРОВАНИЕ ДВО- ИЧНОГО ЧИСЛА ПО K» INPUT «ВВЕДИ K»; K IF I MOD K <> 0 THEN I = (I \ K + 1) * K PRINT «ДВОИЧНОЕ ПРЕДСТАВ- ЛЕНИЕ ИСХОДНОГО ЧИСЛА С УЧЕТОМ ПОДГОТОВКИ» PRINT «ДЛЯ ГРУППИРОВАНИЯ» FOR H = I - 1 TO 0 STEP -1: PRINT A(H); : NEXT T = 0 FOR Y = 1 TO I \ K FOR X = 1 TO K B(X, Y) = A(T): T = T + 1 NEXT X, Y: PRINT 'ВЫВОД ОБРАЗОВАННЫХ ГРУПП PRINT «ВЫВОД ОБРАЗОВАН- НЫХ ГРУПП В ОБРАТНОМ ПО- РЯДКЕ» FOR Y = 1 TO I \ K FOR X = 1 TO K PRINT B(X, Y); NEXT X: PRINT </pre>	<pre> ger; BEGIN clrscr; {очистка экрана} WRITELN('Проверка призна- ка делимости числа N в дво- ичной системе счисления'); WRITELN('Введи N'); READ(n); WRITELN('на число M'); WRITELN('Введи M');READ(m); i:=0; {Перевод десятичного числа в двоичное} WHILE n > 0 DO BEGIN a[i]:= n MOD 2; p:= n DIV 2; n:=p;i:=i+1; END; WRITELN('Двоичное пред- ставление числа'); FOR j:= i - 1 DOWNT0 0 DO {Вывод двоичного числа} WRITE(a[j]); WRITELN; {Группируем двоичное число в группы по K} WRITELN('Группирование двоичного числа по K'); WRITELN('Введи K'); READ(k); IF i MOD k<>0 THEN i:=(i DIV k+1) * k; WRITELN('Двоичное пред- ставление исходного числа с учетом подготовки'); WRITELN('для группирова- ния'); FOR h:=i-1 DOWNT0 0 DO WRITE(a[h]); t:=0; FOR y:= 1 TO i DIV k DO FOR x:= 1 TO k DO BEGIN b[x, y]:=a[t]; t:=t+1; END; </pre>
--	--

<pre> NEXT Y 'СЛОЖЕНИЕ ОБРАЗОВАННЫХ ГРУПП S = 0 FOR X = 1 TO K FOR Y = 1 TO I \ K S = S + B(X, Y) NEXT Y C(X) = S IF C(X) >= 1 AND X <> K THEN S = S \ 2 IF C(X) MOD 2 = 0 AND X <> K THEN C(X) = 0 ELSE C(X) = 1 IF X = K THEN C(K) = S ' PRINT C(X); NEXT X S = C(K) WHILE S > 0 C(K) = S MOD 2: P = S \ 2 S = P: K = K + 1 WEND: PRINT 'ВЫВОД РЕЗУЛЬТАТА СЛОЖЕ- НИЯ ОБРАЗОВАННЫХ ГРУПП PRINT «СУММА ГРУПП» FOR J = K - 1 TO 1 STEP -1 PRINT C(J); : NEXT J 'ПЕРЕВОД ДВОИЧНОГО ЧИСЛА (РЕЗУЛЬТАТА СЛОЖЕНИЯ ГРУПП) В ДЕСЯТИЧНОЕ ЧИСЛО FOR J = 1 TO K - 1 Z = C(J) * 2 ^ (J - 1) D = D + Z NEXT J: PRINT PRINT «ДЕСЯТИЧНОЕ ПРЕД- СТАВЛЕНИЕ РЕЗУЛЬТАТА СЛОЖЕНИЯ ГРУПП D=«; D 'ПРОВЕРКА ДЕЛИМОСТИ IF D MOD M = 0 THEN PRINT «ПРИЗНАК ДЕЛИМОСТИ ВЫ- ПОЛНЯЕТСЯ» </pre>	<pre> WRITELN; {Вывод образованных групп} WRITELN('Вывод образо- ванных групп в обратном порядке'); FOR y:=1 TO i DIV k DO BE- GIN WRITELN; FOR x:=1 TO k DO WRITE(b[x, y]);END; WRITELN; {Сложение образованных групп} s:= 0; FOR x:= 1 TO k DO BEGIN FOR y:= 1 TO i DIV k DO s:= s + b[x, y]; c[x]:= s; IF (c[x] >= 1) AND (x <> k) THEN s:= s DIV 2; IF(c[x] MOD 2=0) AND (x<>k) THEN c[x]:=0 ELSE c[x]:= 1; IF x = k THEN c[k]:= s; END; s:=c[k]; WHILE s > 0 DO BEGIN c[k]:=s MOD 2; p:= s DIV 2; s:= p; k:=k + 1;END; WRITELN; {Вывод результата сложения образованных групп} WRITELN('Сумма групп'); FOR j:=k - 1 DOWNT0 1 DO WRITE(C[J]); {Перевод двоичного числа (результата сложения групп) в десятичное} FOR j:=1 TO k - 1 DO BEGIN z:=c[j]; FOR r:=1 TO j-1 DO z:=z*2; d:=d + z; END; WRITELN; WRITELN('Десятичное пред- ставление результата сложе- ния групп d=',d); </pre>
---	--

	<pre> {Проверка делимости} IF d MOD m = 0 THEN WRITELN('Признак делимости выполняется'); Readkey; END.</pre>
--	--

3.5. ПОДГОТОВКА И ПРОВЕДЕНИЕ КОНФЕРЕНЦИЙ

Подготовка к конференции начинается с создания комиссии (инициативной группы учащихся) и выбора руководителя, которые:

- определяют тему и дату проведения;
- уточняют цели, задачи, проблемы выносимые на конференцию;
- готовят план проведения;
- составляют и оформляют тексты объявлений;
- намечают вопросы, которые будут предметом обсуждения, и своевременно доводят их до сведения участников;
- подбирают литературу;
- оформляют книжную выставку;
- разрабатывают тематику докладов;
- распределяют докладчиков;
- разрабатывают и оформляют плакаты, эпитаффы, диаграммы к конференции;
- подготавливают технические средства (диа- и мультипроекторы, компьютеры);

Подготовка к докладам по информатике имеет свою специфику, в частности, доклад готовят 2-3 человека. Один из них готовит текст доклада, второй – программное сопровождение, третий подбирает иллюстрации, чертежи. Все вместе подбирают литературу для выставки. Руководитель консультирует работу при подготовке к докладу, может рекомендовать другие книги или просмотр тех или иных интернет-сайтов, добавить новые факты, соответствующие тематике конференции, и др.

Порядок проведения конференции может быть таким:

1. Открытие конференции. Вступительное слово руководителя с разъяснением, почему выбрана именно эта тема, ее значение и основная идея.

2. Выступления учащихся с докладами, демонстрации компьютерных экспериментов.

3. Обсуждение докладов.

4. Подведение итогов. Руководитель дает краткий анализ и оценку докладов и демонстраций. Даются рекомендации для дальнейшей работы по теме. Обсуждается тема и дата проведения следующей конференции. Выбор темы конференции определяется интересом и желанием учащихся.

Можно предложить ряд тематических конференций, связанных с интернетом:

- учебные сайты в интернете;
- новинки по программным и информационным технологиям в интернете;
- техническое обеспечение компьютеров.

Пример научно-практической конференции «**Число π (пи)**».

Цель конференции – ознакомление с различными методами вычисления значения числа π .

Рекомендуемая литература:

1. *Лютикас В.С.* Школьнику о теории вероятности. М.: Просвещение, 1976.

2. *Новожилов Б.В.* Метод Монте-Карло. М.: Знание, 1966.

3. *Соболь Н.М.* Метод Монте-Карло: Популярные лекции по математике. Вып. 46. М.: Наука, 1968.

4. *Фихтенгольц Г.М.* Курс дифференциального и интегрального исчисления. Т.2. М.: Наука, 1969. С. 367-368.

5. *Основы информатики и вычислительной техники: Пробный учебник для средних учебных заведений/ А.Г. Кушниренко, Г.В. Лебедев, Р.Я. Сворень.* М.: Просвещение, 1990.

Вопросы для обсуждения:

1. Каковы достоинства каждого метода?
2. Насколько сложна реализация рассматриваемого метода?
3. Каковы возможности используемого численного метода?
4. Чем определяется точность данного метода?

Ниже приведены четыре метода вычисления значения числа π .

Метод прямоугольников

Суммирование площадей прямоугольников, вписанных в полу-
круг.

Пусть $A(a;0)$, $B(b;0)$. Опишем на AB полуокружность как на диаметре.

Разделим отрезок AB на n равных частей точками x_1, x_2, \dots, x_n и восстановим из них перпендикуляры до пересечения с полуокружностью. Длина каждого такого перпендикуляра – это значение функции $f(x) = \sqrt{1-x^2}$.

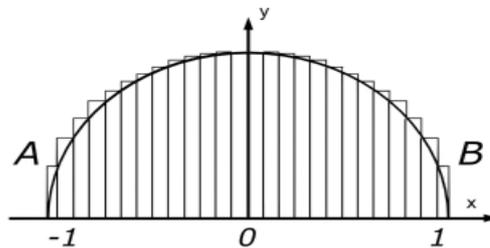


Рис. 21

Из рис.21 видно, что площадь S полукруга можно вычислить по формуле

$$S = ((b-a)/n) * (f(x_1) + f(x_2) + \dots + f(x_n)).$$

В нашем случае $b=1$, $a=-1$. Тогда $\pi = 2 * S$.

Значения π будут тем точнее, чем больше точек деления будет на отрезке AB . Облегчить однообразную вычислительную работу поможет компьютер. Ниже приведены программы вычислений, написанные на языке Basic и Turbo Pascal 7.0.

Программа 1.1	Программа 1.2
<pre> 10 REM ВЫЧИСЛЕНИЕ ПИ 20 REM МЕТОД ПРЯМОУГОЛЬ- НИКОВ 30 INPUT N 40 DX=1/X 50 FOR I=0 TO N-1 60 F=SQR(1-X^2) 70 X=X+DX 80 A=A+F 90 NEXT I </pre>	<pre> Program P1_2; {Вычисление Пи. Метод прямоугольников} uses crt; var n,i: Integer; dx,x,a,f,p: Real; begin clrscr; {очистка экрана} write('Кол-во: N='); readln(n); </pre>

<pre> 100 P=4*DX*A 110 PRINT «ЗНАЧЕНИЕ ПИ РАВ- НО»;P 120 STOP </pre>	<pre> dx := 1/n; x := 0; a := 0; for i := 0 to n-1 do begin f := sqrt(1-sqr(x)); x := x+dx; a := a+f; end; p := 4*dx*a; writeln('Значение пи рав- но',p:3:10); readkey; end. </pre>
--	---

Метод Монте-Карло

Это фактически метод статистических испытаний. Свое экзотическое название он получил от города Монте-Карло в княжестве Монако, знаменитого своими игорными домами. Дело в том, что метод требует применения случайных чисел, а одним из простейших приборов, генерирующих случайные числа, может служить рулетка. Впрочем, можно получить случайные числа и при помощи дождя.

Для опыта приготовим кусок картона, нарисуем на нем квадрат и впишем в квадрат четверть круга. Если такой чертеж некоторое время подержать под дождем, то на его поверхности останутся следы капель. Подсчитаем число следов внутри квадрата и внутри четверти круга. Очевидно, что их отношение будет приблизительно равно отношению площадей этих фигур, так как попадание капель в различные места чертежа равновероятно. Пусть N_{kr} – число капель в кругу, N_{kv} – число капель в квадрате, тогда

$$\pi = 4 * N_{kr} / N_{kv}. \quad (1)$$

Дождь можно заменить таблицей случайных чисел, которая составляется с помощью компьютера по специальной программе (можно воспользоваться одной из многочисленных публикаций таких таблиц, например, в брошюрах 2 или 3). Каждому следу капли поставим в соответствие два случайных числа, характеризующих его положение вдоль осей Ox и Oy (см. рис.22).

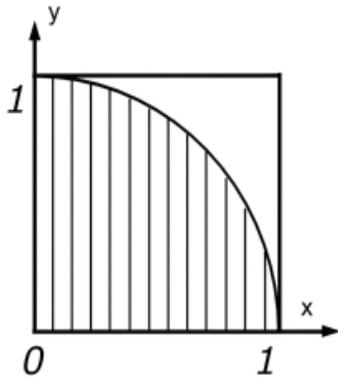


Рис. 22

Случайные числа можно выбрать из таблицы в любом порядке, например, подряд. Пусть первое четырехзначное число в таблице – 3265. Из него можно «приготовить» пару чисел, каждое из которых больше нуля и меньше единицы: $x=0.32$, $y=0.65$. Эти числа будем считать координатами капли, т.е. капля как будто попала в точку $(0.32; 0.65)$.

Аналогично поступаем и со всеми выбранными случайными числами. Если окажется, что для точки $(x_i; y_i)$ выполняется неравенство $x_i^2 + y_i^2 > 1$, значит, она лежит вне круга. Если $x_i^2 + y_i^2 \leq 1$, то точка лежит внутри круга.

Для подсчета значения π снова воспользуемся формулой (1). Ошибка вычислений по этому методу, как правило, пропорциональна $\sqrt{D/N}$, где D – некоторая постоянная, а N – число испытаний. В нашем случае $N=N_{kv}$. Из этой формулы видно: для того, чтобы уменьшить ошибку в 10 раз, нужно увеличить N в 100 раз. Ясно, что применение метода Монте-Карло стало возможным только благодаря компьютерам. Программа 2.1 и программа 2.2 реализуют описанный метод.

Программа 2.1	Программа 2.2
<pre> 10 REM ВЫЧИСЛЕНИЕ ПИ 20 REM МЕТОД МОНТЕ-КАРЛО 30 INPUT N 40 M=0 50 FOR I=1 TO N 60 T=INT(RND(1)*10000) 70 X=INT(T/100) 80 Y=T-X*100 90 IF X^2+Y^2<10000 THEN M=M+1 100 NEXT I 110 P=4*M/N 120 PRINT «ЗНАЧЕНИЕ ПИ РАВНО»;P 130 STOP </pre>	<pre> Program P2_2; { Вычисление числа Пи. Метод Монте-Карло} uses crt; var n,i,m: Longint; x,y,p: Real; begin clrscr; {очистка экрана} write('Кол-во N='); readln(n); m := 0; for i := 0 to n do begin x:=Random(10000)/10000; y:= Random(10000)/10000; if sqr(x)+sqr(y) <= 1 then m:=m+1; </pre>

```

end;
p := 4*m/n;
writeln('Значение пи равно',p:3:9);
readkey;
end.

```

Метод «падающей иглки»

Возьмем обыкновенную швейную иглку и лист бумаги. На листе проведем несколько параллельных прямых так, чтобы расстояния между ними были равны и превышали длину иглки.

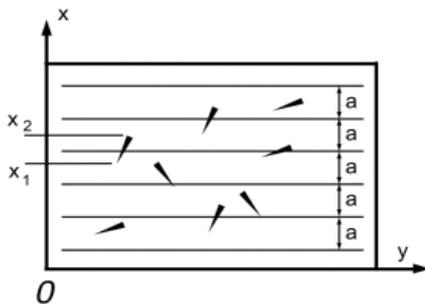


Рис. 23

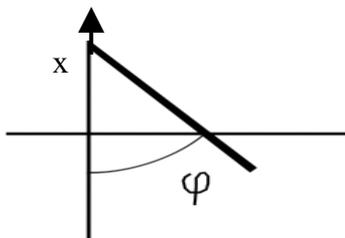


Рис. 24

Чертеж должен быть достаточно большим, чтобы случайно брошенная игла не упала за его пределы. Введем обозначения: a – расстояние между прямыми линиями, L – длина иглки. Положение случайным образом брошенной на чертеж иглы (см. рис. 23) определяется расстоянием x от ее середины до ближайшей прямой и углом, который игла образует с перпендикуляром, опущенным из середины иглы на ближайшую прямую (см. рис. 24).

Ясно, что $0 \leq x \leq \frac{L}{2}$,

$$-\frac{\pi}{2} \leq \varphi \leq \frac{\pi}{2}.$$

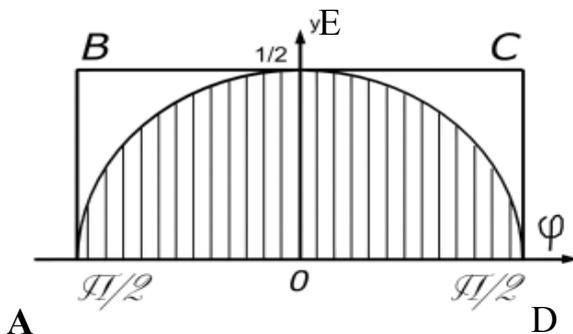


Рис. 25

На рис. 25 изобразим графически функцию $y = 0.5 * L * \cos(\varphi)$. Все возможные расположения иглы характеризуются точками с координатами $(\varphi; y)$, расположенными на участке ABCD. Заштрихованный участок AED – это точки, которые соответствуют

случаю пересечения иглы с прямыми линиями. Вероятность события – «игла пересекла прямую» – вычисляется по формуле

$$P(A) = \frac{S_{AED}}{S_{ABCD}}, \text{ где } S_{AED} = \int_{-\frac{\pi}{2}}^{\frac{\pi}{2}} 0.5 \cos(\varphi) d\varphi = 1$$

$$S = a \times \pi / 2, \text{ т. е. } P(A) = 2 \times L / a \times \pi.$$

Вероятность $P(A)$ можно приблизительно определить многократным бросанием иглы.

Пусть иглу бросали на чертеж N раз и k раз она упала, пересекая одну из прямых, тогда при достаточно большом S имеем $P(A) = k/S$.

$$\text{Отсюда } \pi = 2 \times L \times S / (a \times k).$$

Построение модели. В описанном выше методе используется тригонометрическая функция \cos . При вычислении тригонометрических функций на Бейсике углы из градусной меры приходится переводить в радианную, где используется значение π . Поэтому для построения нашей модели мы используем проекцию иглы (рис.21). Координаты проекции иглы на ось OY не влияют на наши расчеты. Координату $X1$ зададим датчиком случайных чисел. Координата $X2$ может изменяться в промежутке $X1 \leq X2 \leq X1 + L$, L – длина иглы. Условие пересечения иглы прямой $X1 \leq A \leq X2$, где A – координата x проекции прямой на OX . Для конкретности возьмем $A=2$ (расстояние между прямыми), $L=1$ (длина иглы), количество линий на чертеже 20. Описанная модель реализована в программах 3.1 и 3.2.

Программа 3.1	Программа 3.2
<pre> 10 REM ВЫЧИСЛЕНИЕ ПИ 20 REM МЕТОД ПАДАЮЩЕЙ ИГОЛКИ 30 L=1 : REM ДЛИНА ИГЛЫ 40 A=2 : REM РАССТОЯНИЕ МЕЖДУ ПРЯМЫМИ 50 INPUT N: REM N – КОЛ-ВО БРОСКОВ 55 K=0 : REM K – ПЕРЕСЕЧЕ- НИЙ 60 FOR I=1 TO N </pre>	<pre> Program P3_2; { Вычисление Пи. Метод падающей иголки } uses crt; var n,j,l,a,k: Longint; x1,y1,x2,y2,s,ro,p: Real; begin clrscr; {очистка экрана} l:= 1; {длина иглы} a:=2;{расстояние между пря- мыми} </pre>

<pre> 70 FOR A=2 TO 20 75 REM X1, X2 -КООРДИНАТЫ ПРОЕКЦИИ ИГЛЫ НА ОХ 80 X1=RND(1)*20 90 X2=X1+L*RND(1) 100 IF X1<=A AND X2<=A THEN K=K+1 110 NEXT A 120 NEXT I 130 P=2*L*N/A*K 140 PRINT «ЗНАЧЕНИЕ ПИ РАВ- НО»; P 150 STOP </pre>	<pre> write('Кол-во N='); readln(n); k:=0;{кол-во пересечений.} for j := 1 to n do begin x1:=Random(10000)/10000; y1:=Random(10000)/10000; x2:=Random(10000)/10000; y2:=Random(10000)/10000; s:=abs(y2-y1)/sqrt(sqr(y2- y1)+sqr(x2-x1)); ro:=(Random(10000)/10000)*2- 1; if abs(ro)<=s/2 then k:=k+1; end; p:=n/k; writeln('Значение Пи рав- но',p:3:9); readkey; end. </pre>
---	--

Замечание. Изложенный метод представляет собой вариацию метода статистических испытаний. Он интересен и с дидактической точки зрения, так как помогает совместить простой опыт с составлением довольно сложной математической модели.

Вычисление π с помощью ряда Тейлора

Обратимся к рассмотрению произвольной функции $f(x)$. Предположим, что для нее в точке x_0 существуют производные всех порядков до n -го включительно. Тогда для функции $f(x)$ можно записать ряд Тейлора:

$$f(x) = f(x_0) + \frac{f'(x_0)}{1!} (x-x_0) + \frac{f''(x_0)}{2!} (x-x_0)^2 + \frac{f'''(x_0)}{3!} (x-x_0)^3 + \dots$$

Пусть теперь $f(x) = \arctg(x)$ при $-1 \leq x \leq 1$ и $x_0 = 0$. Тогда

$$\arctg(x) = x - \frac{x^3}{3} + \frac{x^5}{5} - \frac{x^7}{7} + \frac{x^9}{9} - \dots$$

Если $x = 1$, $\arctg(1) = \pi/4$, значит,

$$\pi = 4 * \left(1 - \frac{1}{3} + \frac{1}{5} - \frac{1}{7} + \frac{1}{9} - \dots \right).$$

Вычисления с помощью этого ряда будут тем точнее, чем больше членов ряда будет задействовано. Реализация данного способа приведена в программах 4.1 и 4.2

Программа 4.1	Программа 4.2
<pre> 10 REM ВЫЧИСЛЕНИЕ ПИ 20 REM РАЗЛОЖЕНИЕМ В РЯД ТЕЙЛОРА 30 INPUT N 40 A=1 50 FOR I=1 TO N 60 D=1/(2*I+1) 70 F=(-1)^I*D 80 A=A+F 90 NEXT I 100 P=4*A 110 PRINT» ЗНАЧЕНИЕ ПИ РАВНО»; P 120 STOP </pre>	<pre> Program P4_2; { Вычисление Пи. Разложением в ряд Тейлора } uses crt; clrscr; {очистка экрана} var n,i: Integer; a,d,f,p:Real; begin write('Кол-во N='); readln(n); a := 1; for i := 1 to n do begin d := 1/(2*i+1); if odd(i) then f:=-1*d else f:=1*d; a:=a+f; end; p:=4*a; writeln('Значение Пи', p:3:10); readkey; end. </pre>

Разнообразие описанных способов позволяет использовать знания и умения, полученные на уроках математики и информатики, что очень полезно для общего развития школьников. Реализация выбранных способов требует практических навыков, умений составлять математические модели, алгоритмы и программы.

3.6. ПРОВЕДЕНИЕ КОНКУРСОВ ПО ИНФОРМАТИКЕ

Конкурсы играют очень важную роль во внеклассной работе, так как с их помощью вносится элемент соревнования, усиливается интерес к предмету. Они могут входить составной частью в массовые мероприятия и могут быть отдельным самостоятельным

мероприятием. Существуют разные виды конкурсов: викторины, эстафеты, аттракционы, КВН.

Приведем пример конкурса «*КВН в кабинете информатики*», состоящий из следующих этапов:

- игр «Счастливый случай», «Поле чудес», «Спортлото»;
- математический фокус;
- ребус;
- математический ребус.

Организация конкурса.

Класс делится на две команды. Приглашаются гости. Из числа гостей выбирают жюри. В состав жюри входит и учитель информатики. Обязанность жюри – фиксировать правильность выполнения задания, время, оценивать каждый этап и подвести итоги КВН.

Содержание конкурса.

1-й этап. Игра «Счастливый случай»

Команды заранее готовят по 10 вопросов друг другу. Составляют программу проверки правильности ответов. Заносят в компьютер. Команды отвечают на предложенные вопросы. Время, отведенное на выполнение задания, можно ограничить. Правильные ответы оцениваются некоторым количеством баллов. Ниже приведен пример программы, задающей 3 вопроса. Правильный ответ на каждый вопрос дает один балл. Переменная X хранит сумму набранных баллов.

Программа 1.1	Программа 1.2
<pre> 5 REM СЧАСТЛИВЫЙ СЛУЧАЙ 10 X=0 20 PRINT «КАК НАЗЫВАЕТСЯ ПОСЛЕДОВАТЕЛЬНОСТЬ КОМАНД» 30 PRINT «ИСПОЛНИТЕЛЮ ДЛЯ РЕШЕНИЯ ПОСТАВЛЕН- НОЙ ЗАДАЧИ ?» 40 INPUT A\$ 50 IF A\$=«АЛГОРИТМ» THEN X=X+1 60 PRINT « НАЗОВИТЕ ЭЛЕ- МЕНТАРНУЮ ЕДИНИЦУ ИН-</pre>	<pre> Program P1_2; uses crt; var a,b,c:string; x:word; begin clrscr; {очистка экрана} x:=0; writeln('Как называется после- довательность команд'); writeln('исполнителю для реше- ния поставленной задачи ?'); readln(a); if a='алгоритм' then x:=x+1;</pre>

<pre> ФОРМАЦИИ» 70 INPUT B\$ 80 IF B\$=«БИТ» THEN X=X+1 90 PRINT « ОСНОВНОЕ УСТРОЙСТВО ЭВМ, ВЫПОЛНЯЮЩЕЕ ЗАДАНИЕ» 100 PRINT « ПРОГРАММЫ ПРЕОБРАЗОВАНИЯ И ОСУЩЕСТВЛЯЮЩИЕ» 110 PRINT « УПРАВЛЕНИЕ ВСЕМ ВЫЧИСЛИТЕЛЬНЫМ ПРОЦЕССОМ В ЭВМ» 120 INPUT C\$ 130 IF C\$= «ПРОЦЕССОР» THEN X=X+1 140 PRINT « ИЗ ТРЕХ ВОЗМОЖНЫХ УГАДАЛИ «; X 150 END </pre>	<pre> writeln('Назовите элементарную единицу информации'); readln(b); if b='бит' then x:=x+1; writeln('Основное устройство ЭВМ, выполняющее заданные'); writeln('программы преобразования и осуществляющие'); writeln('управление всем вычислительным процессом в ЭВМ'); readln(c); if c='процессор' then x:=x+1; writeln('Из трех возможных угадали',x); readkey; end. </pre>
--	--

2-й этап. Игра «Поле чудес»

Ведущий предлагает капитанам подойти к компьютерам и сыграть в «Поле чудес». При оценке учитывается правильность ответа, а также количество попыток. Члены жюри контролируют ход игры. Максимальная оценка – 5 баллов. Ниже приведена программа игры.

Программа 2.1	Программа 2.2
<pre> 5 REM ПОЛЕ ЧУДЕС. ТЕМА: ИНФОРМАТИКА 10 PRINT «ЯЗЫК ПРОГРАММИРОВАНИЯ» 20 INPUT A\$=« РАПИРА» 30 X=LEN (A\$) 40 PRINT «КОЛИЧЕСТВО БУКВ В СЛОВЕ»; X 50 DIM B\$ (X) 60 FOR I=1 TO X 70 B\$ (I)=MID\$ (A\$, I,1) 80 NEXT I 90 PRINT « ВЫ НАЗОВЕТЕ </pre>	<pre> Program P2_2; uses crt; label 90,190,210,220,230; var x,i,k:word; a,c,d,e:string[30]; begin clrscr; {очистка экрана} writeln('Язык программирования'); a:='рапира'; x:=length(a); writeln('Количество букв в слове',x); </pre>

<pre> СЛОВО ? « 100 INPUT C\$ 110 IF C\$ = « ДА » THEN 190 120 PRINT « НАЗОВИТЕ БУКВУ » 130 INPUT D\$ 140 K=K+1 150 FOR I=1 TO X 160 IF D\$=B\$ (I)THEN PRINT»B\$(«; I ;»)=«; B\$ (I) 170 NEXT I 180 GOTO 90 190 INPUT E\$ 200 IF E\$=A\$ THEN 210 ELSE 220 210 PRINT « ВЫ УГАДАЛИ КОЛ- ВО ПОПЫТОК» ; K : GOTO 230 220 PRINT « ВЫ ПРОИГРАЛИ « 230 END </pre>	<pre> 90: writeln('Вы назовете сло- во?'); readln(c); if c='да' then goto 190; writeln('Назовите букву'); readln(d); k:=k+1; for i:=0 to x do if a[i]=d then begin write('a['i,']=');writeln(d);end; goto 90; 190: readln(e); if a=e then goto 210; goto 220; 210: writeln('Вы угадали. Кол-во попыток',k);goto 230; 220:writeln('Вы проиграли'); readkey; 230:end. </pre>
--	--

3-й этап. Игра « Спортлото »

Этот этап заранее не объявляется. Ведущий предлагает коман-дам составить программу, моделирующую игру «Спортлото»: одной из команд – 5 из 36, другой – 6 из 49. В течение 10 минут команды должны написать и ввести программы в компьютер. Жюри оценивает работу по пятибалльной системе, в оценке учитывается работоспособность и интерфейс программы. Ниже приведен пример программы.

Программа 3.1	Программа 3.2
<pre> 5 REM СПОРТЛОТО 5 ИЗ 36 10 DIM X(5): DIM Y(5) 20 PRINT «ЗАПОЛНИТЕ КАР- ТОЧКУ» 30 FOR I=1 TO 5 40 INPUT X(I) 50 NEXT I 60 PRINT «ВЫИГРЫШНЫЕ НОМЕРА» 70 FOR I=1 TO 5 80 Y(I)=INT(RND(36))+1 </pre>	<pre> Program P3_2; uses crt; var i,j:byte; x,y:array[1..5] of word; begin clrscr; {очистка экрана} writeln(' Заполните карточку'); for i:=1 to 5 do readln(x[i]); writeln('Выиграшные номера'); for i:=1 to 5 do begin </pre>

<pre> 90 PRINT»Y(« ;I; «)=«;Y(I) 100 NEXT I 110 FOR I =1 TO 5 120 FOR J =1 TO 5 130 IF X(I)=Y(J) THEN PRINT «УГАДАЛ»;Y(J) 140 NEXT J 150 NEXT I </pre>	<pre> y[i]:=trunc(random(36)+1); writeln('y['i,']=',y[i]); end; for i:=1 to 5 do for j:=1 to 5 do begin if x[i]=y[j] then writeln('Угадал',y[j]); end; end; readkey; end. </pre>
--	--

4-й этап. Математический фокус

Дается следующее задание: «От суммы номеров дня и месяца рождения отнимите ваш возраст (т.е. число полных лет). От суммы дня рождения и возраста отнимите номер месяца рождения. От суммы вашего возраста и номера месяца отнимите число, обозначающее день рождения. Сообщите результаты».

Ведущий вводит результаты в компьютер и сообщает возраст, дату и месяц рождения капитанам. Далее предлагает командам разгадать секрет фокуса и написать программу. Выполнение задания оценивается по пятибалльной системе. Ниже приведен пример программы для демонстрации фокуса.

Программа 4.1	Программа 4.2
<pre> 5 REM ФОКУС 6 REM A, B, C – РЕЗУЛЬТАТЫ 10 INPUT A, B, C 20 X = (A + B)/2 30 Y = (A + C)/2 40 Z = (B+C)/2 50 PRINT «ДЕНЬ РОЖДЕНИЯ»; X 60 PRINT «НОМЕР МЕСЯЦА РОЖ- ДЕНИЯ»; Y 70 PRINT «ВОЗРАСТ»; Z 80 END </pre>	<pre> Program P4_2; uses crt; var a,b,c,x,y,z:real; begin Readln(a,b,c); x:=(a+b)/2; y:=(a+c)/2; z:=(b+c)/2; writeln('День рождения',x); writeln('месяц рождения',y); writeln('возраст',z); readkey; end. </pre>

5-й этап. Ребус

Командам заранее дается домашнее задание придумать ребус. Команды обмениваются заготовками и приступают к решению. Следующее задание: с помощью графического редактора или графических операторов написать программу, которая позволит получить на экране данный ребус. Оценивается оригинальность ребуса, правильность решения, максимальное использование графических возможностей компьютера, сходство с оригиналом. Максимальная оценка этого задания – 10 баллов.

6-й этап. Математический ребус

Ведущий предлагает капитанам вытянуть карточки с математическими ребусами, к примеру:

1. КО + ЛЯ = ОЛ – Я
2. БЕ * РУ * 4 = БУЕР

В приведенных ребусах каждая буква соответствует какой-либо цифре. Предлагается привести компьютерное решение математических ребусов. Ниже приведена программа решения первого математического ребуса. Максимальная оценка за выполнение задания – 5 баллов.

Программа 5.1	Программа 5.2
<pre>5 REM МАТЕМАТИЧЕСКИЙ РЕБУС 10 FOR K=1 TO 9 20 FOR O=1 TO 9 30 FOR L =1 TO 9 40 FOR I=1 TO 9 50 IF K=O THEN 170 60 IF K=L THEN 160 70 IF K=I THEN 150 80 IF O=L THEN 160 90 IF O=I THEN 150 100 IF L=I THEN 150 110 A = K * 10 + O 120 B = L * 10 + I 130 C = O * 10 + L 140 IF A + B = C - I THEN PRINT K; O; L; I 150 NEXT I 160 NEXT L</pre>	<pre>Program P5_2; Uses crt; Label 150; VAR k,o,l,i,a,b,c:integer; BEGIN For k:=1 to 9 do For o:=1 to 9 do for l:=1 to 9 do for i:=1 to 9 do begin if k=o then goto 150; if k=l then goto 150; if k=i then goto 150; if o=l then goto 150; if o=i then goto 150; if l=i then goto 150; a:= k * 10 + o; b:= l * 10 + i; c:= o * 10 + l;</pre>

170 NEXT O 180 NEXT K	if a+b=c-i then write(k,' ',o,' ',l,' ',i,' '); 150: end; readkey; end.
--------------------------	--

Решение: 1534, 1754, 1864, 1974, 2518, 2738, 2958, 4827, 4937, 5712, 5932.

После подведения итогов КВН участники вместе с преподавателем обсуждают и анализируют программы.

3.7. ПРОЕКТНАЯ ФОРМА ОБУЧЕНИЯ

Учебный проект достаточно широко используется при изучении различных тем и курсов информационных технологий. Проектная форма (нелинейная технология обучения) несовместима с задачей последовательного (линейного) изложения материала, требующего пошагового овладения научными понятиями, логически последовательного прохождения тем курса, чаще всего построенных в форме «лекция – практикум». Главной особенностью учебных проектов является проблема постановки учебных целей и задач по предметной области, которые позволяют преподавателю формировать путь их достижения, предложить необходимый инструментарий, методический материал, инструкции, опыт. Проектная деятельность учащихся требует достаточно длительного времени. Поэтому целесообразно использовать учебные проекты во внеклассных занятиях по информатике как в индивидуальной, так и в коллективной форме обучения.

Основные этапы нелинейной технологии обучения:

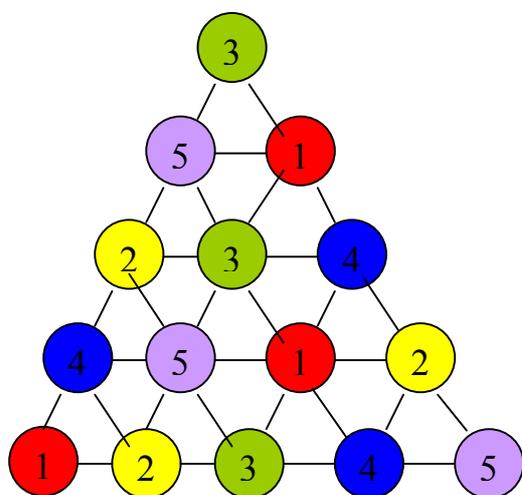


Рис.26

1. Постановка задачи.
2. Обзор источников по предметной области (обзор литературы).
3. Выбор и освоение инструментов познания.
4. Составление графика выполнения проекта.
5. Выполнение работ по проекту.
6. Анализ результатов, выводы.
7. Систематизация и обобщение собственных знаний по данной предметной области.

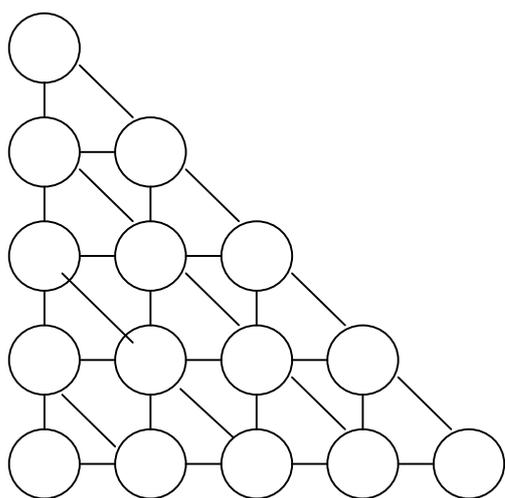


Рис. 27

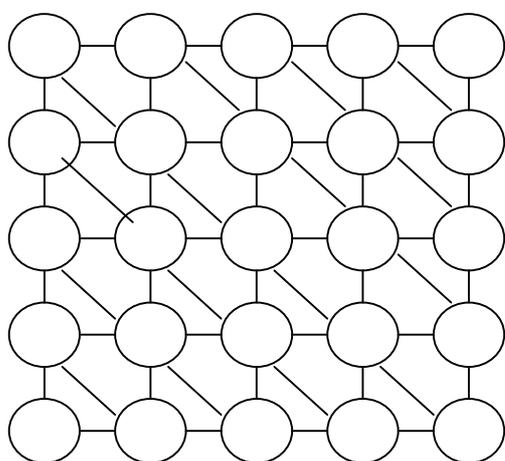


Рис. 28

В качестве примера предлагаем проект «Решение головоломки «Цветной квадрат»». Данный проект можно предложить ученику или группе учеников после изучения основ алгоритмизации и программирования.

В журнале «Квант» (1990. № 9, с. 69) предложена головоломка «Цветной треугольник»: на узлах правильной треугольной сетки в треугольнике (с n узлами на стороне) расставить фишки n цветов так, чтобы на линиях сетки, параллельных сторонам, не было одноцветных фишек. Приведены два решения для $n=5$. На рис. 26 показано одно из решений данной головоломки (фишки из n цветов обозначены через 1, 2, ..., n): 1 – красный, 2 – желтый, 3 – зеленый, 4 – синий, 5 – фиолетовый.

Для решения данной головоломки используем компьютер.

Очевидно, что самое простое решение головоломки «Цветной треугольник» для $n=5$ – организация 15 вложенных циклов с проверкой условия «цветной треугольник». Данное решение будет носить частный характер.

Видоизменим предложенную головоломку. Треугольник на рис. 26 преобразуем в треугольник на рис.27, а потом дополним треугольник до квадрата рис.28.

Получим головоломку «Цветной квадрат», удовлетворяющий следующему условию: расставить фишки n цветов так, чтобы на линиях сетки, параллельных сторонам и одной из диагонали квадрата, не было одноцветных фишек.

Решение головоломки «Цветной квадрат» приведено в программе 1. Программа написана на Turbo Pascal 7.0. Опишем алгоритм решения головоломки «Цветной квадрат».

1. Генератором перестановок без повторения формируем всевозможные строки «Цветного квадрата» (Процедура Perist).

2. Удаляем строки так, чтобы на столбцах не было фишек, одинаковых по цвету с фишками, расположенными на первой строке. Отобранные строчки переименовываем в массив (massiv) строк, одновременно подсчитываем количество отображенных строк – K. (Процедура RenDel)

3. Организуем размещение из K строк по N; для этого используем генерацию сочетаний без повторения номеров строк массива с последующим использованием процедуры перестановок без повторения, проверки условия «Цветной квадрат» и вывод результата (Процедура Print).

Приведем некоторые результаты решений:

n=3 – одно решение; n=4 – решений нет; n=5 – шесть решений (программа работает около 20–25 минут); n=6 – решения нет (программа проработала более 2,5 часов); n=7 – решение есть; (компьютер Pentium 166). (Приложение 1).

Для получения решения головоломки «Цветной треугольник» нужно поменять строку с комментарием {**} на *For j1:=1 to n-f do* и добавить перед {***} *f:=f+1*. В разделе переменных объявить *f:type*, а также изменить процедуру вывода результата: в строчке с комментарием {4*} добавить *f:=0*; строчку с {5*} заменить на *for j1:=1 to n-f do*. В разделе описаний переменных в процедуре вывода описать *f:byte*.

Программа 1

```
{головоломка о цветных шарах; «Цветной квадрат»}
Program P1_0;
uses crt;
clrscr; {очистка экрана}
const nmax=9;mmax=2800;
label dd,ss;
type
  t1=array[1..nmax] of byte;
  t2=array[1..mmax,1..nmax] of byte;
var
  el,sup,inf,c,v:t1; per,massiv:t2;
  n,i,j,x,i1,j1,h,l:byte;
  k,r,q,a,m:integer;
procedure Perist( var el:t1; i:byte);
var
```

```

j,tmp:byte;
BEGIN
  for j:=i to n do
  begin
    if i<>j then begin
      tmp:=el[i];
      el[i]:=el[j];
      el[j]:=tmp
    end;
    if i<n then
      Perist(el,i+1) {рекурсивный вызов процедуры}
    else
      begin {присваивание перестановки 2-мерному массиву}
        for i:=1 to n do
          per[k,i]:=el[i];
          k:=k+1;
        end;
        if i<>j then begin
          tmp:=el[i];
          el[i]:=el[j];
          el[j]:=tmp
        end
      end
    end
  END;
PROCEDURE RenDel(n:byte;var k:integer;var massiv:t2;var
per:t2);
  label dd;
  VAR i,j,x: byte;
  BEGIN
dd: for i:=2 to k do
  for j:=1 to n do
    if (per[i,j]=per[1,j]) then
      begin x:=i;
        for i:=x to k-1 do
          for j:=1 to n do per[i,j]:=per[i+1,j];
          for j:=1 to n do per[k,j]:=0;k:=k-1;goto dd;
        end; k:=k-1;
      for i:=1 to k do
        for j:=1 to n do
          massiv[i,j]:=per[i,j];
        END;
PROCEDURE Print(var n:byte);
  VAR i1,j1:byte;

```

```

BEGIN      {4*}
  for i1:=1 to n do
  begin
    for j1:=1 to n do {5*}
      write(massiv[v[i1],j1], ' ');writeln;
    end; {6*}
  END;
Begin {Основная программа}
  {Вводим исходные данные}
  write ('Введите число элементов n');
  read(n); k:=1;
  for i:=1 to n do el[i]:=i;
  {Вызываем процедуру формирования перестановок и про-
  цедуру удаления строк перестановки и переименование их в мас-
  сив строк }
  Perist(el,1);
  RenDel(n,k,massiv,per);
  {Организация размещений строк массива без повторения;
  для этого используем генерацию сочетаний без повторения номе-
  ров строк массива с последующим использованием процедуры пе-
  рестановок без повторения}
  j:=0;r:=1;q:=k;k:=1;
  for i:=1 to n do
  begin
    sup[i]:=q-i+1;inf[i]:=n-i+1;c[i]:=inf[i]
  end;
  while j<=n do
  begin
    for i:=n downto 1 do
      el[i]:=c[i];
    Perist(el,1);
    for a:=1 to q-1 do {Проверка условия «цветной квадрат» }
    begin
      for i1:=1 to n do
        v[i1]:=per[a,i1];
      for i1:=2 to n do
        for j1:=1 to n do {**}
          begin
            for h:=1 to n do
              begin
                if i1-h>=0 then
                  if (massiv[v[i1],j1]=massiv[v[i1-h],j1]) then goto ss;
              end;
            end;
          end;
        end;
      end;
    end;
  end;

```

```

    for l:=1 to n do
    begin
    if (i1-l>=0) and (j1+l<=n) then
    if massiv[v[i1],j1]=massiv[v[i1-l],j1+l] then goto ss;
    end;
    end;    {***}
Print(n);writeln;m:=m+1;{Вызов процедуры вывода результата}
ss: end;{Конец проверки условия «цветной квадрат»}
k:=1; j:=1;
while (j<=n) and (c[j]=sup[j]) do
j:=j+1;
if j<=n then c[j]:=c[j]+1;
for i:=j-1 downto 1 do
begin
inf[i]:=c[i+1]+1;
c[i]:=inf[i];
end;
end; Write('kol-vo m=',m);
writeln(' end');
Readkey;
End.

```

Приложение 1

```

N=3    3 1 2
        2 3 1
        1 2 3

```

```

N=5    5 1 2 3 4  5 1 2 3 4  5 1 2 3 4  4 5 1 2 3  4 5 1 2 3  4 5 1 2 3
        4 5 1 2 3  3 4 5 1 2  2 3 4 5 1  3 4 5 1 2  2 3 4 5 1  1 2 3 4 5
        3 4 5 1 2  1 2 3 4 5  4 5 1 2 3  2 3 4 5 1  5 1 2 3 4  3 4 5 1 2
        2 3 4 5 1  4 5 1 2 3  1 2 3 4 5  1 2 3 4 5  3 4 5 1 2  5 1 2 3 4
        1 2 3 4 5  2 3 4 5 1  3 4 5 1 2  5 1 2 3 4  1 2 3 4 5  2 3 4 5 1

```

Внимательное рассмотрение полученного решения позволило заметить одно обстоятельство.

Цветной квадрат для $n=5$ получается из квадрата

```

1 2 3 4 5
1 2 3 4 5
1 2 3 4 5
1 2 3 4 5
1 2 3 4 5

```

«круговой перестановкой» цифр строк квадрата:

первая строчка – 0 круговых перестановок;
 вторая строчка – 1 круговая перестановка;
 третья строчка – 2 круговые перестановки;
 четвертая строчка – 3 круговые перестановки;
 пятая строчка – 4 круговые перестановки;

Обозначим это решение через количество перестановок строк исходного квадрата – (0, 1, 2, 3, 4). Второе решение получается умножением этих цифр на 2 – (0, 2, 4, 6, 8), что равнозначно (0, 2, 4, 1, 3), т. к. 5 круговых перестановок приводят к исходной строчке. Третье решение – умножением на 3 – (0, 3, 6, 9, 12) или (0, 3, 1, 4, 2). Умножение на 4 – (0, 4, 8, 12, 16) или (0, 4, 3, 2, 1) дает одинаковые числа на линиях сетки, параллельные фиксированной диагонали. Далее при умножении на другие числа решения повторяются. Итого нашли 3 решения для «Цветного квадрата» для n=5.

1 2 3 4 5	1 2 3 4 5	1 2 3 4 5	1 2 3 4 5
5 1 2 3 4	4 5 1 2 3	3 4 5 1 2	2 3 4 5 1
4 5 1 2 3	2 3 4 5 1	5 1 2 3 4	3 4 5 1 2
3 4 5 1 2	5 1 2 3 4	2 3 4 5 1	4 5 1 2 3
2 3 4 5 1	3 4 5 1 2	4 5 1 2 3	5 1 2 3 4

Аналогичные рассуждения позволяют предположить для головоломки «Цветной квадрат» при n=7 пять решений: (0, 1, 2, 3, 4, 5, 6); (0, 2, 4, 6, 1, 3, 5); (0, 3, 6, 2, 5, 1, 4); (0, 4, 1, 5, 2, 6, 3); (0, 5, 3, 1, 6, 4, 2). Построение «цветных квадратов» подтверждают выдвинутое предположение. Тогда можно выдвинуть предположение, что данный алгоритм работает для любых нечетных n при решении головоломки «цветной квадрат». Для N-«цветного квадрата» решений будет n-2. Решения головоломки: (0, (1*k) mod n, (2*k) mod n, ..., (n-1)*k mod n), где k=1, 2, ..., n-2. Программа 2 реализует проверку выдвинутой гипотезы.

Программа 2

```

{Программа 2. «Цветной квадрат»}
Program P2;
uses crt;
var
  a:array[1..11,1..11] of byte;
  cir:array[1..11] of byte;
  i,j,n,k,q:byte;
  h,l:byte;
  
```

```

PROCEDURE Print(var n:byte);{Процедура вывода результата }
  VAR i,j:byte;
BEGIN
  for i:=1 to n do begin
    for j:=1 to n do
      write(a[i,j], ' ');writeln; end;
  END;
PROCEDURE Krug;{Процедура круговых перестановок строк
цветного квадрата}
  VAR i,j,k,q:byte;
BEGIN
  for i:=2 to n do
    for j:=1 to n do a[i,j]:=a[1,j];
    for i:=2 to n do
      for k:=1 to cir[i] do
        begin
          q:=a[i,n];
          for j:=n downto 2 do
            a[i,j]:=a[i,j-1];
            a[i,1]:=q;
          end;
        END;
  Begin
  writeln('Введи n'); read(n);
  for j:=1 to n do a[1,j]:=j;
  writeln('Введи k-е решение');read(k);
  for i:=2 to n do begin cir[i]:=((i-1)*k) mod n;end;
  Krug;
  {Проверка условия «цветной квадрат»}
  for i:=2 to n do
    for j:=1 to n do
      begin
        for h:=1 to n do
          begin
            if i-h>0 then
              if a[i,j]=a[i-h,j] then exit;
            end;
          for l:=1 to n do
            begin
              if (i-l>0) and (j+l<=n) then
                if a[i,j]=a[i-l,j+l] then exit;
            end;
          end;
        end;
      end;
  end;

```

```
Print(n);  
readln;  
End.
```

3.8. ШКОЛЬНАЯ ЭЛЕКТРОННАЯ ПЕЧАТЬ

Несмотря на сегодняшнее обилие научной, научно-популярной литературы и периодических изданий, школьная печать не утратила своей роли. Конечно, компьютер внес свои коррективы в подбор и подготовку материала, однако цели и задачи остались прежними, и то, о чем пойдет речь, для большинства учителей не является новым, многие из них довольно давно и успешно используют школьную печать.

Под умелым руководством учителя школьная печать является важным фактором, организующим и направляющим жизнь школьников. Печать в школе традиционно осуществляется в виде газет, календарей, бюллетеней, викторин, информационных листков и др. Опыт показывает, что хорошо составленная и интересно оформленная газета привлекает многих читателей: поддерживает интерес к предмету, дает материал для размышлений в часы досуга, выступает организатором школьных мероприятий – викторин, олимпиад, кружков, освещает основные направления деятельности школы по предмету, отмечает знаменательные события и факты, направляет индивидуальную деятельность учащихся.

Организаторами выпуска школьной печати могут являться предметные кружки, отдельные классы и даже группы учащихся, проявившие инициативу. Каждый номер выходит при участии учителя. Школьная печать накладывает определенные требования к подбору материала: он должен быть полезным, интересным и доступным для широкого круга учащихся. Как правило, в печати находят место задачи повышенной трудности, занимательные задачи, головоломки, загадки, ребусы, кроссворды, освещение одной конкретной темы или вопроса, обзор новинок литературы и аннотация к интересным материалам, к примеру, в рубрике «По страницам журнала «Информатика и образование».

Выпуск стенной печати осуществляется редколлегией (постоянной или временной). В ходе подготовки стенной печати указывается тема и срок выпуска, планируются рубрики, тематики статей.

Название может быть постоянным, если выпуск осуществляется одной редколлегией. В газете могут быть следующие рубрики:

1) Наш календарь (краткое сообщение по истории развития информатики и вычислительной техники).

2) Новые информационные технологии (новинки компьютерной техники, программного обеспечения).

3) Информатика в персоналиях (освещает биографии и научную деятельность выдающихся людей, внесших вклад в становление и развитие информатики и вычислительной техники).

4) Наш словарь (объясняются смысл и происхождение предметных терминов).

5) Информатика в нашей школе (о работе предметных кружков, о предстоящих или прошедших мероприятиях).

6) За страницами учебника (список статей, опубликованных в научно-популярных изданиях по различным тематикам).

7) Готовься к экзаменам. Приводятся примерные вопросы и задачи к выпускному экзамену по информатике, перечень дополнительной литературы для подготовки к экзамену.

8) Выбирай профессию. Рассказы о профессиях, связанных с информатикой и компьютерами. Приводятся примерные вступительные тесты и билеты с ответами и решениями по информатике.

9) Программисты и информатики шутят. К примеру, можно использовать материалы «Веселых уроков» из «Информатики и образования» №5 за 1987 г., №2, 4 за 1988 г. и др.

Систематичность выпуска стенной печати дает возможность популяризировать и направлять на более глубокое изучение предмета. Материал можно подбирать из научных книг, журналов, газет, собирая его в отдельные папки или конверты, что облегчит подготовку номеров стенной печати.

Публикация школьной электронной газеты состоит в подготовке документов HTML и всех сопроводительных файлов (изображений, мультимедиа и прочего). Напомним: HTML – Hyper Text Markup Language – язык разметки гипертекста. Гипертекст – это многомерный текст, т.е. такая организация документов, при которой один документ или текст может включать в себя разнонаправленные ссылки на другие. Иными словами, способ представления информации при помощи связей между документами.

Пример создания и просмотра простейшей гипертекстовой электронной газеты.

1. Запустите текстовый редактор Блокнот. (Пуск – Программы – Стандартные – Блокнот).

2. Введите Документ 1 и сохраните под именем htm2.htm; документ 2 под именем htm4.htm; документ 3 – htm5.htm; документ 4 – htm6.htm; документ 6 – htm7.htm. Графический файл pic11.gif поместите вместе с указанными файлами, к примеру на рабочем столе.

3. Запустите программу Internet Explorer (Пуск – Программы - Internet Explorer).

4. Дайте команду Файл – Открыть. Нажмите кнопку Обзор и откройте файл htm2.htm. Электронная газета на экране вашего компьютера. На рис. 29 представлен внешний вид первой страницы газеты.

Документ 1

```
<HTML>
<HEAD>
  <META http-equiv=«Content-Type»
  content=«A-text/htm2; charset=windows-1251»>
  <TITLE> ГАЗЕТА </TITLE>
</HEAD>
<BODY>
  <CENTER>
    <H1> РАПИРА </H1>
    <H2> Веселый урок информатики</H2>
    <H2>Школьная электронная газета школы №104 г. Уфы </H2>
  </H3>
</CENTER>
<HR>
<P>
  <H4> СОДЕРЖАНИЕ </H4>
  <IMG src=pic11.gif align=right width=160 height=140 >
  <OL type=1>
    <LI><A href=HTM4.htm> Советы начинающим...</A>
    <LI><A href=HTM5.htm> Мысли и фразы...</A>
    <LI><A href=HTM6.htm> Чтобы это значило ?</A>
  </P>
</BODY>
</HTML>
```

Документ 2

```
<HTML>
<HEAD>
<META http-equiv=«Content-Type»
content=«A-text/htm2; charset=windows-1251»>
<TITLE> ГАЗЕТА </TITLE>
</HEAD>
<BODY>
<H3>Советы начинающим...</H3>
<UL>
<LI>Чтобы овладеть искусством программирования,
    нужно в первую очередь научиться владеть собой.
<LI>Если по-твоему программа составлена правильно,
    это еще не значит, что с этим согласится машина.
<LI>Не разговаривай с друзьями и близкими на алгоритми-
    ческом языке – тебя могут неправильно понять.
<LI>Занимаясь улучшением работающей программы, помни,
    что это верный способ сделать ее хуже.
</BODY>
</HTML>
```

Документ 3

```
<HTML>
<HEAD>
<META http-equiv=«Content-Type»
content=«A-text/htm2; charset=windows-1251»>
<TITLE> ГАЗЕТА </TITLE>
</HEAD>
<BODY>
<H3>Мысли и фразы...</H3>
<UL>
<LI>Если вы не умеете программировать – это не самый
    худший из ваших недостатков.
<LI>Программист не стареет – устаревают его программы.
<LI>Программисты, как и ЭВМ, бывают 1, 2, 3, 4-го и т.д.
    поколений.
<LI>Каждая новая программа – это, как правило, хорошо за-
    бытая старая.
```

Диагностика: лучший способ еще раз убедиться, что во всем

виновата машина.

</BODY>
</HTML>

Документ 4

```
<HTML>
<HEAD>
<META http-equiv=«Content-Type»
content=«A-text/htm2; charset=windows-1251»>
<TITLE> ГАЗЕТА </TITLE>
</HEAD>
<BODY>
<H3>Чтобы это значило ?</H3>
<OL>
<LI><A href=НТМ7.htm#1>Считало</A>
<LI><A href=НТМ7.htm#2>Запоминало</A>
<LI><A href=НТМ7.htm#3>Казало</A>
<LI><A href=НТМ7.htm#4>Крутило-запоминало</A>
<LI><A href=НТМ7.htm#5>Ходило-приставало</A>
<LI><A href=НТМ7.htm#6>Моргало</A>
<LI><A href=НТМ7.htm#7>Зависало</A>
</OL>
</BODY>
</HTML>
```

Документ 5

```
<HTML>
<HEAD>
<META http-equiv=«Content-Type»
content=«A-text/htm2; charset=windows-1251»>
<TITLE> ГАЗЕТА </TITLE>
</HEAD>
<BODY>
<H3>Что бы это значило ? (Ответы)</H3>
<OL>
<A name=1> <! --Первая метка-->
<LI>Процессор
```

```

<A name=2> <! --Вторая метка-->
<LI>ОЗУ
<A name=3> <! -- Третья метка-->
<LI>Дисплей
<A name=4> <! -- Третья метка-->
<LI>Диск
<A name=5> <! -- Третья метка-->
<LI>Пользователь
<A name=6> <! -- Третья метка-->
<LI>Курсор
<A name=7> <! -- Третья метка-->
<LI>Операционная система
</OL>
</BODY>
</HTML>

```

Создавать HTML документы можно не только вручную, но и с помощью визуальных редакторов, таких, как Front Page Express, Front Page 2000 или других, которые без особого труда позволяют создавать Web-страницы, электронные публикации, что облегчает подготовку школьной электронной публикации.

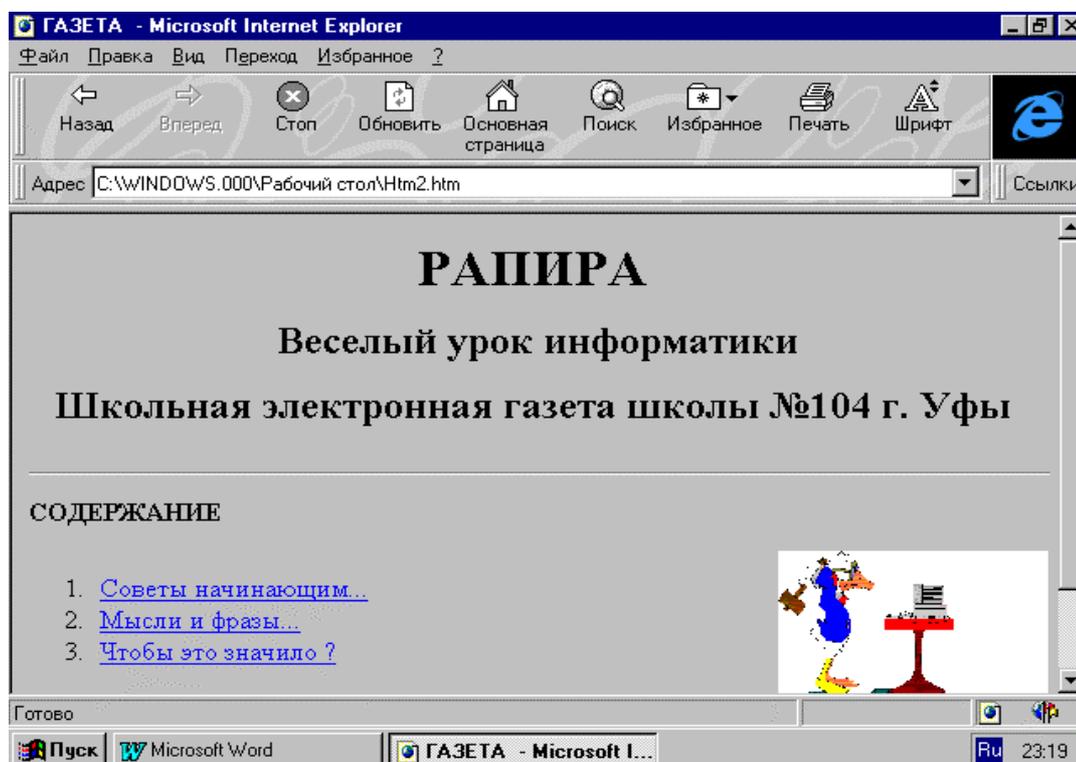


Рис. 29

ГЛАВА 4. РЕШЕНИЕ ЗАНИМАТЕЛЬНЫХ ЗАДАЧ

4.1. ЧИСЛОВАЯ ЗАГАДКА ЦИФРОВЫХ КЛАВИШ

На клавиатуре микрокалькулятора (компьютера) цифровые клавиши размещаются обычно так:

7	8	9
4	5	6
1	2	3

Составим четырехзначные числа, которые набираются по часовой или против часовой стрелки из цифр в углах любого прямоугольника, образованного цифровыми клавишами. Например: 4631, 2365, 9317.

Оказывается, что эти числа делятся на 11 без остатка. Предлагаем: поэкспериментировать с другими числами; поискать числа, которые набираются указанным способом, но не делятся на 11, ответить, в чем секрет чисел, которые разделились на 11 без остатка.

Анализ показывает, что таких чисел 72. С каждой клавиши можно составить по четыре числа по часовой стрелке и по четыре числа против часовой стрелки, т.е. всего от одной клавиши восемь чисел. Всего клавиш 9. Итого 72 числа. Эту задачу можно решить на микрокалькуляторе, но т.к. количество чисел 72, решение задачи доверим компьютеру. Решение задачи приведено в программе 1.1 на языке Qbasic и в программе 1.2 на языке Turbo Pascal.

Программа 1.1	Программа 1.2
<pre>S = 1 'ФОРМИРОВАНИЕ ЭЛЕМЕН- ТОВ МАССИВА FOR I = 1 TO 3 FOR J = 1 TO 3 A(I, J) = S S = S + 1 NEXT j, i ' ФОРМИРОВАНИЕ ЦИФР ЧИСЛА, КОТОРЫЕ НАБИРА- ЮТСЯ ПО ЧАСОВОЙ СТРЕЛ-</pre>	<pre>Program P1_2; Uses crt; LABEL 260,270,280,285; var i,j,s,x,y,l:INTEGER; ch,z:LONGINT; clav: file of integer; a: ARRAY[1..3,1..3] of integer; BEGIN assign (clav,'igr.dat'); rewrite (clav); s:= 1; {Формирование элементов</pre>

КЕ ИЗ ЦИФР, В УГЛАХ ПРЯМОУГОЛЬНИКА, ОБРАЗОВАННОГО ЦИФРОВЫМИ КЛАВИШАМИ'

```

FOR X = -2 TO 2
IF X = 0 THEN 285
FOR Y = -2 TO 2
IF Y = 0 THEN 280
FOR I = 3 TO 1 STEP -1
FOR J = 3 TO 1 STEP -1
IF I+X<1 OR I+X>3 THEN 270
IF J+Y<1 OR J+Y>3 THEN
260
' Получение искомого числа
CH=A(I,J)*1000+A(I+X,J)*100
+A(I+X,J+Y)*10+A(I,J+Y)
' ПРОВЕРКА ДЕЛИМОСТИ
НАЦЕЛО ЧИСЛА НА 11
Z = CH MOD 11
' ВЫВОД ЦИФР ЧИСЛА, ЕСЛИ
УСЛОВИЕ ДЕЛИМОСТИ ЧИСЛА
ВЫПОЛНЯЕТСЯ
IF Z = 0 THEN PRINT A(I, J); A(I
+ X, J); A(I + X, J + Y); A(I, J + Y)
L = L + 1 ' ПОДСЧЕТ КОЛИЧЕСТВА
ПРОВЕРЯЕМЫХ ЧИСЕЛ
260 NEXT J
270 NEXT I
280 NEXT Y
285 NEXT X
PRINT «KOL=»; L ' ВЫВОД КО-
ЛИЧЕСТВА ПРОВЕРЯЕМЫХ
ЧИСЕЛ
300 END

```

массива}

```

FOR i:= 1 TO 3 DO
FOR j:= 1 TO 3 DO
BEGIN
a[i,j]:=s;
s:=s+1;
END;
{ Формирование числа, кото-
рые набираются по часовой
стрелке из цифр, в углах прямо-
угольника, образованного циф-
ровыми клавишами}
FOR x:= -2 TO 2 DO
BEGIN
IF (x=0) THEN GOTO 285;
FOR y:= -2 TO 2 DO
BEGIN
IF (y=0) THEN GOTO 280;
FOR i:= 3 DOWNT0 1 DO
BEGIN
FOR j:= 3 DOWNT0 1 DO
BEGIN
IF (i+x<1) OR (i+x>3) THEN
GOTO 270;
IF (j+y<1) OR (j+y>3) THEN
GOTO 260;
{Получение искомого числа}
ch:= a[i,j]*1000+a[i+x,j]*100+ a[i+
x,j+y]*10+a[i,j+y];
{Проверка делимости нацело
числа на 11}
z:= ch MOD 11;
{ Вывод цифр числа, если ус-
ловие делимости числа выпол-
няется}
IF (z=0) THEN
WRITELN(a[i,j],a[i+x,j],a[i+x,j+y],a[
i,j+y]);
Write (clav, a[i,j]);
L:= L+1;{Подсчет количества
проверяемых чисел}
260: END;
270: END;

```

	<pre> 280: END; 285: END; WRITELN('kol=', L:8:4); {Вывод количества проверяемых чисел} Close (clav); END. </pre>
--	--

Примечание к программам. Представим цифровые клавиши как элементы массива:

```

A31   A32   A33
A21   A22   A23
A11   A12   A13

```

В приложении 1 приведены числа, которые набираются по часовой стрелке. Общее количество чисел 36. Для того, чтобы получить числа, образованные из цифр путем набора против часовой стрелки, нужно изменить строчки: получение искомого числа и вывод цифр числа, если выполняется условие делимости.

Для языка Qbasic:

```

ch=a(i,j)*1000+a(i,j+y)*100+a(i+x,j+y)*10+a(i+x,j)
IF z=0 THEN PRINT a(i,j); a(i,j+y); a(i+x,j+y); a(i+x,j).

```

Для языка Turbo Pascal:

```

ch=a[i,j]*1000+a[i,j+y]*100+a[i+x,j+y]*10+a[i+x,j];
IF z=0 THEN WRITELN (a[i,j];a[i,j+y];a[i+x,j+y];a[i+x,j]).

```

В приложении 2 приведены числа, которые набираются против часовой стрелки. Общее количество их 36, т.е. все числа, образованные указанным способом, делятся на 11 без остатка.

Приложение 1

```

9317   6974
9328   3641
8217   6985
8239   5874
7128   3652
7139   2541
9647   5896
6314   4785
9658   2563
8547   1452
6325   4796

```

Приложение 2

```

9713   6589
9823   6479
8712   3146
8932   5698
7821   5478
7931   3256
9746   2145
6413   4587
9856   2365
8745   1254
6523   4697

```

5214	1463	5412	1364
8569	3971	8965	3179
7458	3982	7854	3289
5236	2871	5632	2178
4125	2893	4521	2398
7469	1782	7964	1287
4136	1793	4631	1397

Рассмотрим числа, образованные указанным выше способом, и разделим их на две группы:

- 1) цифра разряда тысяч числа меньше цифры единиц;
- 2) цифра разряда тысяч числа больше цифры единиц.

Например: 1254, 1397 и 9746, 8712.

Разложим числа на суммы.

$$\text{Первая группа: } 1254 = 1221 + 33,$$

$$1397 = 1331 + 66.$$

$$\text{Вторая группа: } 9746 = 6446 + 3300,$$

$$8712 = 2112 + 6600.$$

Числа 33, 66, 3300, 6600 делятся на 11 без остатка. Числа 1221, 1331, 6446, 2112 четырехзначные, симметричные числа. Аналогичным образом можно разложить все числа, приведенные в приложениях 1 и 2. Покажем, что все симметричные числа делятся без остатка на 11.

Рассмотрим симметричные числа в общем виде и разложим на суммы:

$$\begin{aligned} abba &= a*1000 + b*100 + b*10 + a = a*(990 + 10) + b*(99 + 1) + \\ &+ b*10 + a = a*990 + a*10 + b*99 + b + b*10 + a = a*990 + b*99 + \\ &+ b*11 + a*11. \end{aligned}$$

Все слагаемые делятся на 11 без остатка. Таким образом, все четырехзначные числа, которые набираются по часовой или против часовой стрелки, в углах любого прямоугольника, образованные цифровыми клавишами, делятся на 11 без остатка.

4.2. НАЗОЙЛИВАЯ РАЗНОСТЬ

Напишите четырехзначное число, не все цифры которого одинаковы. Из цифр написанного числа составьте два новых числа: d_{\max} – наибольшее возможное число и d_{\min} – наименьшее возможное число. Найдите разность $gaz = d_{\max} - d_{\min}$ и проделайте то же самое с полученной разностью. Повторяя эти действия ка-

кое-то количество раз, вы непременно придете к разности 6174, которая все время и будет повторяться при продолжении процесса.

Пусть, например, первоначальное число есть 4818.

dmax = 8841; dmin = 1488; raz = 7353;
 dmax = 7533; dmin = 3357; raz = 4176;
 dmax = 7641; dmin = 1467; raz = 6174;
 dmax = 7641; dmin = 1467; raz = 6174 и т. д.

Имеет ли место указанное выше явление для любого четырехзначного числа? Предоставим ответить на этот вопрос компьютеру. Опишем алгоритм проверки указанного явления.

1. Ввод четырехзначного числа – x.
2. Организация одномерного массива A(4) для хранения цифр четырехзначного числа.
3. Отделение цифр четырехзначного числа и присвоение этих цифр элементам массива (a(1), a(2), a(3), a(4)).
4. Проверка условия, что не все цифры одинаковы.
5. Упорядочивание элементов массива по возрастанию.
6. Составление наибольшего возможного числа dmax и его вывод.
7. Составление наименьшего возможного числа dmin и его вывод.
8. Нахождение разности между максимально возможным и минимально возможным. Вывод разности.
9. Организация счетчика повторения алгоритма. Вывод значения счетчика.
10. Проверка условия, что разность равна 6174 или счетчик равен 100.

Если выполняются приведенные выше условия, то конец работы. В противном случае перейти к пункту 3.

Данный алгоритм реализован в программе 2.

Программа 2.1	Программа 2.2
INPUT « ВВЕДИТЕ РАЗРЯДНОСТЬ ЧИСЛА»; N INPUT « ВВЕДИТЕ CONST»; C INPUT « ВВЕДИТЕ ЧИСЛО»; X DIM A(N): PRINT X 'ОТДЕЛЕНИЕ ЦИФР ЧИСЛА 30 FOR I = 1 TO N	Program P2_2; Uses crt; label 30,50; var a:array [1..4] of integer; n,c,x,i,d,q,z,j,v,w,pop, dmax,dmin,raz:integer; begin writeln('Введите разрядность n');

<pre> D = X MOD 10 A(I) = D X = X \ 10 NEXT I Q = 1 'ПРОВЕРКА, ЧТО НЕ ВСЕ ЦИФРЫ ОДИНАКОВЫ FOR I = 2 TO N IF A(I)=A(1) THEN Q=Q+1 IF Q = N THEN 50 NEXT I 'СОРТИРОВКА ЦИФР ПО ВОЗРАСТАНИЮ FOR I = 1 TO N FOR J = 1 TO N IF A(I)<A(J) THEN Z= A(I): A(I)=A(J): A(J)=Z NEXT J NEXT I 'НАХОЖДЕНИЕ МАКСИ- МАЛЬНО И МИНИМАЛЬНО ВОЗМОЖНЫХ ЧИСЕЛ 'НАХОЖДЕНИЕ РАЗНОСТИ DMAX = 0: DMIN = 0 FOR I = 1 TO N DMAX=DMAX+A(I)*10^(I-1) DMIN=DMIN+A(I)*10^(N-I) NEXT I PRINT «DMAX=»; DMAX PRINT «DMIN=»; DMIN X=DMAX-DMIN:POP=POP+1 PRINT «RAZ=»; X, «POP=»; POP PRINT IF X = C OR POP = 50 THEN 50 ELSE GOTO 30 50 END </pre>	<pre> read(n); writeln('Введите const c'); read(c); writeln('Введите число x'); read(x); {Отделение цифр числа} 30: for i:=1 to n do begin d:=x mod 10; a[i]:=d; x:=x div 10; end; q:=1; {Проверка, что не все цифры одинаковы} for i:=2 to n do begin if a[i]=a[1] then q:=q+1; if q=n then goto 50; end; {Сортировка по возрастанию} for j:=1 to n-1 do for i:=1 to n-j do begin IF a[i]>a[i+1] then begin z:=a[i]; a[i]:=a[i+1];a[i+1]:=z; end;end; {Нахождение максимально и минимально возможных чисел} {Нахождение разности} dmax:=0; dmin:=0;v:=1;w:=1; for i:=1 to n-1 do w:=w*10; for i:= 1 to n do begin dmax:=dmax+a[i]*v; dmin:= dmin+a[i]*w; v:=v*10;w:=w div 10; end; writeln('dmax=', dmax); writeln('dmin=', dmin); x:=dmax-dmin;pop:=pop+1; </pre>
--	--

	<pre>writeln('raz=',x,'pop=',pop); writeln; if (x=c) or (pop=50) then goto 50 else goto 30; 50: readkey; end.</pre>
--	---

Результат работы программы для чисел 1960 и 1999 приведен в приложении 1, где pop – количество повторений алгоритма до достижения разности 6174.

Приложение 1

1960		1999	
dmax=9610		dmax=9991	
dmin=169		dmin=1999	
raz=9441	pop=1	raz=7992	pop=1
dmax=9441		dmax=9972	
dmin=1449		dmin=2799	
raz=7992	pop=2	raz=7173	pop=2
dmax=9972		dmax=7731	
dmin=2799		dmin=1377	
raz=7173	pop=3	raz=6364	pop=3
dmax=7731		dmax=6543	
dmin=1377		dmin=3456	
raz=6354	pop=4	raz=3087	pop=4
dmax=6543		dmax=8730	
dmin=3456		dmin=378	
raz=3087	pop=5	raz=8352	pop=5
dmax=8730		dmax=8532	
dmin=378		dmin=2358	
raz=8352	pop=6	raz=6174	pop=6
dmax=8532			
dmin=2358			
raz=6174	pop=7		

В программе 3 организована проверка четырехзначных чисел от 1000 до 2000.

Программа 3.1	Программа 3.2
<pre> N = 4 C = 6174: DIM A(N) FOR V = 1000 TO 2000 X = V: PRINT X; 'ОТДЕЛЕНИЕ ЦИФР ЧИСЛА 30 FOR I = 1 TO N D = X MOD 10 A(I) = D X = X \ 10 NEXT I Q = 1 'ПРОВЕРКА, ЧТО НЕ ВСЕ ЦИФРЫ ОДИНАКОВЫ FOR I = 2 TO N IF A(I)=A(1) THEN Q=Q+1 IF Q = N THEN 50 NEXT I 'СОРТИРОВКА ЦИФР ПО ВОЗРАСТАНИЮ FOR I = 1 TO N FOR J = 1 TO N IF A(I)<A(J) THEN Z= A(I): A(I) = A(J): A(J) = Z NEXT J NEXT I 'НАХОЖДЕНИЕ МАКСИ- МАЛЬНО И МИНИМАЛЬНО ВОЗМОЖНЫХ ЧИСЕЛ 'НАХОЖДЕНИЕ РАЗНОСТИ DMAX = 0: DMIN = 0 FOR I = 1 TO N DMAX=DMAX+A(I)*10^(I-1) DMIN=DMIN + A(I)*10^(N-I) NEXT I X=DMAX-DMIN: POP=POP+1 IF X=C OR POP=50 THEN 50 ELSE GOTO 30 50 PRINT POP; « «; POP = 0: NEXT V </pre>	<pre> Program P3_2; Uses crt; const n=4; const c=6174; label 30,50; var a:array [1..4] of integer; b,x,i,d,q,z,j,v,w,pop,dmax, dmin,raz:integer; begin for b:=1000 to 1200 do begin x:=b; write(x); {Отделение цифр числа} 30: for i:=1 to n do begin d:=x mod 10; a[i]:=d; x:=x div 10; end; q:=1; {Проверка, что не все цифры одинаковы} for i:=2 to n do begin if a[i]=a[1] then q:=q+1; if q=n then goto 50; end; {Сортировка цифр по возрас- танию} for j:=1 to n-1 do for i:=1 to n-j do begin IF a[i]>a[i+1] then begin z:=a[i]; a[i]:=a[i+1];a[i+1]:=z; end; end; {Нахождение максимально и минимально возможных чисел} {Нахождение разности} </pre>

	<pre> dmax:=0; dmin:=0;v:=1;w:=1; for i:=1 to n-1 do w:=w*10; for i:= 1 to n do begin dmax:=dmax+a[i]*v; dmin:= dmin+a[i]*w; v:=v*10;w:=w div 10; end; x:=dmax-dmin;pop:=pop+1; if (x=c) or (pop=50) then goto 50 else goto 30; 50:write('b=',b,' pop=',pop,' '); pop:=0; end; readkey; end. </pre>
--	---

Результат работы программы для чисел от 1000 до 1119 приведен в приложении 2.

Приложение 2

```

1000 5 1001 4 1002 3 1003 3 1004 7 1005 7 1006 7 1007 3
1008 3 1009 4 1010 4 1011 5 1012 4 1013 6 1014 4 1015 6
1016 6 1017 4 1018 6 1019 4 1020 3 1021 4 1022 3 1023 3
1024 7 1025 7 1026 7 1027 3 1028 3 1029 4 1030 3 1031 6
1032 3 1033 5 1034 2 1035 7 1036 1 1037 5 1038 6 1039 3
1040 7 1041 4 1042 7 1043 2 1044 3 1045 5 1046 3 1047 6
1048 5 1049 3 1050 7 1051 6 1052 7 1053 7 1054 5 1055 5
1056 4 1057 3 1058 2 1059 7 1060 7 1061 6 1062 7 1063 1
1064 3 1065 4 1066 5 1067 5 1068 7 1069 7 1070 3 1071 4
1072 3 1073 5 1074 6 1075 3 1076 5 1077 3 1078 2 1079 7
1080 3 1081 6 1082 3 1083 6 1084 5 1085 2 1086 7 1087 2
1088 5 1089 3 1090 4 1091 4 1092 4 1093 3 1094 3 1095 7
1096 7 1097 7 1098 3 1099 3 1100 4 1101 5 1102 4 1103 6
1104 4 1105 6 1106 6 1107 4 1108 6 1109 4 1110 5 1111
1112 5 1113 4 1114 6 1115 4 1116 6 1117 6 1118 4 1119 6

```

Рядом с четырехзначным числом справа – количество повторений алгоритма до достижения разности 6174. Работа программы для чисел от 1000 до 9999 показала, что данное явление справедливо для всех четырехзначных чисел, у которых не все цифры одинаковы.

4.3. СИММЕТРИЧНАЯ СУММА

Напишите любое число. Прибавьте к нему число с переставленными цифрами. То же самое проделайте с полученной суммой. Опыт показывает, что повторяя эти действия некоторое количество раз, вы непременно в каком-либо из результатов получите число, которое одинаково читается слева направо и справа налево.

Приведем несколько примеров:

38	139	48017
+ 83	+ 931	+ 71084
---	----	-----
121	1070	119101
	+ 0701	+ 101911
	-----	-----
	1771	221012
		+ 210122

		431134

Иногда для достижения симметричного результата необходимо проделать большее число шагов. Например, для числа 89 только 24-й шаг приводит к симметричному результату:

8 813 200 023 188.

Решение задачи о симметричных числах приведено в программе 4.

Программа 4.1	Программа 4.2
<pre> DEFLNG A-Z INPUT «ВВЕДИТЕ ЧИСЛО»; X DIM A(100), B(100) 10 N = 0 A\$ = STR\$(X) N = LEN(A\$) MNOR = X: MOBR = 0 'ВЫДЕЛЕНИЕ ЦИФР ЧИСЛА В </pre>	<pre> Program P4_2; Uses crt; label 10,100; var a,b: array[1..4] of integer; n,x,d,i,j,q,f,g,h,mobr,mnor,pop:longint; m:string; begin </pre>

<pre> МАССИВ FOR I = 1 TO N D = X MOD 10 A(I) = D X = X \ 10 NEXT I ' FOR I = N - 1 TO 1 ' PRINT A(I); ' NEXT I N = N - 1: J = N PRINT 'ФОРМИРОВАНИЕ МАССИВА С ПЕРЕСТАВЛЕННЫМИ ЦИФРА- МИ 'FOR I = 1 TO N ' B(I) = A(J) ' PRINT B(I); ' MOBR = MOBR+B(I)*10^(I-1) J = J - 1 NEXT I H = 0: F = 0: G = 0 PRINT 'ПРОВЕРКА СИММЕТРИЧНО- СТИ ЧИСЛА FOR I = 1 TO N F = A(I): G = B(I) IF F = G THEN H = H + 1 IF H = N THEN 100 F = 0: G = 0 NEXT I 'PRINT MNOR; MOBR 'ВЫЧИСЛЕНИЕ СУММЫ X = MNOR+MOBR: POP = POP + 1 IF POP = 100 THEN END GOTO 10 100 PRINT «РЕЗУЛЬТАТ СИМ- МЕТРИЧНОЕ ЧИСЛО»; MNOR; «ШАГОВ»; POP </pre>	<pre> writeln('Введите число x'); read(x); 10: n:= 0; str(x,m); n:=length(m); mnor:=x; mobr:=0; {Выделение цифр числа в массив} for i:= 1 to n do begin d:=x MOD 10; a[i]:=d; x:=x div 10; end; {for i:=n downto 1 do write(a[i]);writeln;} j:=n;q:=1; {Формирование массива с пе- реставленными цифрами} for i:=1 to n do begin b[i]:=a[j]; mobr:=mobr+b[i]*q; q:=q*10; j:=j-1; end; h:=0;f:=0;g:=0; { for i:=n downto 1 do write(b[i]);writeln;} {Проверка симметричности числа} for i:=1 to n do begin f:=a[i]; g:=b[i]; if f=g then h:=h+1; if h=n then goto 100; f:=0; g:=0; end; {write(mnor, mobr);} {Вычисление суммы} x:=mnor+mobr; pop:=pop+1; if pop=100 then exit; goto 10; </pre>
---	--

	<pre> 100: writeln('Результат симметричное число ', mpor,'шагов ', pop); readkey; end.</pre>
--	--

Ограничением решения задачи является то, что если сумма превышает значение 2147483647, программа не работает. Компьютер показывает переполнение. Программа 2 позволяет решить вопрос о приведенных выше ограничениях. Если необходимо фиксировать процессы формирования цифр и результаты сложения, необходимо удалить знак комментариев в строках, где они встречаются. Для увеличения диапазона решения задачи нужно изменить ограничения: pop – количество шагов, в программе максимальное число pop – 100, а также размерность массивов, в программе размерность равна 100. В приложениях 1 и 2 приведены результаты работы программы 5.

Программа 5.1	Программа 5.2
<pre> DIM A(100):DIM B(100): DIM C(100) INPUT «ВВЕДИТЕ ЧИСЛО»; A\$ 'ЗАНЕСЕНИЕ ЦИФР ЧИСЛА В МАССИВ A N = LEN(A\$) FOR I = 1 TO N W\$ = MID\$(A\$, I, 1) A(I) = VAL(W\$) NEXT I REM FOR I = 1 TO N REM PRINT A(I); REM NEXT I: 10 J = N: REM PRINT IF POP = 100 THEN END 'ФОРМИРОВАНИЕ МАССИВА B C ПЕРЕСТАВЛЕННЫМИ ЦИФРАМИ FOR I = 1 TO N B(I) = A(J) REM PRINT B(I);</pre>	<pre> Program P5_2; Uses crt; Clrscr; {очистка экрана} label 10,100,200; var n,i,j,k,pop,h,f,g:word; v,w:string; a,b,c:array[1..100] of byte; begin writeln('Введите число'); read(v); {Занесение цифр числа массив A} n:=length(v); for i:=1 to n do begin w:=copy(v,i,1); val(w,a[i],k); end; { for i:=1 to n do write(a[i]);writeln;} 10: j:=n; if pop=100 then exit; {Формирование массива B с переставленными цифрами}</pre>

<pre> J = J - 1 NEXT I: REM PRINT 'ПРОВЕРКА СИММЕТРИЧНОСТИ ЧИСЛА H = 0: F = 0: G = 0 FOR I = 1 TO N F = A(I): G = B(I) IF F = G THEN H = H + 1 IF H = N THEN 100 F = 0: G = 0: NEXT I ' СЛОЖЕНИЕ ЭЛЕМЕНТОВ МАССИВА K = 0 FOR I = N TO 1 STEP -1 C(I) = A(I) + B(I) + K IF C(I) >= 10 THEN K = 1 IF C(I) < 10 THEN K = 0 IF C(1) >= 10 THEN GOTO 200 IF C(I) >= 10 THEN C(I) = C(I) - 10 NEXT I FOR I = 1 TO N A(I) = 0: B(I) = 0 A(I) = C(I): C(I) = 0 REM PRINT A(I); NEXT I POP = POP + 1: GOTO 10 'ВЫВОД РЕЗУЛЬТАТА РАБОТЫ ПРОГРАММЫ 100 PRINT «РЕЗУЛЬТАТ СИММЕТРИЧНОЕ ЧИСЛО» FOR I = 1 TO N PRINT A(I); NEXT I: PRINT : PRINT «ШАГОВ»; POP: END 'ПОДПРОГРАММА ФОРМИРОВАНИЯ СУММЫ, ЕСЛИ ПРИ СЛОЖЕНИИ РАЗРЯДНОСТЬ СУММЫ УВЕЛИЧИВАЕТСЯ 200 A(1) = 1: A(2) = C(1) - 10: C(1) = 0 </pre>	<pre> for i:=1 to n do begin b[i]:=a[i]; { write(b[i]);} j:=j-1; end;{ writeln;} {Проверка симметричности числа} h:=0; f:=0; g:=0; for i:=1 to n do begin f:=a[i]; g:=b[i]; if f=g then h:=h+1; if h=n then goto 100; f:=0; g:=0; end; pop:=pop+1; {Сложение элементов массива} k:=0; for i:=n downto 1 do begin c[i]:=a[i]+b[i]+k; if c[i]>=10 then k:=1; if c[i]<10 then k:=0; if c[1]>=10 then goto 200; if c[i]>=10 then c[i]:=c[i]-10; end; for i:=1 to n do begin a[i]:=0; b[i]:=0; a[i]:=c[i]; c[i]:=0; {write(a[i]);} end; goto 10; {Вывод результата работы программы} 100: writeln('Результат симметричное число'); for i:=1 to n do write(a[i]);writeln; writeln('шагов', pop);exit; {Подпрограмма формирования суммы, если при сложении разрядность суммы увеличивается} 200: a[1]:=1; a[2]:=c[1]-10; c[1]:=0; </pre>
---	---

<pre> FOR I = 2 TO N A(I + 1) = 0 A(I + 1) = C(I): C(I) = 0 NEXT I: N = N + 1 FOR I = 1 TO N REM PRINT A(I); NEXT I: POP = POP + 1 GOTO 10 </pre>	<pre> for i:=2 to n do begin a[i+1]:=0; a[i+1]:=c[i];c[i]:=0; end; n:=n+1; { for i:=1 to n do write(a[i]);writeln;} goto 10; readkey; end. </pre>
---	---

Приложение 1

Введите число

89

результат симметричное число

8813200023188

шагов 24

Ведите число

675886

результат симметричное число

866181668

шагов 9

Введите число

689797

результат симметричное число

88388477488388

Приложение 2

**ТАБЛИЦА ДВУХЗНАЧНЫХ ЧИСЕЛ И ЧИСЛА ШАГОВ,
ПРИ КОТОРОМ ДОСТИГАЕТСЯ СИММЕТРИЧНОСТЬ**

Чис- ло	Ша- ги										
10	1	26	1	42	1	58	2	74	1	91	2
11	0	27	1	43	1	59	3	75	2	92	1
12	1	28	2	44	0	60	1	76	2	93	2
13	1	29	1	45	1	61	1	77	0	94	2
14	1	30	1	46	2	62	1	78	4	95	3
15	1	31	1	47	1	63	1	80	1	96	4

16	1	32	1	48	2	64	2	81	1	97	6
17	1	33	0	49	2	65	1	82	2	98	24
18	1	34	1	50	1	66	0	83	1	99	0
19	2	35	1	51	1	67	2	84	2		
20	1	36	1	52	1	68	3	85	2		
21	1	37	2	53	1	69	4	86	3		
22	0	38	1	54	1	70	1	87	4		
23	1	39	2	55	0	71	1	88	0		
24	1	40	1	56	1	72	1	89	24		
25	1	41	1	57	2	73	2	90	1		

4.4. УЧЕБНО-ИССЛЕДОВАТЕЛЬСКАЯ ФОРМА ОБУЧЕНИЯ

Учебные пособия главным образом подчинены задаче формирования знаний, умений и навыков и мало ориентированы на развитие творческих начал учащихся.

Учебное исследование – это не только познавательная деятельность учащихся под руководством учителя, но и метод обучения самой исследовательской деятельности. Приобщение к ней делает учебу производительным трудом, который состоит и в приобретении новых знаний, и в овладении новыми способами деятельности.

Проблема состоит в том, чтобы в процессе обучения смоделировать исследовательскую деятельность учащихся. Все определяется в постановке задачи, направленной на реализацию исследовательской деятельности. Умение подбора и постановки задач такого характера зависит от профессионального уровня руководителя (преподавателя или учителя). В различных изданиях приводится немало интересного материала с хорошим творческим «потенциалом».

Статья « Удивительные приключения периодических дробей» (Квант. 1989. № 8) определила выбор темы нашего исследования. Исследовательская деятельность учащихся связана с проверкой некоторых «загадок», связанных с периодическими дробями.

Изучение исходного материала выдвигает решение следующих задач:

- определение периода дроби;
- определение длины периодических дробей;
- перестановка цифр периода дробей.

Рассмотрим решение перечисленных задач.

Задача 1. Определение периода дроби. Для решения задачи определения периода дроби учащимся предлагается ознакомиться с алгоритмом и программой решения данной задачи (№ 84.4) по книге «Московские олимпиады по программированию» [3]. Ниже приведена программа решения данной задачи для периода $1/p$. Здесь цифры периода дроби, длина периода и знаменатель дроби $1/p$ занесены в файл данных, чтобы эти данные в дальнейшем использовать в других программах.

Программа 6.1	Программа 6.2
<pre>' ОПРЕДЕЛЕНИЕ ДЛИНЫ ПЕРИОДА ДРОБИ OPEN «PERIOD» FOR OUTPUT AS #1 OPEN «DLINA « FOR OUTPUT AS #2 OPEN «ZNAMEN» FOR OUTPUT AS #3 INPUT «ВЕДИТЕ ЗНАЧЕНИЕ P»; P PRINT «ZNAM»; P PRINT #3, P CLOSE #3 L = 0 M = 1 MOD P FOR K = 2 TO P M = 10 * M MOD P NEXT K J = M 80 M = 10 * M L = L + 1 PRINT M \ P; PRINT #1, M \ P; M = M MOD P IF M <> J THEN 80 PRINT «DLINA «; L PRINT #2, L CLOSE #1 CLOSE #2 120 END</pre>	<pre>Program P6_2; Uses crt; Label vt; var dlina:text; period:text; znamen:text; p,m,k,j,i,z:integer; begin assign(znamen,'ZNAMEN.DAT'); rewrite(znamen); writeln('vvedi znachenie p'); readln(p); write(znamen,p); close(znamen); i:=0; m:=1 mod p; for k:=2 to p do m:=10*m mod p; j:=m; assign (period,'PERIOD.DAT'); rewrite(period); vt: m:=10*m; i:=i+1;z:=trunc(m/p); write(period,z); m:=m mod p; if m<>j then goto vt; close(period); assign(dlina,'DLINA.DAT'); rewrite(dlina); writeln(i); write(dlina,i);</pre>

	close(dlina); readkey; end.
--	-----------------------------------

Задача 2. Определение длины периода дроби. При решении задачи учащимся предстоит удостовериться в следующих утверждениях:

С точки зрения соотношения между L -длинами периода дроби $1/p$ и самим p все простые числа подразделяются на три категории:

1) «полнопериодные» простые, у которых длина периода на 1 меньше знаменателя: 7 ($L=6$), 17 ($L=16$), 19 ($L=18$), 23 ($L=22$), 29 ($L=28$) и т. д.;

2) простые числа с нечетной длиной периода: 3 ($L=1$), 31 ($L=15$), 37 ($L=3$) и т. д.;

3) «неполнопериодные» простые с четной длиной периода: 11 ($L=2$), 13 ($L=6$), 73 ($L=8$), 89 ($L=44$), 101 ($L=4$) и т. д.

Было обнаружено достаточно устойчивое отношение численности этих групп в пропорции 9: 8: 7. Решение поставленной задачи начнем с поиска простых чисел в заданном интервале. Программа 7 осуществляет поиск простых чисел в интервале от 1 до 9999 и запись простых чисел в файл данных «prost».

Программа 7.1	Программа 7.2
<pre>' Простые числа в интервале от 1 до 9999 OPEN «PROST» FOR OUTPUT AS #6 FOR M = 1 TO 9999 STEP 2 FOR K = 3 TO INT(SQR(M)) + 1 STEP 2 A = M MOD K IF A = 0 GOTO 70 NEXT K PRINT #3, M; 70 NEXT M CLOSE #6 END</pre>	<pre>Program P7_2; Uses crt; Clrscr; {очистка экрана} label 70; var prost: text; m, k, a: integer; begin assign (prost, 'PROST.DAT'); rewrite (prost); m:=3; write(prost,'3 '); repeat k:=3; repeat a:=m mod k; if a=0 then goto 70; k:=k+2; until k>round(sqrt(m))+1; write(prost,' ');</pre>

	<pre> write(prost,m); 70: m:=m+2; until m>9999; close (prost); readkey; end.</pre>
--	---

Программа 8 реализует решение задачи 2. В файл данных «dlina» записываются значения простых чисел и длины периодов. В программе следующие обозначения: K1 – количество простых чисел, удовлетворяющих условию 1-й категории, K2 – 2-й категории, K3 – 3-й категории, Z – количество простых чисел в указанном интервале.

Программа 8.1	Программа 8.2
<pre> OPEN «DLINAPR» FOR OUTPUT AS #4 OPEN «PROST» FOR INPUT AS #6 WHILE NOT (EOF(6)) INPUT #6, N M = 1: L = 0 M = M MOD N FOR K = 2 TO N M = 10 * M MOD N NEXT K J = M 80 M = 10 * M L = L + 1 M = M MOD N IF M <> J THEN 80 PRINT #4, «N=»; N; «L=»; L, Z = Z + 1 IF N-1=L THEN K1 = K1 + 1 IF INT(L/2)<>L/2 THEN K2=K2+1 IF INT(L/2)=L/2 AND N-1<>L THEN K3=K3+1 WEND CLOSE #6 CLOSE #4 PRINT «K1=»; K1,»K2=»;K2,</pre>	<pre> Program P8_2; Uses crt; Label vt; var dlinapr, prost: text; n,k1, k2, k3,i,k,m,j,z,q: integer; begin assign (dlinapr, 'DLINAPR.DAT'); rewrite (dlinapr); assign(prost,'PROST.DAT'); reset (prost); while not EOF(prost) do begin read(prost,n); write('n=',n, ' '); begin m:=1; j:=0;i:=0; m:=m mod n; for k:=2 to n do m:= 10*m mod n; j:=m; vt: m:=10*m; i:=i+1; m:= m mod n; if m<>j then goto vt; write(dlinapr,'n=',n,'i=',i); z:=z+1; write('i=',i, ' '); if (n-1=i) then k1:=k1+1;</pre>

<pre> «K3=«;K3 PRINT «Z=«; Z END </pre>	<pre> if i mod 2<>0 then k2:=k2+1; if (i mod 2=0) and (n-1<>i) then k3:=k3+1; end; end; close(prost); close(dlinapr); writeln('k1=',k1,' k2=',k2,' k3=',k3); writeln('z=',z); readkey; end. </pre>
---	--

Полученные значения для интервала простых чисел от 1 до 9999:
K1=467, K2=410, K3=352, Z=1229.

Исследования с длинами периодов можно продолжить. Например, в (1) доказывается, что для $1/p$ длина периода l есть делитель $p-1$. Но спрашивается, какие?

Задача 3. Перестановка цифр периода дроби.

Проверяем утверждение:

Если число N , состоящее из цифр периода дроби $1/p$, где p – простое, умножить на 2, 3, 4, ..., $p-1$, то они получаются из числа N перестановкой цифр.

Программа 9 позволяет получить цифры числа N , умноженные на 2, 3, 4, ..., p . Полученные результаты выводятся на экран и сохраняются в файле данных для дальнейшего исследования приключения периодических дробей. В программе 9 используется решение задачи 1.

Программа 9.1	Программа 9.2
<pre> OPEN «DLINA» FOR INPUT AS #2 OPEN «UM-PERIOD» FOR OUTPUT AS #5 INPUT #2, R PRINT «R»; R CLOSE #2 DIM D(R): DIM C(R) OPEN «PERIOD» FOR INPUT AS #1 T = 0 WHILE NOT (EOF(1)) </pre>	<pre> Program P9_2; Uses crt; label pt,dv; type massiv=array [1..100] of word; type dvmassiv=array[1..100,1..100] of word; var umperiod,dlina,period:text; znamen:text; d:massiv; c:dvmassiv; t,k,j,i,r,x,n,g,q:integer; begin </pre>

<pre> T = T + 1 INPUT #1, D(T) WEND CLOSE #1 OPEN «ZNAMEN» FOR INPUT AS #3 INPUT #3, N CLOSE #3 K = 0 FOR J = 1 TO N FOR I = R TO 1 STEP -1 C(I) = D(I) * J + K IF C(I) < 10 THEN K = 0 IF C(I) >= 10 THEN K = (C(I) - C(I) MOD 10) / 10 IF C(I) >= 10 THEN C(I) = C(I) MOD 10 IF C(1) >= 10 THEN GOTO 200 NEXT I 50 FOR I = 1 TO R PRINT C(I); PRINT #5, C(I); NEXT I: PRINT GOTO 60 200 C(2) = C(1) MOD 10: C(1) = (C(1) - C(1) MOD 10) / 10 FOR I = 2 TO R C(I + 1) = C(I) NEXT I: X = R + 1 R = X GOTO 50 PRINT 60 NEXT J CLOSE #5 END </pre>	<pre> assign(dlina,'DLINA.DAT'); reset(dlina); while not EOF(dlina) do begin read(dlina,r);writeln('dlina r=',r); end; close(dlina); assign(znamen,'ZNAMEN.DAT'); reset(znamen); while not EOF(znamen) do begin read(znamen,n); writeln('znamen n=',n); end; close(znamen); assign(period,'PERIOD.DAT'); reset(period);write('period='); while not EOF(period) do begin for t:=1 to r do begin read(period,g);d[t]:=g; write(d[t]); end;writeln; end; close(period); as- sign(umperiod,'UMPERIOD.DAT'); rewrite(umperiod); k:=0; for j:=1 to n do begin for i:=r downto 1 do begin c[j,i]:=d[i]*j+k;q:=c[j,i]; if c[j,i]<10 then k:=0; if c[j,i]>=10 then k:=q div 10; if c[j,i]>=10 then c[j,i]:=q mod 10; end; if c[j,1]>=10 then goto dv; goto pt; dv: c[j,2]:=c[j,j] mod 10; c[j,1]:=c[j,1] div 10; </pre>
---	--

	<pre> for i:=2 to r do c[j,i+1]:=c[j,i]; r:=r+1; pt: for i:=1 to r do begin write(' ', c[j,i]); write(umperiod, c[j,i]); end;writeln; end; close (umperiod); readkey; end.</pre>
--	--

Далее учащимся предлагается:

- самостоятельное исследование утверждений в [3], которые не вошли в данный материал;
- самостоятельное экспериментирование со значениями периодов дробей;
- выдвижение собственных гипотез;
- экспериментальная проверка выдвинутых гипотез.

4.5. ЗАДАЧИ О КРОСС-СУММАХ

Пересекающиеся ряды чисел с одинаковыми суммами называют кросс-суммами. Направление расположения чисел указывается заранее, чаще всего вдоль линий какой-нибудь симметричной фигуры; числа, помещаемые в точках пересечения двух или нескольких линий, включаются в сумму чисел вдоль каждой из этих линий. Условия кросс-сумм назовем «магическими». Одинаковые суммы, удовлетворяющие «магическим условиям» – константами «магических фигур».

Рассмотрим следующую задачу.

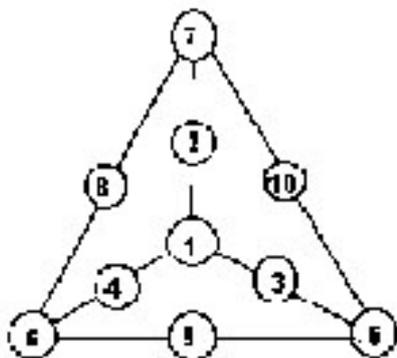


Рис. 30

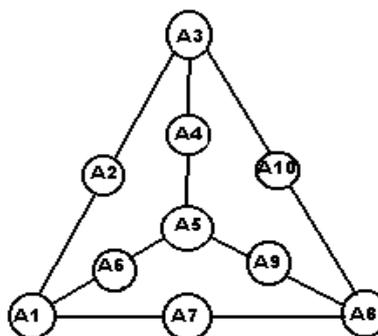


Рис. 31

Задача 1. В 10 кружках, расположенных вдоль сторон и радиусов равностороннего треугольника (рис.30), можно разместить десять натуральных чисел от 1 до 10 так, что сумма чисел, расположенных по сторонам и углам каждого из трех маленьких треугольников, будет равна одной и той же сумме K (константе кросс-суммы для данной фигуры). Например, на рис.31 показан один из способов расположения десяти чисел $1+2+7+8+6+4 = 28$
 $1+4+6+9+5+3 = 28$ $1+3+5+10+7+2=28$. Здесь константа кросс-суммы равна 28.

Определить константы и количество треугольников, удовлетворяющих выше указанным «магическим условиям». По вопросу существования и оценки констант «магических» фигур. Возьмем «магическое» условие удовлетворяющее «магической» фигуре на рис.31.

$$a_1+a_2+a_3+a_4+a_5+a_6=K$$

$$a_3+a_4+a_5+a_9+a_8+a_{10}=K$$

$$a_1+a_6+a_5+a_9+a_8+a_7=K$$

Сложим эти равенства:

$$3K=a_1+a_2+a_3+a_4+a_5+a_6+a_3+a_4+a_5+a_9+a_{10}+a_1+a_6+a_5+a_9+a_8+a_7=$$

$$= a_1+a_2+a_3+a_4+a_5+a_6+a_7+a_8+a_9+a_{10}+ 2a_5+a_1+a_3+a_4+a_6+a_8+a_9;$$

$a_1+a_2+a_3+a_4+a_5+a_6+a_7+a_8+a_9+a_{10}$ – дают сумму чисел от 1 до 10 и она равна 55. Тогда $3K=55+2a_5+a_1+a_3+a_4+a_6+a_8+a_9$.

Найдем минимальную константу. Чтобы константа была минимальной, должно $a_5=1$ и $a_1, a_3, a_4, a_6, a_8, a_9$ принимали бы значения 2,3,4,5,6,7.

$$55+2*1+2+3+4+5+6+7=84$$

$$3K=84, \text{ отсюда } K=28.$$

Найдем максимальную константу. Чтобы константа была максимальной, должно $a_5=10$ и $a_1, a_3, a_4, a_6, a_8, a_9$ принимали бы значения 9,8,7,6,5,4.

$$55+2*10+9+8+7+6+5+4=114$$

$$3K=114, \text{ отсюда } K=38.$$

Получили $K_{\min}=28$, $K_{\max}=38$, остальные $28 < K < 38$.

Решение задач о кросс-суммах сводится к рассмотрению перестановок первых N натуральных чисел с проверкой «магических» свойств перечисленных фигур. Воспользуемся алгоритмом, приведенным в методическом пособии для учителей и преподавателей средних учебных заведений «Изучение основ информатики и вычислительной техники» под ред. А.П. Ершова и В.М. Монахова (с.199, упр.33). Приведем данный алгоритм.

```

алг упражнение 33
нач цел i, цел таб A[1:100], C[1:100], лит признак
  для i от 1 до 100
    нц
      C[i]
    кц
  A[1]:=0; i:=1;
  признак:=«продолжить»
  пока признак=«продолжить»
    нц
      A[i]:=F[i]+1
      если A[i]<=100
        то если C[A[i]]=0
          то C[A[i]]:=1
          если i<100
            то A[i+1]:=0
            i:=i+1
          иначе печать (A)
            C[A[i]]:=0
        все
      все
    иначе если i>1
      то i:=i-1
      C[A[i]]:=0
    иначе признак:=«закончить»
  все
кц
кон

```

В данном алгоритме используется вспомогательный алгоритм <ПЕЧАТЬ(A)>, печатающий элементы таблицы. Для решения нашей задачи мы использовали вспомогательный алгоритм проверки «магических» условий. Если выполняются «магические» условия $a_1+a_2+a_3+a_4+a_5+a_6 = a_3+a_4+a_5+a_9+a_8+a_{10} = a_1+a_6+a_5+a_9+a_8+a_7$, то выводятся на печать расположения чисел в данном треугольнике и константы. Реализация данного алгоритма приведена в программе 10.

Программа 10.1	Программа 10.2
<pre> 6 N = 10 10 DIM A(N + 1) 20 DIM C(N) 30 FOR I = 1 TO N 40 C(I) = 0 50 NEXT I 60 A(1) = 0 70 I = 1 80 A(I) = A(I) + 1 90 IF A(I) <= N THEN GOTO 110 ELSE GOTO 300 110 IF C(A(I)) = 0 THEN GOTO 130 120 GOTO 80 130 C(A(I)) = 1 140 IF I < N THEN GOTO 160 ELSE GOTO 200 160 A(I + 1) = 0 170 I = I + 1 180 GOTO 80 200 A = 0: C = 0: B = 0 201 H=H+1 203 IF H=1814401 THEN GOTO 340 208 A = A(1) + A(2) + A(3) + A(4) + A(5) +A(6) 209 C = A(5) + A(4) + A(3) + A(10) + A(9) + A(8) 210 B = A(1) + A(6) + A(5) + A(9) + A(8) + A(7) 211 IF A<28 AND A>38 THEN GOTO 80 214 IF A = B AND B = C THEN GOTO 1000 227 C(A(I)) = 0 230 GOTO 80 300 IF I > 1 THEN GOTO 310 ELSE GOTO 340 310 I = I - 1 320 C(A(I)) = 0 330 GOTO 80 340 PRINT </pre>	<pre> Program P10_2; Uses crt; const n=10; label 80,110,130,160,200,300,310,340,10 00; var c,a:array[1..n+1] of byte; i,d,b,s,k,z,w,x,y,l,o,h,u,q,j,g,f:integer; begin for i:=1 TO n do c[i]:=0; a[1]:=0; i:=1; 80: a[i]:=a[i]+1; if a[i]<=n then goto 110 else goto 300; 110:if c[a[i]]=0 then goto 130; goto 80; 130:c[a[i]]:=1; if i<n then goto 160 else goto 200; 160: a[i+1]:=0; i:=i+1; goto 80; 200: d:=0;s:=0;b:=0; d:=a[1]+a[2]+a[3]+a[4]+a[5]+a[6]; s:=a[5]+a[4]+a[3]+a[10]+a[9]+a[8]; b:=a[1]+a[6]+a[5]+a[9]+a[8]+a[7]; if (d<28) and (d>38) then goto 80; if (d=b) and (b=s) then goto 1000; c[a[i]]:=0; goto 80; 300: if i>1 then goto 310 else goto 340; 310: i:=i-1; c[a[i]]:=0; goto 80; 340:write(k,' ',z,' ',w,' ',x,' ',y,' ',l,' ',o,' ',u,' ',q,' ',j,' ',g,' ',f); exit; 1000: c[a[i]]:=0; k:=k+1; { kol-vo mag} {write(a[1],a[2],a[3],a[3],a[4],a[5],a[6], </pre>

<pre> K;Z;W;X;Y;L;O;U;Q;J;G;F: END 1000 C(A(I)) = 0 1010 K = K + 1 1100 IF A=28 THEN Z=Z+1 1110 IF A=29 THEN W=W+1 1120 IF A=30 THEN X=X+1 1200 IF A=31 THEN Y=Y+1 1210 IF A=32 THEN L=L+1 1220 IF A=33 THEN O=O+1 1221 IF A=34 THEN U=U+1 1222 IF A=35 THEN Q=Q+1 1230 IF A=36 THEN J=J+1 1240 IF A=37 THEN G=G+1 1250 IF A=38 THEN F=F+1 1400 PRINT «CONST= «; A 1500 GOTO 80 </pre>	<pre> a[7],a[8],a[8],a[9],a[10]);} writeln; writeln('CONST=',d); if d=28 then z:=z+1; if d=29 then w:=w+1; if d=30 then x:=x+1; if d=31 then y:=y+1; if d=32 then l:=l+1; if d=33 then o:=o+1; if d=34 then u:=u+1; if d=34 then q:=q+1; if d=36 then j:=j+1; if d=37 then g:=g+1; if d=38 then f:=f+1; goto 80; readkey; end. </pre>
--	---

Решение задачи 1. Всего 6528 треугольников, удовлетворяющих «магическим» условиям; с константой 28 – 96 треугольников, 29 – 192, 30 – 480, 31 – 864, 32 – 1248, 33 – 768, 34 – 1248, 35 – 864, 36 – 480, 37 – 192, 38 – 96.

Приведем несколько задач о «кросс-суммах».

Задача 2. Расположить первые последовательно натуральные числа в форме треугольника так, чтобы образовалась одна и та же сумма вдоль каждой из его сторон (рис.32а).

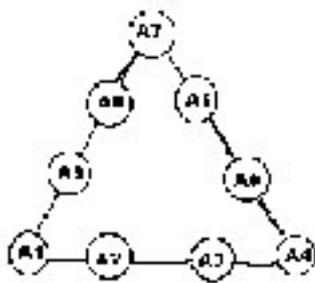


Рис.32а

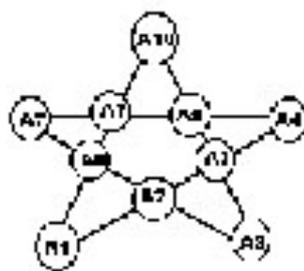


Рис.32б

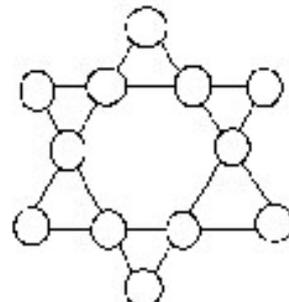


Рис.32в

Задача 3. Расставьте числа от 1 до 10 по кружкам пятиконечной звездочки (рис. 32б) так, чтобы сумма чисел в четырех кружках каждого из пяти лучей была равна $a_1+a_2+a_3+a_4 = a_4+a_5+a_6+a_7 = a_7+a_8+a_2+a_9 = a_9+a_3+a_5+a_{10} = a_{10}+a_6+a_8+a_1$.

Задача 4. Расставить числа от 1 до 12 по кружкам шестиконечной звездочки (рис. 32в) так, чтобы сумма чисел в четырех кружках каждого из шести лучей была равна.

Задача 5. На рис. 32с своеобразный орнамент, состоящий из 16 маленьких треугольников.

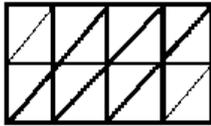


Рис.32с

Некоторые группы из соседних четырех маленьких треугольников образуют большие треугольники. Нетрудно подметить шесть больших треугольников, «вплетенных» один в другой. Впишите в каждый маленький треугольник орнамента одно из целых чисел от 1 до 16 (не повторяя их) таким образом, чтобы сумма чисел в любом из шести больших треугольников была равна.

4.6. ЗАНИМАТЕЛЬНЫЕ ЭТЮДЫ

Основная задача учителя – поддержание интереса к изучаемому предмету. Одним из способов является подбор занимательных задач. Говоря о занимательности, мы имеем в виду занимательность формы или содержания заданий, в которые они облакаются. Педагогически оправданная занимательность имеет целью привлечь внимание к заданиям, активизировать их мыслительную деятельность, работать с литературой. Ниже приведенные этюды должны, на наш взгляд, решить поставленные задачи. Программы написаны на Qbasic и Turbo Pascal.

Этюд 1. Возьмите любое натуральное число. Найдите сумму квадратов цифр этого числа. То же самое сделайте с полученной суммой. Повторите эти действия многократно. Что получилось?

Программа 11 реализует приведенный алгоритм.

Программа 11.1	Программа 11.2
<pre>'prog 1.1 CLS DEFLNG A-Z INPUT «Введите исследуемое число»; z INPUT «Введите количество шагов»; n m = z</pre>	<pre>Program P11_2; Uses crt; VAR z,n,m,sum,c,i:INTEGER; BEGIN Clrscr; {очистка экрана} WRITELN('Введите исследуемое число');</pre>

<pre> FOR i = 1 TO n PRINT m; «->»; sum = 0 DO WHILE m > 0 c = m MOD 10 sum = sum + c ^ 2 m = m \ 10 LOOP m = sum NEXT i END </pre>	<pre> READ(z); WRITELN('Введите количество шагов'); READ(n); m:= z; FOR i:= 1 TO n DO BEGIN WRITE(m,'->'); sum:= 0; WHILE m > 0 DO BEGIN c:= m MOD 10; sum:= sum + c * c; m:= m DIV 10; END; m:= sum; END; readkey; END. </pre>
---	---

Результат работы показывает, что с некоторого шага возникает периодически повторяющаяся числовая последовательность: 145, 42, 20, 4, 16, 37, 58, 89 либо 1. Программа 12 определяет, с какого шага возникает указанная последовательность.

Программа 12.1	Программа 12.2
<pre> CLS DEFLNG A-Z INPUT «ВВЕДИТЕ ИССЛЕДУЕМОЕ ЧИСЛО»; Z M = Z DO PRINT M; «->»; SUM = 0 DO WHILE M > 0 C = M MOD 10 SUM = SUM + C ^ 2 M = M \ 10 LOOP K = K + 1 M = SUM </pre>	<pre> Program P12_2; Uses crt; VAR z,n,m,sum,c,i,k:INTEGER; BEGIN Clrscr; {очистка экрана} WRITELN('Введите исследуемое число'); READ(z); m:= z; REPEAT WRITE(m,'->'); sum:= 0; WHILE m > 0 DO BEGIN c:= m MOD 10; </pre>

<pre> LOOP UNTIL M = 145 OR M = 1 PRINT PRINT «K=»; K END </pre>	<pre> sum:= sum + c*c; m:= m DIV 10; END; k:= k + 1; m:= sum; UNTIL (m = 145) OR (m=1); WRITELN('k=',k); readkey; END. </pre>
--	---

Этюд 2. Возьмите какое-нибудь натуральное число. Если число четное, то разделите его на два. Если оно нечетное, то прибавьте к нему 101. Снова: если полученное число N четное, то берем $N/2$, а если нечетное, то берем $N+101$. И так далее. Что получается?

Программа 13 реализует данный алгоритм.

Программа 13.1	Программа 13.2
<pre> 'PROG 3.1 CLS DEFLNG A-Z INPUT «ВВЕДИТЕ ИССЛЕДУЕМОЕ ЧИСЛО»; Z INPUT «ВВЕДИТЕ КОЛИЧЕСТВО ШАГОВ»; N FOR I = 1 TO N PRINT Z; «->»; IF Z MOD 2 = 0 THEN Z = Z / 2 ELSE Z = Z + 101 NEXT I END </pre>	<pre> PROGRAM P13_2; USES CRT; VAR Z,N,I:INTEGER; BEGIN WRITELN('ВВЕДИТЕ ИСС- ЛЕДУЕМОЕ ЧИСЛО'); READ(Z); WRITELN('ВВЕДИТЕ КОЛИ- ЧЕСТВО ШАГОВ'); READ(N); FOR I:= 1 TO N DO BEGIN WRITE(Z,'->'); IF Z MOD 2=0 THEN Z:=Z DIV 2 ELSE Z:= Z + 101 END; READKEY; END. </pre>

Работа программы показывает, что с некоторого шага возникает периодически повторяющаяся числовая последовательность, начинающаяся числами 102, 51, ... и заканчивающаяся числами

...128, 64, 32, 16, 8, 4, 2, 1. Программа 14 определяет, с какого шага возникает указанная последовательность.

Программа 14.1	Программа 14.2
<pre>'PROG 4.1 CLS DEFLNG A-Z INPUT «ВВЕДИТЕ ИССЛЕДУЕМОЕ ЧИСЛО»; Z K = 0 WHILE Z <> 102 PRINT Z; «->»; IF Z MOD 2 = 0 THEN Z = Z / 2 ELSE Z = Z + 101 K = K + 1 WEND PRINT «K=»; K END</pre>	<pre>Program P14_2; Uses crt; VAR Z,K:INTEGER; BEGIN WRITELN('ВВЕДИТЕ ИССЛЕДУЕМОЕ ЧИСЛО'); READ(Z); WHILE Z<>102 DO BEGIN WRITE(Z,'->'); IF Z MOD 2 = 0 THEN Z:= Z DIV 2 ELSE Z:= Z + 101; K:=K+1; END; WRITELN('K=',K); READKEY; END.</pre>

Этюд 3. Числовой фриз. Напишите два ряда единиц и соедините их зигзагом, как в приводимом примере (знаками # отмечены места, куда в дальнейшем будут вписываться числа):

```

1 * 1 * 1 * 1 * 1 * 1 * 1 * 1 * 1 * 1 * 1 * 1 * 1 *
* 1 * # * # * # * # * # * # * # * # * # * #
* * 1 * # * # * # * # * # * # * # * # * # *
* * * 1 * # * # * # * # * # * # * # * # * #
* * 1 * # * # * # * # * # * # * # * # * # *
* 1 * # * # * # * # * # * # * # * # * # * #
* * 1 * # * # * # * # * # * # * # * # * # *
* * * 1 * 1 * 1 * 1 * 1 * 1 * 1 * 1 * 1 * 1 * 1 *

```

А теперь начнем заполнять места, которые отмечены знаками # по «правилу ромба»: для четырех соседних чисел, расположенных ромбом, –

$$\begin{array}{ccc}
 & C & \\
 3 & & B \\
 & Ю &
 \end{array}$$

по формуле $B = (C * Ю + 1) / 3$.

Ниже приведенная программа 15 реализует описанную выше арифметическую игру.

Программа 15.1	Программа 15.2
<pre> CLS DIM F\$(24, 8) FOR I = 1 TO 24 FOR J = 1 TO 8 IF I MOD 2=1 AND J MOD 2=1 THEN F\$(I, J) = «#» IF I MOD 2=0 AND J MOD 2=1 THEN F\$(I, J) = «*» IF I MOD 2=1 AND J MOD 2=0 THEN F\$(I, J) = «*» IF I MOD 2=0 AND J MOD 2=0 THEN F\$(I, J) = «#» NEXT J NEXT I LOCATE 1, 9 PRINT «ЧИСЛОВОЙ ФРИЗ» FOR I = 1 TO 24 STEP 2 F\$(I,1)=«1» F\$(I+1,8)=«1» NEXT I F\$(2, 2) = «1» F\$(3, 3) = «1» F\$(4, 4) = «1» F\$(3, 5) = «1» F\$(2, 6) = «1» F\$(3, 7) = «1» F\$(1, 7) = «*» F\$(1, 3) = «*» F\$(2, 4) = «*» F\$(1, 5) = «*» F\$(2, 8) = «*» FOR I = 1 TO 24 FOR J = 1 TO 8 LOCATE J*2+1, I*3+2 PRINT F\$(I, J) NEXT J NEXT I LOCATE 20, 9 PRINT «ФОРМУЛА ФРИЗА В , = </pre>	<pre> PROGRAM P15_2; USES CRT; VAR F:ARRAY[1..24,1..8] OF STRING; I,J,K:BYTE; N:CHAR; B,S,U,Z:WORD; BEGIN CLRSCR; FOR I:= 1 TO 24 DO FOR J:= 1 TO 8 DO BEGIN IF (I MOD 2=1) AND (J MOD 2=1) THEN F[I, J]:='#'; IF (I MOD 2 = 0) AND (J MOD 2=1) THEN F[I, J]:='*'; IF (I MOD 2=1) AND (J MOD 2 = 0) THEN F[I, J]:='*'; IF (I MOD 2=0) AND (J MOD 2 = 0) THEN F[I, J]:='#'; END; GOTOXY(9,1); WRITELN('Ч И С Л О В О Й Ф Р И З '); FOR I:= 0 TO 11 DO BEGIN K:=2*I+1; F[K,1]:='1'; F[K + 1, 8]:='1'; END; F[2, 2]:='1'; F[3, 3]:='1'; F[4, 4]:='1'; F[3, 5]:='1'; F[2, 6]:='1'; F[3, 7]:='1'; F[1, 7]:='*'; F[1, 3]:='*'; F[2, 4]:='*'; F[1, 5]:='*'; F[2, 8]:='*'; FOR I:= 1 TO 24 DO BEGIN FOR J:= 1 TO 8 DO </pre>

<pre> (C*Ю+1)/3» PRINT «ДЛЯ ПРОДОЛЖЕНИЯ НАЖМИТЕ НА ENTER» DO WHILE N\$ = «» N\$ = INKEY\$ LOOP IF N\$ = CHR\$(13) THEN FOR I = 1 TO 23 FOR J = 1 TO 8 IF F\$(I, J) = «#» THEN S = VAL(F\$(I - 1, J - 1)) U = VAL(F\$(I - 1, J + 1)) Z = VAL(F\$(I - 2, J)) F\$(I, J) = STR\$((S*U+1)/Z) END IF LOCATE J*2+1, I*3+2 PRINT F\$(I, J) NEXT J NEXT I END IF END </pre>	<pre> BEGIN GOTOXY(I*3+2, J*2+1); WRITE(F[I, J]); END; WRITELN; END; GOTOXY(9, 20); WRITELN('ФОРМУЛА ФРИЗА В = C * Ю + 1) / 3'); WRITELN('ДЛЯ ПРОДОЛЖЕ- НИЯ НАЖМИТЕ ENTER'); WHILE N<>#13 DO N:= READKEY; IF N=#13 THEN BEGIN FOR I:= 1 TO 23 DO BEGIN FOR J:= 1 TO 8 DO BEGIN IF F[I, J]='#' THEN BEGIN VAL(F[I-1, J-1, S, B]); VAL(F[I-1, J+1, U, B]); VAL(F[I - 2, J], Z, B); STR((S*U+1) DIV Z, F[I, J]); END; GOTOXY(I*3+2, J*2+1); WRITE(F[I, J]); END; WRITELN; END; END; READKEY; END. </pre>
--	---

В результате выполнения нашей программы получится числовой фриз (фризом или бордюром называют периодически повторяющийся рисунок на бесконечной полосе).

```

1 * 1 * 1 * 1 * 1 * 1 * 1 * 1 * 1 * 1 * 1 * 1 * 1
* 1 * 2 * 2 * 4 * 2 * 1 * 3 * 2 * 4 * 1
* * 1 * 3 * 7 * 7 * 1 * 2 * 5 * 7 * 3 * 1
* * * 1 * 10 * 12 * 3 * 1 * 3 * 17 * 5 * 2 * 1

```

* * 1 * 3 * 17 * 5 * 2 * 1 * 10 * 12 * 3 * 1
 * 1 * 2 * 5 * 7 * 3 * 1 * 3 * 7 * 7 * 1
 * * 1 * 3 * 2 * 4 * 1 * 2 * 2 * 4 * 2 * 1
 * * * 1 * 1 * 1 * 1 * 1 * 1 * 1 * 1 * 1 * 1

Наблюдаются несколько любопытных закономерностей:

1. Все деления выполняются нацело, т.е. все числа в таблице будут натуральными.
2. В каждой строчке вновь появляется единица.
3. После этого строчки можно не продолжать, потому что новые единицы образуют точно такой же зигзаг, как тот, с которого начали, только перевернутый вверх ногами, так что полная таблица будет периодической.

Нечто похожее происходит, если единицы заменить нулями, умножение заменить сложением, а деление – разностью. Можно попробовать и другие зигзаги. После разбора этюдов учитель рекомендует теоретически обосновать полученные результаты. Указывает литературу, в которой они могут найти ответы на поставленные задачи.

ГЛАВА 5. РЕШЕНИЕ МАТЕМАТИЧЕСКИХ ЗАДАЧ

5.1. РЕШЕНИЕ МАТЕМАТИЧЕСКИХ ЗАДАЧ МЕТОДОМ ОБОБЩЕНИЯ И АНАЛОГИИ

Слово аналогия в переводе с греческого языка означает соответствие, сходство. Аналогия – весьма эффективный эвристический инструмент познания. Применение аналогии предполагает следующие действия: построение аналогов различных заданных объектов и отношений; нахождение соответственных элементов в аналогичных предложениях; составление предложений или задач, аналогичных данным; проведение рассуждений по аналогии.

Под применением метода обобщения следует понимать составление и решение задач, порожденных исходной задачей. Применение обобщения содержит следующие действия: замену части данных в исходной задаче другими данными без замены заключения задачи; при обобщении данных или искомым условий задачи; добавление новых условий при сохранении исходных данных.

Реализацию этих методов рассмотрим на конкретных примерах.

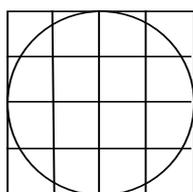


Рис. 33

Задача 1. На клетчатой бумаге нарисовали окружность целого радиуса R с центром на пересечении линий. Найти K – количество клеток, целиком лежащих внутри этой окружности (рис. 33). Обобщим задачу.

На клетчатой бумаге нарисована криволинейная трапеция. Найти: K – количество клеток, целиком лежащих внутри этой криволинейной трапеции. Для решения этой задачи мысленно проведем оси координат так, чтобы они совпадали с линиями на бумаге и ось Ox ограничивала криволинейную трапецию снизу. Примем длину стороны клетки за единицу масштаба осей координат. Определим функцию, задающую криволинейную трапецию и линии, ограничивающие трапецию.

Решение задачи сводится к алгоритму вычисления площади криволинейной трапеции методом прямоугольников. Ниже приведен алгоритм, суммирующий площади «левых» прямоугольников.

алг метод прямоугольников (вещ a , b , цел n , вещь S)

```

рез S
нач вещь x, h, цел i
h:=(b-a)/h; s:=0
x:=a; i:=1
пока i<=n
нц
  S:=S+f(x);i:=i+1
  x:=x+h
кц
S:=S*h

```

кон

Для решения нашей задачи примем $h=1$, вместо a , b , $f(x)$ берем только целые части. Допускаем, что функция $f(x)$ возрастает на всем промежутке $[a, b]$. Если $f(x)$ убывает на $[a, b]$, используем алгоритм, суммирующий площади «правых» прямоугольников. Если же функция немонотонная, то промежуток $[a, b]$ разбиваем условиям монотонности. Площадь находим как сумму площадей образовавшихся фигур. Вычисленная площадь соответствует количеству клеток, целиком лежащих внутри криволинейной трапеции.

Вернемся к задаче 1. Задающая функция $f(x) = \text{sqr}(r^2 - x^2)$ возрастает на промежутке $[-r; 0]$. Вычисляем количество клеток, целиком лежащих внутри четверти окружности. Умножив результат на 4, получаем количество клеток, целиком лежащих внутри этой окружности. Исходные данные: $f=-r$, $b=0$, $h=1$. Ниже приведены программы, написанные на языках Qbasic и Turbo Pascal.

Программа 1.1	Программа 1.2
<pre> 'PROG 1 INPUT R A = -R: B = 0 H = 1: S = 0: X = A I = 1: N = (B - A) / H WHILE I <= N Y = (SQR(R ^ 2 - X ^ 2)) G = FIX(Y) </pre>	<pre> Program P1_2; Uses crt; Var r,a,b,h,s,i,x,k,g:integer; y,n:real; Begin Clrscr; {очистка экрана} read(r); a:=-r; b:=0; h:=1;s:=0;x:=a; </pre>

<pre> S = S + G I = I + 1 X = X + H WEND K = S * 4 PRINT K END </pre>	<pre> i:=1;n:=(b-a)/h; WHILE i <= n DO begin y:=(SQRT(r*r - x*x)); g:=TRUNC(y); s:=s+g; i:=i+1; x:=x+h; end; k:=s*4; writeln(k); readkey; End. </pre>
---	--

В табл. 1 приведены значения K для некоторых значений R.

Таблица 1

R	2	3	4	5	6	7	8	9	10	11	12	13
K	4	16	32	60	88	120	164	216	276	332	392	476

После упрощения программа выглядит следующим образом.
(В программах 2.1 и 2.2 реализован алгоритм суммы прямоугольников).

Программа 2.1	Программа 2.2
<pre> ' PROG 2 INPUT R FOR X = 1 TO R Z = R ^ 2 - X ^ 2 IF Z < 0 THEN 100 U = SQR(Z) Q = FIX(U) S = S + Q 100 NEXT X K = S * 4 PRINT K END </pre>	<pre> Program P2_2; Uses crt; label 100; Var x,r,q,s,k:integer; z,u:real; Begin Clrscr; {очистка экрана} Read(r); FOR x:= 1 TO r DO Begin z:=r*r-x*x; IF z < 0 THEN goto 100; u:=SQRT(z); q:=TRUNC(u); s:=s+q; 100:end; k:=s*4; </pre>

	writeln(k); readkey; End.
--	---------------------------------

Задача 2. Трехмерное пространство разбито на кубики с ребром длиной 1. Сколько кубиков помещается в сфере радиуса R, центр которой находится в вершине одного из кубиков?

В программах 3.1 и 3.2 приведена программа решения задачи 2. Здесь использовалась аналогия решения задачи 1, реализованная в программах 2.1 и 2.2.

Программа 3.1	Программа 3.2
<pre>'PROG 3 INPUT R FOR X = 1 TO R FOR Y = 1 TO R Z = R ^ 2 - X ^ 2 - Y ^ 2 IF Z < 0 THEN 100 U = SQR(Z) Q = FIX(U) V = V + Q 100 NEXT Y NEXT X K = V * 8 PRINT K END</pre>	<pre>Program P3_2; Uses crt; Label 100; Var r,x,y,q,v,k:integer; z,u:real; Begin Clrscr; {очистка экрана} read(r); FOR x:=1 TO r DO FOR y:=1 TO r DO begin z:=r*r-x*x-y*y; IF z < 0 THEN GOTO 100; u:=SQRT(z); q:=TRUNC(u); v:=v+q; 100: end; k:=v*8; writeln(k); readkey; end.</pre>

В табл. 2 приведены значения K для соответствующих значений R.

Таблица 2

R	2	3	4	5	6	7	8	9	10	11
K	8	56	136	304	624	1016	1568	2368	3280	4512

5.2. РЕШЕНИЕ ЗАДАЧ НА ТЕМУ « РАЗЛОЖЕНИЕ НАТУРАЛЬНОГО ЧИСЛА НА НАТУРАЛЬНЫЕ СЛАГАЕМЫЕ»

Обучение школьников решению задач обычно осуществляется при решении готовых задач. Между тем существенную роль играет их составление учащимися.

В методическом отношении развитие темы задачи ценно тем, что приучает учащихся к переконструированию задач, что является одним из основных приемов поиска решения.

В качестве исходной рассмотрим задачу 1. Программы решения приведены на языках Бейсик и Паскаль.

Задача 1. Разложение на слагаемые. Напечатать все представления натурального числа N суммой натуральных чисел. Перестановка слагаемых нового способа не дает. Подсчитать количество разложений.

Программа 1.1	Программа 1.2
<pre> 10 REM разложить на слагаемые 20 INPUT «N=»; N: DIM M(N) 30 M(1) = N: K = 1: I = 1: GOTO 80 40 T = M(K) - 1: S = T + I - K + 1 50 FOR I = K TO N 60 IF S > T THEN M(I) = T: S = S - T ELSE M(I) = S: GOTO 80 70 NEXT I 80 FOR K = 1 TO I: PRINT M(K); : NEXT 81 Z = Z + 1 90 PRINT 100 FOR K = I TO 1 STEP -1 110 IF M(K) > 1 GOTO 40 120 NEXT K 125 PRINT «КОЛ-ВО РАЗЛОЖЕ- НИЙ»; Z 130 END </pre>	<pre> Program P1_2; { Разложить на слагаемые } Uses crt; Label 4,8; var m : array [1..30] of Integer; k,i,t,s,z,n: Integer; begin write('введи n='); readln(n); m[1] := n; k := 1; i := 1; z := 0; goto 8; 4:t := m[k] - 1; s := t + i - k + 1; </pre>

	<pre> for i := k to n do if s > t then begin m[i] := t; s := s - t; end </pre>
--	---

	<pre> else begin m[i] := s; goto 8; end; 8:for k := 1 to i do write(m[k]); writeln; z := z + 1; for k := i downto 1 do if m[k] > 1 then goto 4; writeln('Кол-во разложе- ний',z); readkey; end. </pre>
--	---

Для n=7 получили следующее разложение:

- 7
- 6 1
- 5 2
- 5 1 1
- 4 3
- 4 2 1
- 4 1 1 1
- 3 3 1
- 3 2 2
- 3 2 1 1
- 3 1 1 1 1
- 2 2 2 1
- 2 2 1 1 1
- 2 1 1 1 1 1
- 1 1 1 1 1 1 1

Количество разложений равно 15. Ниже приведена таблица количеств разложений для некоторых натуральных чисел.

Таблица 2

N	1	2	3	4	5	6	7	8	9	10	11
Z	1	2	3	5	7	11	15	22	30	42	56
N	12	13	14	15	16	17	18	19	20	21	
Z	77	101	135	176	231	287	385	490	627	792	

Задача 2. Разложить натуральное число N на три натуральных слагаемые. Подсчитать количество разложений.

Программа 2.1	Программа 2.2
<pre> 10 REM РАЗЛОЖИТЬ НА ТРИ СЛАГАЕМЫЕ 20 INPUT «N=»; N: DIM M(N) 30 M(1) = N: K = 1: I = 1: GOTO 80 40 T = M(K) - 1: S = T + I - K + 1 50 FOR I = K TO N 60 IF S > T THEN M(I) = T: S = S - T ELSE M(I) = S: GOTO 80 70 NEXT I 80 IF I <> 3 GOTO 90 85 FOR K = 1 TO I: PRINT M(K); : NEXT 86 Z = Z + 1 90 PRINT 100 FOR K = I TO 1 STEP -1 110 IF M(K) > 1 GOTO 40 120 NEXT K 125 PRINT «КОЛ-ВО РАЗЛО- ЖЕНИЙ»; Z 130 END </pre>	<pre> Program P2_2; { Разложить на три слагаемые } Uses crt; Clrscr; {очистка экрана} label 4,8,9; var m : array [1..30] of Integer; k,i,t,s,z,n: Integer; begin write('введи n='); readln(n); m[1] := n; k := 1; i := 1; z := 0; goto 8; 4:t := m[k] - 1; s := t + i - k + 1; for i := k to n do if s > t then begin m[i] := t; s := s - t; end else begin m[i] := s; goto 8; end; 8:if i <> 3 then goto 9; for k := 1 to i do write(m[k]); writeln; z := z + 1; 9:for k := i downto 1 do if m[k] > 1 then goto 4; writeln('Кол-во разложений ',z); readkey; end. </pre>

Для N=7 получаем:

```

5 1 1
4 2 1
3 3 1
3 2 2

```

Количество равно 4.

Задача 3. Разложить натуральное число N на три различные натуральные слагаемые. Перестановка слагаемых нового способа не дает. Подсчитать количество разложений.

Программа 3.1	Программа 3.2
<pre> 10 REM РАЗЛОЖЕНИЕ НА ТРИ РАЗЛИЧНЫЕ СЛАГАЕМЫЕ 20 INPUT «N=»; N: DIM M(N) 30 M(1) = N: K = 1: I = 1: GOTO 80 40 T = M(K) - 1: S = T + I - K + 1 50 FOR I = K TO N 60 IF S > T THEN M(I) = T: S = S - T ELSE M(I) = S: GOTO 80 70 NEXT I 80 IF I <> 3 GOTO 90 81 IF M(1) = M(2) GOTO 90 82 IF M(1) = M(3) GOTO 90 83 IF M(2) = M(3) GOTO 90 85 FOR K = 1 TO I: PRINT M(K); 86 NEXT 87 Z = Z + 1 88 PRINT 90 FOR K = I TO 1 STEP -1 110 IF M(K) > 1 GOTO 40 120 NEXT K 125 PRINT «КОЛ-ВО РАЗЛОЖЕ- НИЙ»; Z 130 END </pre>	<pre> Program P3_2; { Разложение на три различные слагае- мые} Uses crt; label 4,8,9; var m : array [1..30] of Integer; k,i,t,s,z,n: Integer; begin Clrscr; {очистка экрана} write('ÿГ«ЁвГ n: n='); readln(n); m[1] := n; k := 1; i := 1; z := 0; goto 8; 4:t := m[k] - 1; s := t + i - k + 1; for i := k to n do if s > t then begin m[i] := t; s := s - t; end else begin m[i] := s; goto 8; end; 8:if i <> 3 then goto 9; if m[1] = m[2] then goto 9; if m[1] = m[3] then goto 9; if m[2] = m[3] then goto 9; for k := 1 to i do write(m[k]); writeln; z := z + 1; 9:for k := i downto 1 do if m[k] > 1 then goto 4; writeln('Кол-во разложений ',z); readkey; end. </pre>

Для N=7:

4 2 1

Количество разложений равно 1.

Задача 4. Разложить натуральное число N на не более V натуральных слагаемых. Подсчитать количество разложений. Перестановка слагаемых новое разложение не дает.

Программа 4.1	Программа 4.2
<pre> 10 REM РАЗЛОЖЕНИЕ НА НЕ БОЛЕЕ V СЛАГАЕМЫЕ 20 INPUT «N=»;N: DIM M(N) 25 INPUT V 30 M(1)=N:K=1:I=1:GOTO 80 40 T=M(K)-1:S=T+I-K+1 50 FOR I=K TO N 60 IF S>T THEN M(I)=T: S=S-T ELSE M(I)=S:GOTO 80 70 NEXT I 80 IF I>V GOTO 90 85 FOR K=1 TO I:PRINT M(K);:NEXT 86 Z=Z+1 90 PRINT 100 FOR K=I TO 1 STEP -1 110 IF M(K)>1 GOTO 40 120 NEXT K 125 PRINT «КОЛ-ВО РАЗЛО- ЖЕНИЙ»;Z 130 END </pre>	<pre> Program P4_2; {Разложение на не более v слагаемые} Uses crt; label 4,8,9; var m : array [1..30] of Integer; k,i,t,s,z,n,v: Integer; begin Clrscr; {очистка экрана} write('ÿГ¤ЁВГ n: n='); readln(n); write('ÿГ¤ЁВГ v: v='); readln(v); m[1] := n; k := 1; i := 1; z := 0; goto 8; 4:t := m[k] - 1; s := t + i - k + 1; for i := k to n do if s > t then begin m[i] := t; s := s - t; end else begin m[i] := s; goto 8; end; 8:if i > v then goto 9; for k := 1 to i do write(m[k]); writeln; z := z + 1; 9:for k := i downto 1 do if m[k] > 1 then goto 4; writeln('Кол-во разложений ',z); readkey; end. </pre>

Для $N=7$ и $V=3$ получаем

7

6 1
 5 2
 5 1 1
 4 3
 4 2 1
 3 3 1
 3 2 2

Количество разложений, удовлетворяющих данному условию, равно 8.

Задача 5. Разложить натуральное число N на натуральные слагаемые, которые не превосходят число M . Подсчитать количество разложений. Перестановка слагаемых нового разложения не дает.

Программа 5.1	Программа 5.2
<pre> 10 REM РАЗЛОЖИТЬ НА СЛАГАЕМЫЕ НЕ ПРЕВОСХО- ДЯЩИЕ M 20 INPUT «N=»;N:DIM M(N) 25 INPUT «M=»;M 30 M(1)=N: K=1: I=1:GOTO 80 40 T=M(K)-1:S=T+I-K+1 50 FOR I=K TO N 60 IF S>T THEN M(I)=T:S=S-T ELSE M(I)=S:GOTO 80 70 NEXT I 80 FOR K=1 TO I 81 IF M(K)>M THEN 90 82 PRINT M(K);: NEXT 83 Z=Z+1 90 PRINT 100 FOR K=I TO 1 STEP -1 110 IF M(K)>1 GOTO 40 120 NEXT K 125 PRINT «КОЛ-ВО РАЗ- ЛОЖЕНИЙ»; Z 130 END </pre>	<pre> Program P5_2; { Разложить на слагаемые не превосходящие C} Uses crt; label 4,8,9; var m: array [1..30] of Integer; k,i,t,s,z,n,c: Integer; begin Clrscr; {очистка экрана} write('Введи n: n='); readln(n); write('Введи c: c='); readln(c); m[1] := n; k := 1; i := 1; z := 0; goto 8; 4:t := m[k] - 1; s := t + i - k + 1; for i := k to n do if s > t then begin m[i] := t; s := s - t; end else begin m[i] := s; goto 8; end; 8:for k := 1 to i do begin if m[k] > c then goto 9; </pre>

	<pre> write(m[k]); end; writeln; z := z + 1; 9:for k := i downto 1 do if m[k] > 1 then goto 4; writeln('Кол-во разложений ',z); readkey; end. </pre>
--	---

Для $N=7$ и $M=3$ получаем:

```

3 3 1
3 2 2
3 2 1 1
3 1 1 1 1
2 2 2 1
2 2 1 1 1
2 1 1 1 1 1
1 1 1 1 1 1 1

```

Количество разложений 8. Аналогично можно решить следующую задачу.

Задача 6. Разложить натуральное число N на K слагаемые, не превосходящие натуральное число M . Подсчитать количество разложений.

5.3. РЕШЕНИЕ ЗАДАЧ НА ТЕМУ «ПРОСТЫЕ ЧИСЛА»

Компьютер вносит в учебный процесс принципиально новые познавательные средства, в частности, вычислительный эксперимент.

Статья И.С. Зельцера и Б.А. Кордемского «Занятные стайки простых чисел» (Математика в школе. 1988 г. № 6, с. 49-51) определила выбор темы и конкретные задачи вычислительного эксперимента на тему «Простые числа». Решение поставленных проблем рассмотрим на конкретных задачах.

Задача 1. Проверить, является ли данное число простым.

Программа 1 выводит утвердительный ответ в случае, если введенное число простое.

Программа 1.1	Программа 1.2
<pre> INPUT M IF M MOD 2=0 GOTO 70 FOR K=3 TO INT(SQR(M))+1 STEP 2 A=M MOD K IF A=0 GOTO 70 NEXT K PRINT « ЧИСЛО ПРОСТОЕ»; M 70 END </pre>	<pre> Program P1_2; Uses crt; label 70; var k,a,m:longint; begin write('m=');readln(m); if (m mod 2)=0 then goto 70; k:=3; repeat a:=m mod k; if a=0 then goto 70; k:=k+2; until k>round(sqrt(m))+1; writeln('число простое ',m); 70: readkey; end. </pre>

Задача 2. Вывести все простые числа в указанном интервале.

Ниже приводится решение задачи для интервала от 1 до 9999.
При желании интервал можно изменить.

Программа 2.1	Программа 2.2
<pre> FOR M=1 TO 9999 STEP 2 FOR K=3 TO INT(SQR(M))+1 STEP 2 A=M MOD K IF A=0 GOTO 70 NEXT K PRINT M; 70 NEXT M END </pre>	<pre> Program P2_2; Uses crt; var k,a,m,l:integer; label 70; begin m:=1; repeat k:=3; repeat a:=m mod k; if a=0 then goto 70; k:=k+2; until k>round(sqrt(m))+1; write(m,' '); 70: m:=m+2; until m>9999; readkey; end. </pre>

Задача 3. Вывести простые числа вида $3xxx3$, $7xxx7$, $9xxx9$.

Для чисел вида $3xxx3$ и $9xxx9$ простых чисел нет, т.к. числа этого вида делятся на 3.

Для вида $7xxx7$ простых чисел пять: 72227, 75557, 76667, 78887, 79997.

Программа 3.1	Программа 3.2
<pre> FOR L=0 TO 9 M=70007+1110*L FOR K=3 TO INT(SQR(M))+1 STEP 2 A=M MOD K IF A=0 GOTO 70 NEXT K PRINT M; 70 NEXT L END </pre>	<pre> Program P3_2; Uses crt; label 70; var k,a,m,l:longint; begin for l:=0 to 9 do begin m:=70007+1110*l; k:=3; repeat a:=m mod k; if a=0 then goto 70; k:=k+2; until k>round(sqrt(m))+1; write(m, ' '); 70:end; readkey; end. </pre>

Задача 4. Сколько полиндромических пятизначных простых чисел, т.е. вида $axuxa$?

Компьютер выдал 93 значения.

Программа 4.1	Программа 4.2
<pre> F=0: CLS FOR L=1 TO 9 STEP 2 FOR X=0 TO 9 FOR Y=0 TO 9 M=L*10001+X*1010+Y*100 FOR K=3 TO INT(SQR(M))+1 STEP 2 A=M MOD K IF A=0 GOTO 70 NEXT K PRINT M; :F=F+1 </pre>	<pre> Program P4_2; Uses crt; label 70; var f,l,x,y,k,a,m:longint; begin f:=0;l:=1; repeat for x:=0 to 9 do for y:=0 to 9 do begin m:=l*10001+x*1010+y*100; </pre>

<pre> 70 NEXT Y NEXT X NEXT L PRINT :PRINT «Всего»;F END </pre>	<pre> k:=3; repeat a:=m mod k; if a=0 then goto 70; k:=k+2; until k>round(sqrt(m))+1; write(m,' '); f:=f+1; 70:end; l:=l+2; until l>9; writeln ('vsego',f); readkey; end. </pre>
---	--

Задача 5. Сколько четырехзначных, простых чисел с тремя одинаковыми цифрами?

Программа 5.1	Программа 5.2
<pre> FOR I=1 TO 9 FOR J=1 TO 9 M=1110*I+J IF M MOD 2=0 GOTO 70 FOR K=3 TO INT(SQR(M))+1 STEP 2 A=M MOD K IF A=0 GOTO 70 NEXT K PRINT M; 70 NEXT J NEXT I END </pre>	<pre> Program P5_2; Uses crt; label 70; var i,j,m,k,a:longint; begin for i:=1 to 9 do for j:=1 to 9 do begin m:=1110*i+j; if m mod 2=0 then goto 70; k:=3; repeat a:=m mod k; if a=0 then goto 70; k:=k+2; until k>round(sqrt(m))+1; write(m,' '); 70:end; end; readkey; end. </pre>

Данная программа выводит четырехзначные, простые числа, у которых первые три цифры одинаковые. Таких чисел восемь: 1117, 2221, 3331, 4441, 4447, 5557, 6661, 8887.

Чтобы получить простые четырехзначные числа вида $xxux$ нужно в программе 4 изменить строку формирования числа M .

$$M=1101*I+J*10$$

Простых чисел вида $xxux$ – оказалось 12: 1151, 1171, 1181, 3313, 3323, 3343, 3373, 7717, 7727, 7757, 9929, 9949.

Для получения простых чисел вида $xuux$ меняем в программе $M=1011*I+J*100$.

Простых чисел вида $xuux$ 11: 1511, 1811, 3433, 3533, 3733, 3833, 7177, 7477, 7577, 7877, 9199.

Изменение $M=111*I+J*1000$ позволяет получить простые числа вида $uxxx$. Таких 12: 1777, 1999, 2111, 2333, 2777, 2999, 4111, 4999, 5333, 7333, 8111, 8999. Всего четырехзначных, простых чисел с тремя одинаковыми цифрами 43.

Задача 6. Сколько всего симметричных пар-перевертышей среди пятизначных, простых чисел, т.е. вида $abcde - edcba$.

Пятизначных, простых симметричных пар-перевертышей – 1499.

Программа 6.1	Программа 6.2
<pre> FOR M=10001 TO 99999 STEP 2 FOR K=3 TO INT(SQR(M))+1 STEP 2 A=M MOD K IF A=0 GOTO 210 NEXT K F=M\10000 E=(M-F*10000)\1000 V=(M-F*10000-E*1000)\100 W=(M-F*10000-E*1000- V*100)\10 R=M-F*10000-E*1000-V*100- W*10 L=R*10000+W*1000+V*100+E*10 +F IF L MOD 2=0 GOTO 210 FOR J=3 TO INT(SQR(L))+1 STEP 2 Y=L MOD J IF Y=0 GOTO 210 NEXT J </pre>	<pre> Program P6_2; Uses crt; Label 210; var m,k,a,o,j,y,f,e,v,l,r,w:longint; begin m:=10001; while m<=99999 do begin k:=3; repeat a:=m mod k; if a=0 then goto 210; k:=k+2; until k>round(sqrt(m))+1; f:=m div 10000; e:=(m-f*10000) div 1000; v:=(m-f*10000-e*1000) div 100; w:=(m-f*10000-e*1000-v*100) div 10; r:=m-f*10000-e*1000-v*100- w*10; </pre>

<pre>O=O+1 PRINT M;L, 210 NEXT M: PRINT O: END</pre>	<pre>l:=r*10000+w*1000+v*100+e*10 +f; if l mod 2=0 then goto 210; j:=3; repeat y:=l mod j; if y=0 then goto 210; j:=j+2; until j>round(sqrt(l))+1; o:=o+1; writeln(m, ' ',l); 210:m:=m+2; end; writeln('vsego',o); readkey; end.</pre>
--	---

Задача 7. Среди простых чисел есть «квартеты», состоящие из подряд идущих простых чисел, последние цифры которых – последовательность 1, 3, 7, 9. Например: 11, 13, 17, 19 или 101, 103, 107, 109. Сколько таких «квартетов» есть среди N-значных, простых чисел при N>3?

Среди четырехзначных, простых чисел таких «квартетов» 28.

Среди пятизначных, простых чисел таких «квартетов» 104.

Программа 7.1	Программа 7.2
<pre>L=0 FOR P=1001 TO 10000 STEP 10 M(1)=P: l=1: GOTO 10 1 IF M(1)MOD 10=1 THEN M(2)=M(1)+2: l=2:GOTO 10 GOTO 15 2 IF M(2)MOD 10=3 THEN M(3)=M(1)+6: l=3:GOTO 10 GOTO 15 3 IF M(3)MOD 10=7 THEN M(4)=M(1)+8: l=4:GOTO 10 GOTO 15 5 PRINT M(1);M(2);M(3);M(4):GOTO 15 10 FOR K=3 TO INT(SQR(M(l)))+1 STEP 2</pre>	<pre>Program P7_2; Uses crt; Label 1,2,3,5,10,15,20; var l,p,i,k,a,n:longint; m:array[1..4] of longint; begin l:=0;p:=9999; while p<100000 do begin m[1]:=p;i:=1; goto 10; 1: if m[1] mod 10=1 then be- gin m[2]:=m[1]+2;i:=2;goto 10;end; goto 15; 2: if m[2] mod 10=3 then be-</pre>

<pre> A=M(I)MOD(K) IF A=0 THEN GOTO 15 NEXT K L=L+1 IF L=4 GOTO 5 IF L=1 GOTO 1 IF L=2 GOTO 2 IF L=3 GOTO 3 15 L=0: M(2)=0: M(3)=0: M(4)=0 20 NEXT P END </pre>	<pre> gin m[3]:=m[1]+6;i:=3;goto 10;end; goto 15; 3: if m[3] mod 10=7 then be- gin m[4]:=m[1]+8;i:=4;goto 10;end; goto 15; 5: writeln(' ',m[1],' ',m[2],' ',m[3],' ',m[4]); n:=n+1;goto 15; 10: k:=3; repeat a:=m[i] mod k; if a=0 then goto 15; k:=k+2; until k>round(sqrt(m[i]))+1; l:=l+1; if l=4 then goto 5; if l=1 then goto 1; if l=2 then goto 2; if l=3 then goto 3; 15: l:=0; m[2]:=0; m[3]:=0; m[4]:=0; p:=p+2; end; writeln('n=',n); readkey; end. </pre>
---	--

Примечание: Для пятизначных, простых чисел в программе 7 меняем вторую строчку: FOR P=10001 TO 100000 STEP 10.

Задача 8. Стайка «жар-птиц» – простых чисел с «пером» 13 в «хвосте»: 13, 113, 313, 613, ... «Гнездо» их – формула $P=13+100 \cdot Tn$.

Конечно ли число «жар-птиц» в этом гнезде?

Вычисления показали, что таких чисел достаточно много в рассмотренных интервалах.

Программа 8.1	Программа 8.2
INPUT F	Program P8_2; Uses crt;

<pre> FOR G=1 TO F M=100*G+13 FOR K=3 TO INT(SQR(M))+1 STEP 2 A=M MOD K IF A=0 GOTO 65 NEXT K PRINT M; 65 NEXT END </pre>	<pre> label 65; var f,g,m,k,a:longint; begin write('vvedi f: f=');readln(f); for g:=0 to f do begin m:=100*g+13; k:=3; repeat a:=m mod k; if a=0 then goto 65; k:=k+2; until k>round(sqrt(m))+1; write(m, ' '); 65:end; readkey; end. </pre>
---	---

Задача 9. Число 31 – обращенное число 13. Другой «хвост» у стайки простых чисел: 31, 331, 3331, 33331. Их гнездо – формула: $P_k = (10^k - 7) / 3$ (аналог задачи 8).

Программа 9.1	Программа 9.2
<pre> INPUT F FOR G=1 TO F M=(10^G-7)/3 FOR K=3 TO INT(SQR(M))+1 STEP 2 A=M MOD K IF A=0 GOTO 65 NEXT K PRINT M; 65 NEXT G END </pre>	<pre> Program P9_2; Uses crt; label 65; var f,g,m,k,a:longint; begin write('vvedi f: f=');readln(f); for g:=1 to f do begin m:=round(((exp(g*ln(10))-7)/3); k:=3; repeat a:=m mod k; if a=0 then goto 65; k:=k+2; until k>round(sqrt(m))+1; write(m, ' '); 65:end; readkey; end. </pre>

Для $K=7$ получили значения: 31, 331, 3331, 33331, 333331, 3333331. Для $K=8$ и $K=9$ не получили значений. При $K=10$ компьютер указал на переполнение памяти.

5.4. РЕШЕНИЕ ЗАДАЧ НА НАТУРАЛЬНЫЕ ЧИСЛА

Задача 1. Определить натуральное число, введенное с клавиатуры, кратное пяти.

Решение. При решении используется следующее свойство: если в числе N последняя цифра 0 или 5, то число кратно 5. Для выделения младшей цифры используем функцию MOD.

Программа 1.1	Программа 1.2
<pre> 10 INPUT «ВВЕДИТЕ НАТУРАЛЬНОЕ ЧИСЛО»; N 20 M=N MOD 10 30 IF M=0 OR M=5 THEN PRINT «КРАТНО 5» 40 PRINT «НЕ КРАТНО 5» 50 END </pre>	<pre> Program P1_2; Uses crt; Var m,n:integer; begin writeln ('Введите натуральное число n'); readln (n); m:=n mod 10; if(m=0) or (m=5) then writeln('Кратно 5') else writeln('Не кратно 5'); readkey; end. </pre>

Задача 2. Выделить цифры натурального числа Z , введенного с клавиатуры.

Решение.

Программа 2.1	Программа 2.2
<pre> 10 INPUT «ВВЕДИТЕ НАТУ- РАЛЬНОЕ ЧИСЛО»; Z 20 N=Z 'РАБОЧАЯ КОПИЯ ЧИСЛА Z 30 C=N MOD 10 'ОТДЕЛЕНИЯ «МЛАДШЕЙ» ЦИФРЫ 40 PRINT C 'ВЫВОД «МЛАДШЕЙ» ЦИФРЫ </pre>	<pre> Program P2_2; Uses crt; label 30; var z,n,c:integer; begin writeln ('Введите натуральное число z'); readln (z); n:=z;{Рабочая копия числа z} </pre>

<pre> 50 N=N\10 60 IF N>0 THEN 30 70 END </pre>	<pre> 30: c:=n mod 10;{Отделения «младшей» цифры} write(c,' ');{Вывод «младшей» цифры} n:=n div 10; if n>0 then goto 30; readkey; end. </pre>
--	--

Задача 3. Для натурального числа N определить, кратно ли оно 3.

Решение. При решении используем следующее свойство: если сумма цифр числа кратно 3, то и число кратно 3.

Программа 3.1	Программа 3.2
<pre> 10 INPUT «ВВЕДИТЕ НАТУ- РАЛЬНОЕ ЧИСЛО»; Z 20 N=Z 25 SUM=0 30 C=N MOD 10 40 SUM=SUM+C 50 N=N\10 60 IF N>0 THEN 30 70 IF SUM/3=SUM\3 THEN PRINT КРАТНО 3 80 END </pre>	<pre> Program P3_2; Uses crt; Label 30; var z,n,sum,c:integer; begin writeln ('Введите натуральное число z'); readln (z); n:=z; sum:=0; 30: c:=n mod 10; sum:=sum+c; n:=n div 10; if n>0 then goto 30; if sum/3=sum div 3 then writeln('Кратно 3'); readkey; end. </pre>

Задача 4. Определить разрядность натурального числа N, заданного пользователем.

Решение 1.

Программа 4.1.1	Программа 4.1.2.
<pre> 10 INPUT «ВВЕДИ НАТУ- РАЛЬНОЕ ЧИСЛО N»;N 20 КОР=N' РАБОЧАЯ КО- ПИЯ ЧИСЛА N 30 RAZR=0 </pre>	<pre> Program P4_12; Uses crt; label 40; var n,kop, razr:integer; begin writeln ('Введите натуральное </pre>

<pre> 40 КОР=КОР\10'ОТБРАСЫВАНИ Е «МЛАДШЕЙ» ЦИФРЫ 50 RAZR=RAZR+1'КОЛ-ВО ЦИФР В ЧИСЛЕ N 60 IF КОР<>0 THEN 40 70 PRINT «В ЧИСЛЕ»; N;»- »;RAZR;»ЦИФР» 80 END </pre>	<pre> число n'); readln (n); kor:=n;{Рабочая копия числа n} razr:=0; 40: kor:=kor div 10;{Отбрасывание младшей цифры} razr:=razr+1;{Кол-во цифр в числе n} if kor>0 then goto 40; write('В числе ',n,' ', razr,'цифр'); readkey; end. </pre>
--	---

Решение 2.

Другое решение данной задачи заключается в использовании функции преобразования числового значение N в литерную величину. Далее вычисляется длина литерной величины.

Программа 4.2.1	Программа 4.2.2
<pre> 10 INPUT N 20 X\$=STR(N) 30 Y=LEN(X\$)-2 40 PRINT»В числе»; N; «-»; Y; «ЦИФР» 50 END </pre>	<pre> Program P4_22; Uses crt; Clrscr; {очистка экрана} var n,y:integer; x:string; begin readln(n); str(n,x); y:=length(x); writeln(' ',y); readkey; end. </pre>

Замечание: В строке 30 Y=LEN(X\$) вычитываем 2, т.к. STR\$(N) значение длины N увеличивается на 1, а значение длины числа больше показателя степени 10 на 1; Например: $1578=1.578 \cdot 10^3$.

Задача 5. Вывести на экран все трехзначные числа, сумма цифр которых равна N. N вводится с клавиатуры.

Решение 1. Организуем простой перебор трехзначных чисел по выделению цифр трехзначного числа и проверку заданного условия.

Программа 5.1.1	Программа 5.1.2
<pre> 5 INPUT N 10 FOR I=100 TO 999 20 A=I\100 30 B=(I-A*100)\10 40 C=I-A*100-B*10 50 IF A+B+C=N THEN PRINT I 60 NEXT I </pre>	<pre> Program P5_12; Uses crt; var n,i,a,b,c:integer; begin readln(n); for i:=100 to 999 do begin a:=i div 100; b:= (i-a*100) div 10; c:=i-a*100-b*10; if a+b+c=n then write(i,' '); end; end; readkey; end. </pre>

Решение 2.

Программа 5.2.1.	Программа 5.2.2
<pre> 10 ' ЗАДАННАЯ СУММА ЦИФР 20 INPUT «N=» ;N 30 FOR I=0 TO 9; FOR J=0 40 K=N-I-J 50 IF 1<=K AND K<=9 THEN PRINT I+10*J+100*R 60 NEXT J,I 70 END </pre>	<pre> Program P5_22; Uses crt; var n,i,j,k:integer; begin readln(n); for i:=0 to 9 do for j:=0 to 9 do begin k:=n-i-j; if (1<=k) and (k<=9) then write(i+j*10+k*100,' '); end; end; readkey; end. </pre>

Задача 6. Выделить делители натурального числа, введенного с клавиатуры.

Решение. Используем определение: всякое число b , на которое a делится без остатка, называется делителем числа a .

Программа 6.1	Программа 6.2
<pre> 10 INPUT N 20 C=N\2 30 FOR DEL=1 TO C </pre>	<pre> Program P6_2; Uses crt; var n,kol,c,del:integer; begin </pre>

<pre> 40 IF N/DEL=N\DEL THEN PRINT DEL 50 NEXT DEL 60 END </pre>	<pre> readln(n); c:=n div 2; for del:=1 to c do if n/del=n div del then writeln(del, ' '); readkey; end. </pre>
--	---

Задача 7. Найти НОД двух натуральных чисел m и n .

Решение. Алгоритм Евклида:

- 1) если числа равны, то взять любое из них в качестве ответа, в противном случае продолжить выполнение алгоритма;
- 2) определить большее число из двух чисел;
- 3) заменить большее число разностью большего и меньшего чисел;
- 4) начать алгоритм сначала.

Программа 7.1	Программа 7.2
<pre> 10 INPUT M,N 20 X=M: Y=N 30 PRINT 40 IF X=Y THEN 100 50 IF X>Y THEN 80 60 Y=Y-X 70 GOTO 40 80 X=X-Y 90 GOTO 40 100 PRINT»НОД=«;X 110 STOP </pre>	<pre> Program P7_2; Uses crt; label 40,80,100; var m,n,x,y:integer; begin read(m,n); x:=m;y:=n; 40: if x=y then goto 100; if x>y then goto 80; y:=y-x; goto 40; 80: x:=x-y; goto 40; 100: write('НОД=',x); readkey; end. </pre>

Задача 8. Номер трамвайного билета состоит из шести цифр, вывести на экран все счастливые номера трамвайных билетов (т.е. таких, у которых сумма трех первых цифр равна сумме трех последних).

Решение 1.

Программа 8.1.1	Программа 8.1.2
<pre> 10 CLS </pre>	<pre> Program P8_1_2; Uses crt; </pre>

<pre> 20 FOR CH=100000 TO 999999 30 NL=INT(CH/1000) 'ЛЕВАЯ ПОЛОВИНА НОМЕРА 40 NP=CH-1000*NL 'ПРАВАЯ ПОЛОВИНА НОМЕРА 50 S1=0 60 S2=0 70 FOR I=1 TO 3 80 S1=S1+NL MOD 10 90 NL=NL\10 100 S2=S2+NP MOD 10 110 NP=NP\10 120 NEXT I 130 IF s1=s2 THEN PRINT ch;»- номер счастливого трамвайного билетика.» 140 NEXT CH 150 END </pre>	<pre> var ch,nl,np,s1,s2,i:longint; begin for ch:=100000 to 999999 do begin nl:=ch div 1000;{левая по- ловина номера} np:=ch-1000*nl;{правая по- ловина номера} s1:=0;s2:=0; for i:=1 to 3 do begin s1:=s1 + nl mod 10; nl:=nl div 10; s2:=s2 + np mod 10; np:=np div 10; end; if s1=s2 then writeln(ch,' счастливым билет'); end; readkey; end. </pre>
--	---

Решение 2.

Программа 8.2.1	Программа 8.2.2
<pre> 10 P=0 20 FOR S=0 TO 13 30 Q=0 40 FOR K=0 TO 9 50 FOR L=0 TO 9 60 IF K+L>S THEN 90 70 IF K+L<S+10 THEN Q=Q+1 80 NEXT L 90 NEXT K 100 P=P+Q*Q 110 NEXT S 120 PRINT 2*P-1 130 END </pre>	<pre> Program P8_2_2; Uses crt; Label 90; var p,s,q,k,l:longint; begin p:=0; for s:=0 to 13 do begin q:=0; for k:=0 to 9 do for l:=0 to 9 do begin if k+l>s then goto 90; if k+l<s+10 then q:=q+1; 90: end; p:=p+q*q; end; write(2*p-1); readkey; end. </pre>

Задача 9. Задано натуральное число N. Найти и напечатать все его простые делители.

Решение. Используем следующее определение и свойство: число, имеющее какой-нибудь делитель, отличный от себя и единицы, называют составным числом. Всякое составное число представлено в виде произведения простых чисел.

Программа 9.1	Программа 9.2
<pre> 10 'ПРОСТЫЕ ДЕЛИТЕЛИ 20 INPUT «N=»; N 30 I=2: GOTO 70 40 IF N MOD I<>0 THEN I=I+1: GOTO 40 50 IF I>J THEN PRINT I : J=I 60 N=N\I 70 IF N>1 THEN 40 80 END </pre>	<pre> Program P9_2; Uses crt; var i,j,n:integer; begin readln(n); j:=0;i:=2; while n>1 do begin while (n mod i)<>0 do i:=i+1; if i<>j then begin writeln(i); j:=i; end; n:=n div i; end; readkey; end. </pre>

Задача 10. Вывести на экран все трехзначные числа, кратные 3.

Вспомним, что значит число кратно какому-либо числу. Число кратно натуральному числу n, если оно делится на n нацело. На доске записаны программы 10.1, 10.2, 10.3. Учитель предлагает прокомментировать их программы.

Программа 10.1.1	Программа 10.1.2
<pre> 10 FOR X=102 TO 999 STEP 3 20 PRINT X 30 NEXT X 40 END </pre>	<pre> Program P10_1_2 Uses crt; var x:word; begin for x:=34 to 333 do writeln(3*x); end. </pre>

Программа 10.2.1	Программа 10.2.2
<pre> 10 FOR X=100 TO 999 20 IF X/3=INT(X/3) THEN PRINT X 30 NEXT X 40 END </pre>	<pre> Program P10_2_2 Uses crt; var x:word; begin for x:=100 to 999 do if x/3=trunc(x/3) then writeln(x); end. </pre>
Программа 10.3.1	Программа 10.3.2
<pre> 10 FOR X=100 TO 999 20 IF X MOD 3=0 THEN PRINT X 30 NEXT X 40 END </pre>	<pre> Program P10_3_2; uses crt; var x:word; begin for x:=100 to 999 do if x mod 3=0 then writeln(x); end. </pre>

Примерные варианты комментариев.

К программе 10.1. Число 102 кратно 3. Следующее число $102+3=105$ также кратно 3 и т.д. Организуем цикл и будем выводить на экран все трехзначные числа, начиная со 102 с шагом 3.

К программе 10.2. Организуем перебор всех трехзначных чисел с проверкой условия, если частное при делении числа на 3 равно своей целой части, то такие числа выводятся на экран.

К программе 10.3. Если число кратно 3, то его остаток при делении на 3 даст 0. Нам остается организовать перебор всех трехзначных чисел и проверить условие, если остаток при делении на 3 даст 0, вывести это число на экран.

Учитель. Вспомним еще одно свойство кратности числа на 3. Если сумма цифр числа кратно 3, то и само число кратно 3. Если при решении задачи 1 основная идея заключалась в том, чтобы составить число, то теперь мы используем идею получения цифр числа. Например, пусть дано число 326. Чтобы получить цифру сотен, т.е. 3, нужно $326/100=3.26$ и выделить его целую часть $INT(3.26)=3$. Для получения цифры десятков нужно от исходного числа отнять цифру сотен, умноженную на 100, разделить полученное число на 10 и от результата выделить целую часть. $INT((326-3*100)/10)=2$. Для получения цифры единиц нужно от

исходного числа отнять произведение количества сотен на 100 и произведение количества десятков на 10. $326-3*100-2*10=6$. Обратимся к программе 10.4. В строчках 20, 30, 40 получаем цифры числа. В строчке 50 – проверка кратности суммы цифр числа на 3 (аналогично программе 4), в случае выполнения условия – вывод числа на экран. В строчках 10 и 60 организуем перебор всех трехзначных чисел.

Программа 10.4.1	Программа 10.4.2
<pre> 10 FOR X=100 TO 999 20 A=INT(X/100) 30 B=INT((X-A*100)/10) 40 C=X-A*100-B*10 50 IF (A+B+C) MOD 3 = 0 THEN PRINT X 60 NEXT X 70 END </pre>	<pre> Program P10_4_2 uses crt; var x,a,b,c:word; begin for x:=100 to 999 do begin a:=trunc(x/100); b:=trunc((x-a*100)/10); c:=x-a*100-b*10; if (a+b+c) mod 3=0 then writeln(x); end; readkey; end. </pre>

Вывести на экран все четырехзначные числа, кратные 6 (различными способами).

Задача 11. Вывести на печать все совершенные числа в заданном диапазоне от L до R.

Решение. Совершенным называется число, равное сумме своих делителей.

Программа 11.1	Программа 11.2
<pre> 10 INPUT «ВВЕДИТЕ ИНТЕР- ВАЛ ПОИСКА СОВЕРШЕННЫХ ЧИСЕЛ»;L,R 20 K=0 30 FOR CH=L TO R 40 G=INT(SQR(CH)) 50 FOR D=2 TO G 60 IF CH/D=CH\D THEN SUM=SUM+D+CH/D </pre>	<pre> Program P11_2; Uses crt; var m,i,j,k,s,l: integer; begin readln(l,m); for i:=l to m do begin s:=1; j:=1; repeat j:=j+1; </pre>

<pre> 70 NEXT D 80 IF SUM=CH THEN PRINT CH; «СОВЕРШЕННОЕ ЧИСЛО»: K=K+1 90 NEXT CH 100 IF K=0 THEN PRINT «НА УКАЗАННОМ ИНТЕРВАЛЕ СОВЕР. ЧИСЕЛ НЕТ» 110 END </pre>	<pre> k:=i div j; if(i=k*j) AND (j<=k) then begin s:=s+j; if j<k then s:=s+k end; until j>=k; if s=i then writeln(i) end; readkey; end. </pre>
--	---

Задача 12. Найти наибольшее значение суммы M^2+N^2 , когда M и N пробегают все значения от 1 до 100, удовлетворяющих условию $ABS(N^2-M*N-M^2)=1$.

Решение.

Программа 12.1	Программа 12.2
<pre> 10 P=0 20 FOR M=1 TO 100 30 FOR N=1 TO 100 40 IF ABS(N^2-M*N-M^2)<>1 THEN 90 50 S=M^2+N^2 60 IF S>P THEN P=S 90 NEXT N 100 NEXT M 110 PRINT P 120 END </pre>	<pre> Program P12_2; Uses crt; Label 90; var p, m, n,s:integer; begin p:=0; for m:=1 to 100 do for n:=1 to 100 do begin if abs(n*n-m*n-m*m)<>1 then goto 90; s:=m*m+n*n; if s>p then p:=s; 90: end; writeln(p); readkey; end. </pre>

Задача 13. Дано натуральное число CH. Заменить в нем все «единицы» на «пятерки».

Решение 1.

Программа 13.1.1	Программа 13.1.2.
<pre> 10 CLS 20 INPUT «ЗАДАЙТЕ ЧИС- ЛО»;CH 30 N=CH </pre>	<pre> Program P13_1_2; Uses crt; label 60, 120; var ch, n, k, s, a, i, b:longint; </pre>

<pre> 40 K=0 50 S=0 60 IF N=0 THEN 120 70 A=N MOD 10 80 IF A<>1 THEN S=S+A*10^K ELSE S=S+5*10^K 90 N=N\10 100 K=K+1 110 GOTO 60 120 PRINT «ВОТ ВАМ РЕ- ЗУЛЬТАТ-»; S 130 END </pre>	<pre> begin read (ch); n:=ch;k:=0;s:=0; 60: if n=0 then goto 120; a:=n mod 10;b:=1; if a<>1 then begin for i:=1 to k do a:=a*10; s:=s+a; end else begin for i:=1 to k do b:=b*10; s:=s+5*b; end; n:=n div 10; k:=k+1; goto 60; 120:Writeln(s) ; readkey; end. </pre>
--	--

Решение 2.

Программа 13.2.1	Программа 13.2.2
<pre> 10 INPUT «КАКОЕ ЧИСЛО БУ- ДЕМ ПРЕОБРАЗОВЫВАТЬ»; CH 20 K=CH 30 J=1 40 F=K MOD 10 80 IF F=1 THEN F=5 90 S=J*F 100 M=M+S 110 J=J*10 120 K=INT(K/10) 130 IF K<>0 THEN 40 140 PRINT «ПОЛУЧИЛОСЬ M=»; M 150 END </pre>	<pre> Program P13_2_2; Uses crt; label 40; var ch,k,j,f,m,s:longint; begin writeln('какое число будем преобразовывать'); read(ch); k:=ch;m:=0; j:=1; </pre>
	<pre> 40:f:=k mod 10; if f=1 then f:=5; s:=j*f; m:=m+s; </pre>

	<pre> j:=j*10; k:=k div 10; if k <> 0 then goto 40; writeln('poluchilos m=',m); readkey; end.</pre>
--	---

Задача 14. Вычислить факториал $n!$. $n!=1*2*3*...*n$.

Решение. Рекурсивное описание факториала

1. $1!=1$
2. $n!=n(n-1)!$

Программа 14.1	Программа 14.2
<pre> 10 INPUT N□20 F = 1□30 FOR I = 1 TO N□40 F = F * N□50 NEXT I□60 PRINT F□70 END□</pre>	<pre> Program P14_2; Uses crt; var n,i:integer; f:longint; begin readln(n); f:=1; for i:=1 to n do f:=f*i; writeln(f); readkey; end.</pre>

Задача 15. Вычислить числа Фибоначчи.

Решение. Функция Фибоначчи. Ее рекурсивное описание.

1. $fib(1)=fib(2)=1$
2. $fib(n)=fib(n-1)+fib(n-2)$.

Программа 15.1	Программа 15.2
<pre> 10 INPUT «ВВЕДИТЕ НОМЕР ЧЛЕ- НА », N 20 A1 = 0 30 A2 = 1 40 FOR I = 1 TO N 50 R = A1 + A2 60 A1 = A2 70 A2 = R</pre>	<pre> Program P15_2; Uses crt; var a1,a2,n,r,i:integer; begin readln(n); a1:=0; a2:=1; for i:=1 to n do begin</pre>

<pre> 80 NEXT I 90 PRINT N, «- Й ЧЛЕН РАВЕН », A2 </pre>	<pre> r:=a1+a2; a1:=a2; a2:=r; end; write (n,'- й член равен- ',a2) readkey; end. </pre>
--	--

Задача 16. Вычисление факториала, результат которого представлен многоразрядным числом.

Решение.

Программа 16.1	Программа 16.2
<pre> 1 REM ВЫЧИСЛЕНИЕ ФАКТО- РИАЛА ЧИСЛА N 2 DEFLNG A-Z 5 DIM A(200) 10 INPUT N 15 FOR I=2 TO 200 20 A(I)=0:NEXT I 25 A(1)=1:L=1 30 FOR K=1 TO N 35 R2=0:R1=0:I=1 40 IF R2=0 THEN IF I>L THEN 80 45 R=A(I)*K+R2 55 R2=INT (R/1000000) 60 R1=R-R2*1000000 65 A(I)=R1 70 I=I+1 75 GOTO 40 80 L=I-1 85 NEXT K 86 FOR J=1 TO I-1 87 PRINT A(I-J): 88 NEXT J 93 END </pre>	<pre> Program P16_2; Uses crt; label 40,80; var a:array[1..200] of longint; l,r2,r1,r,n,i,j,k,d:longint; begin readln(n); for i:=2 to 200 do a[i]:=0; a[1]:=1;l:=1; for k:=1 to n do begin r2:=0;r1:=0;i:=1; 40: if (r2=0) then if (i>l) then goto 80; r:=a[i]*k+r2; r2:=r div 1000000; r1:=r-r2*1000000; a[i]:=r1; i:=i+1; goto 40; 80: l:=i-1; end; for j:=1 to i-1 do begin write(a[i-j]);write(' ');end; readkey; end. </pre>

Примечание. Каждый 0 в конце вычисленного результата нужно воспринимать как группу 000000.

Задача 17. Ввести вещественное число A и натуральное k . Вычислить и вывести на печать A^k с выполнением следующих условий: операцией возведения в степень пользоваться нельзя; k может оказаться настолько большим, что недопустимо выполнять k умножений.

Решение.

Программа 17.1	Программа 17.2
<pre> 10 'БЫСТРАЯ СТЕПЕНЬ 20 INPUT «A,K=»;A,K 30 PRINT A;»^»;K; 40 B=1 50 IF K MOD 2=0 GOTO 70 60 B=B*A 70 K=K\2:A=A*A 80 IF K>0 GOTO 50 90 PRINT «=»;B 100 END </pre>	<pre> Program P17_2; Uses crt; var a , b:real; k,n:integer; BEGIN writeln('A,K-'); readln(a,k);write(a,'^',k,'='); b:=1; while k>0 do begin n:=k div 2; if(n+n<k) then b:=b*a; k:=n; a:=a*a; end; writeln(b) readkey; END. </pre>

Задача 18. Представить произведение двух натуральных чисел многозначным числом.

Решение.

Программа 18.1	Программа 18.2
<pre> 10 'ПРОГРАММА УМНОЖЕНИЯ ДВУХ НАТУРАЛЬНЫХ ЧИСЕЛ 20 DEFINITA-Z:DIMA (100), B(100), C(200) 30 'ВВОД ЧИСЛА A 40 INPUT «ВВЕДИТЕ ЧИСЛО A»; A\$: KA=LEN(A\$) 50 FOR I=LEN(A\$) TO 1 STEP-1: C\$=MID\$(A\$,I,1) 60 A(KA+1-I)=VAL(C\$):NEXT I 70 'ВВОД ЧИСЛА B 80 INPUT «ВВЕДИТЕ ЧИСЛО B»;B\$: KB=LEN(B\$) </pre>	<pre> Program P18_2; {Програм- ма умножения двух нату- ральных чисел} Uses crt; var aa,bb,cc,ss:string; ka,kb,kc,i,p,q,j,z,k,d:integer; a,b:array[1..100] of byte; c:array[1..200] of byte; begin {Ввод числа A} writeln('Введите число A'); readln(aa);ka:=length(aa); for i:=length(aa) downto 1 </pre>

<pre> 90 FOR I=LEN(B\$) TO 1 STEP -1: C\$=MID\$(B\$,I,1) 100 B(KB+1-I)=VAL(C\$):NEXT I 110 'УМНОЖЕНИЕ МНОГО- ЗНАЧНОГО ЧИСЛА 120 FOR I=1 TO KA:C(I)=0:NEXT I 130 K=0:FOR I=1 TO KB:D=B(I):P=0 140 FOR J=1 TO KA:Q=C(J+K)+D*A(J)+P 150 C(J+K)=QMOD 10:P=Q\10:NEXT J 160 K=K+1:C(K+KA)=P:NEXT I 170 KC=KA+KB-1:IF P>0 THEN KC=KC+1 180 'ПЕЧАТЬ РЕЗУЛЬТАТА 190 FOR I=KC TO 1 STEP- 1:PRINT RIGHT\$(STR\$(C(I)),1); 200 NEXT I </pre>	<pre> do begin cc:=copy(aa,i,1); val(cc,a[ka+1-i],z); end; {Ввод числа B} writeln('Введите число B'); readln(bb);kb:=length(bb); for i:=length(bb) downto 1 do begin ss:=copy(bb,i,1); val(ss,b[kb+1-i],z); end; {Умножение многознач- ного числа} for i:=1 to ka do c[i]:=0; k:=0; for i:=1 to kb do begin d:=b[i];p:=0; for j:=1 to ka do begin q:=c[j+k]+d*a[j]+p; c[j+k]:=q mod 10;p:=q div 10; end; k:=k+1;c[k+ka]:=p; end; kc:=ka+kb-1;if p>0 then kc:=kc+1; {Печать результата} for i:=kc downto 1 do write(c[i]); readkey; end. </pre>
---	---

Задача 19. Написать программу нахождения наименьшего общего кратного (НОК) двух натуральных чисел, используя алгоритм нахождения наибольшего общего делителя (НОД) как вспомогательный.

Решение. Для любых натуральных a и b справедливо $a*b=НОД(a,b)*НОК(a,b)$.

Программа 19.1	Программа 19.2
<pre> 10 INPUT A,B 15 P=A*B 20 X=A:Y=B 40 IF X=Y THEN 100 50 IF X>Y THEN 80 60 Y=Y-X 70 GOTO 40 80 X=X-Y 90 GOTO 40 100 PRINT «НОК=»;P/X 110 END </pre>	<pre> Program P19_2; Uses crt; label 40,80,100; var m,n,p,x,y:integer; begin read(m,n); p:=m*n; x:=m;y:=n; 40: if x=y then goto 100; if x>y then goto 80; y:=y-x; goto 40; 80: x:=x-y; goto 40; 100: write('НОК=',p div x); readkey; end. </pre>

Задача 20. Перевод римских чисел в арабские.

Решение.

Программа 20.1	Программа 20.2
<pre> 20 DATA 1000,M,900,CM,500,D, 400,CD 30 DATA 100,C,90,XC,50,L,40,XL 40 DATA 10,X,9,IX,5,V,4,IV 50 DATA 1,I 60 INPUT «Римское чис- ло»;N\$:N=0 80 READ A,A\$ 90 IF A\$<>LEFT\$(N\$,LEN(A\$)) THEN 130 100 N=N+A 110 N\$=RIGHT\$(N\$,LEN(N\$)- LEN(A\$)) 120 GOTO 90 130 IF N\$<>« THEN 70 140 PRINT «Арабское число»; </pre>	<pre> Program P20_2; Uses crt; label 130; const no=12; ar:array[0..no] of word=(1,4,5,9,10,40,50,90,100,400,500,900,1000); rim:array[0..no] of string[2]=('I','IV','V','IX','X','XL','L','L C','C', 'CD','D','CM','M'); var n:string; k,j,i:integer; begin writeln('Римское число'); readln(n); k:=0; for j:=0 to no do </pre>

<pre>N 150 END</pre>	<pre>begin if rim[no- j]<>copy(n,1,length(rim[no-j])) then goto 130; while rim[no- j]=copy(n,1,length(rim[no-j])) do begin k:=k+ar[no-j]; delete(n,1,length(rim[no-j])); end; 130: end; writeln('Арабское число - ',k); readkey; end.</pre>
----------------------	---

Задача 21. Перевод арабских чисел в римские.

Решение.

Программа 21.1	Программа 21.2
<pre>20 DATA 1000,M,900,CM,500,D,400,CD 30 DATA 100,C,90,XC,50,L,40,XL 40 DATA 10,X,9,IX,5,V,4,IV 50 DATA 1,I 60 INPUT «АРАБСКОЕ ЧИСЛО - «; N 70 REM НАЧАЛО ЦИКЛА «ДО» 80 READ A, A\$ 90 IF N < A THEN 130 100 N\$ = N\$ + A\$ 110 N = N - A 120 GOTO 90 130 IF N <> 0 THEN 70 140 PRINT «РИМСКОЕ ЧИСЛО - «; N\$ 150 END</pre>	<pre>Program P21_2; Uses crt; label 130; const no=13; ar:array[1..no] of word=(1,4,5,9,10,40,50,90,100,40 0,500,900,1000); rim:array[1..no] of string[2]=('I','IV','V','IX','X','XL','L','L C','C','CD','D','CM','M'); var k:string; n,j:integer; begin writeln('Арабское число'); readln(n); k:=""; for j:=no downto 1 do begin if n<ar[j] then goto 130; while n>=ar[j] do begin k:=k+rim[j]; n:=n-ar[j]; end; 130: end; writeln('Римское число - ',k);</pre>

	<pre> readkey; end.</pre>
--	---------------------------

Задача 22. В массиве M(1,9) записаны разряды (цифры) некоторого натурального числа в I-ричной системе счисления (M(1) разряд единиц и т.д.). Отпечатать разряды этого числа в J-ричной системе счисления, начиная с разряда единиц. Числа I, J не превосходят 10.

Решение.

Программа 22.1	Программа 22.2
<pre> 10 'СИСТЕМЫ СЧИСЛЕНИЯ 20 DIM M(9) 30 INPUT I, J 40 FOR K = 1 TO 9: INPUT M(K): NEXT 50 FOR K = 1 TO 9: PRINT M(K); : NEXT 60 N = 0 70 FOR K = 9 TO 1 STEP -1 80 N = N * I + M(K) 90 NEXT K: PRINT N 100 PRINT (N MOD J) 110 N = N \ J 120 IF N > 0 GOTO 100 130 END</pre>	<pre> Program P22_2; Uses crt; uses crt; var i,j,k:integer; n :real; m :array[1..9] of integer; begin writeln('i,j:='); readln(i,j); for k:=9 downto 1 do begin write('m[',k,']:='); readln(m[k]); end; writeln; n:=0; for k:=9 downto 1 do n:=n*i+m[k]; writeln(trunc(n)); repeat write(trunc(n) mod j,' '); n:=trunc(n) div j; until n=0; writeln; readkey; end.</pre>

Задача 23. Деление двух натуральных чисел с любой точностью.

Решение.

Программа 23.1	Программа 23.2
<pre> 20 DIM D(100) 30 INPUT « ВВЕДИТЕ НАТУ- РАЛЬНЫЕ ДЕЛИМОЕ И ДЕЛИ-</pre>	<pre> Program P23_2; Uses crt; var a,b,c,e,n,i:integer; d:array[1..100] of byte;</pre>

<pre> ТЕЛЬ»; A,B 70 INPUT «ЧИСЛО ЗНАКОВ ПОСЛЕ ЗАПЯТОЙ»; N 80 C=INT(A/B): E=C 90 FOR I=1 TO N 100 D(I)=INT(A-B*E)*10/B) 120 A=(A-B*E)*10: E=INT(A/B) 130 NEXT I 140 PRINT «РЕЗУЛЬТАТ» 160 PRINT C».»; 170 FOR I=1 TO N 180 PRINT D(I); 190 NEXT I 200 END </pre>	<pre> begin writeln('Введите натуральные делимое и делитель'); readln(a,b); writeln('Число знаков после запятой');readln(n); c:=a div b;e:=c; for i:=1 to n do begin d[i]:=(a-b*e)*10 div b; a:=(a-b*e)*10;e:=a div b; end; writeln('Результат'); write(c, '.'); for i:=1 to n do write(d[i]); readkey; end. </pre>
---	---

5.5. ЗАДАЧИ ДЛЯ САМОСТОЯТЕЛЬНОГО РЕШЕНИЯ

Задача 1. Найти двузначное число, равное сумме цифры его десятков и квадрата цифры единиц.

Задача 2. Если к сумме цифр двузначного числа прибавить квадрат этой суммы, то снова получится это двузначное число. Найти все такие числа.

Задача 3. Квадрат трехзначного числа оканчивается тремя цифрами, которые как раз составляют взятое число. Найти все такие числа.

Задача 4. Найти четырехзначное число, которое при делении на 133 дает в остатке 125, а при делении на 134 дает в остатке 111.

Задача 5. Найдите трехзначное число, квадрат которого оканчивается тремя одинаковыми цифрами, отличными от нуля.

Задача 6. Найдите четырехзначное число, являющееся точным квадратом, у которого две первые цифры одинаковы и две последние то же одинаковы.

Задача 7. В магазине имеется мастика в ящиках по 16 кг, 17 кг, 21кг. Как организации получить 185 кг мастики, не вскрывая ящики?

Задача 8. Найти все трехзначные числа, равные сумме кубов своих цифр.

Задача 9. В трехзначном числе зачеркнули первую цифру слева; когда полученное число умножили на 7, получилось исходное трехзначное число. Найдите его.

Задача 10. В трехзначном числе, все цифры которого нечетны, зачеркнули среднюю цифру. Оказалось, что полученное двузначное число является делителем исходного числа. Найдите все такие трехзначные числа.

Задача 11. Сумма цифр трехзначного числа кратна 7. Найдите все такие числа.

Задача 12. Четырехзначное число, а также число, записанное теми же цифрами в обратном порядке, оба являются точными квадратами. Найдите эти числа.

Задача 12. Найдите все двузначные числа, сумма цифр которых не меняется при умножении числа на 2, 3, 4, 5, 6, 7, 8, 9.

Задача 14. Сколько слагаемых суммы $1+2+3+\dots$ надо взять, чтобы получилось трехзначное число, состоящее из одинаковых цифр?

Задача 15. Сколько есть целых чисел от 1 до 1998, которые не делятся ни на одно из чисел 6, 10, 15 ?

Задача 16. Найдите наименьшее натуральное число M , обладающее следующим свойством: сумма квадратов одиннадцати последовательных натуральных чисел, начиная с M , является точным квадратом.

Задача 17. Найдите четырехзначное число, в четыре раза меньше числа, записанного теми же цифрами, но в обратном порядке.

Задача 18. Определить: а) наибольшее, б) наименьшее значение отношения трехзначного числа к сумме его цифр. Для каких чисел это наибольшее (наименьшее) значение достигается ?

Задача 19. Пусть X – сумма кубов десяти последовательных натуральных чисел, а Y – сумма этих же натуральных чисел. Докажите, что $X^2 - Y^2$ делится на 300 без остатка.

Задача 20. Сколькими способами можно разменять 1 рубль монетами, достоинством 1, 2, 3 и 5 копеек ?

Задача 21. Составить программу, которая проверяла бы, можно ли заданное число представить в виде: а) произведения двух простых чисел; б) произведения трех простых чисел; в) квадрата какого-либо простого числа; г) куба какого-либо простого числа.

Задача 22. Утверждается, что функция $f=n^2+n+41$ является генератором простых чисел при целых n , кратных 41. Требуется доказать или опровергнуть это утверждение.

Задача 23. Дано простое число. Найти и вывести на печать ближайшее к нему простое число.

Задача 24. Разработать программу поиска трех простых чисел, предшествующих числу M , заданному пользователем, причем $M \geq 10$.

Задача 25. Разработать программу поиска и печати всех чисел-близнецов, принадлежащих заданному диапазону $[N, M]$.

Примечание. Числа-близнецы – два нечетных простых числа, разнящихся на 2. Например: 11 и 13; 17 и 19.

Задача 26. Найти и вывести на экран 3 совершенных числа, большего некоторого M .

Задача 27. Дано натуральное число N . Найти и вывести на экран ближайшее к нему совершенное число.

Задача 28. Разработать программу поиска и печати всех пар дружественных чисел, меньших 10000.

Примечание. Дружественными называются два натуральных числа, каждое из которых равно сумме делителей другого. Можно дать другое определение дружественных чисел: сумма всех делителей одного и другого такого числа равна сумме обоих чисел. Дружественными числами являются, например, 220 и 284.

Задача 29. Найти и вывести на печать все четырехзначные симметричные числа.

Примечание: Симметричные числа, состоящие из двух одинаковых частей (например 245245, 7676).

Задача 30. Вывести четырехзначное число. Заменить в нем первую цифру на: 1) разность первой и последней цифр – если разность положительна, 2) разность последней и первой цифр – в противном случае.

Задача 31. Генератор случайных чисел выдает N четырехзначных чисел. Вывести на печать лишь те из них, в которых совпадают первая и последняя цифры.

Задача 32. Переставить в нем цифры так, чтобы получилось наименьшее (наибольшее) число, записанное теми же цифрами.

Задача 33. Составить программу, выводящую на печать все n -значные числа Армстронга (причем $n < 10$).

Примечание: Числом Армстронга называется число, состоящее из n цифр ($n > 1$) и такое, что сумма его цифр, возведенных в n -ю степень, равна самому числу. Например, $153 = 1^3 + 5^3 + 3^3$.

Задача 34. Вывести на печать все натуральные числа, меньшие заданного, сумма квадратов цифр которых кратна 7.

Задача 35. Составить программу, выводящую все натуральные числа в промежутке $[1, 2000]$, равные сумме кубов своих цифр.

Задача 36. Составить программу, выводящую все натуральные числа в диапазоне $[1, 5000]$, равные кубу суммы своих цифр.

Задача 37. Выдать на экран все четырехзначные числа, у которых сумма первых двух цифр равна сумме двух последних.

Задача 38. Определить номерной знак автомашины, нарушившей правила движения, если по показаниям свидетелей номер записывается тремя цифрами, кратен 2, 5 и 7, а сумма его цифр равна 12.

Задача 39. Номер трамвайного билета состоит из шести цифр. Если сумма цифр номера трамвайного билета делится на 7, то билет можете считать «удачным». Определите, могут ли два билета подряд быть удачными.

Задача 40. Составить программу, печатающие такие номера «счастливых» трамвайных билетов, которые равны квадрату какого-либо натурального числа.

Задача 41. Генератор случайных чисел выдает N натуральных чисел в диапазоне от 1 до 1000. Вывести на экран лишь те из них, в записи которых встречается цифра 6.

Задача 42. В записи натурального числа, введенного с клавиатуры, заменить все «двойки» на «тройки». Вывести на печать общие делители исходного и полученного чисел.

Задача 43. Проанализировав все числа в диапазоне $[10, 100]$, создать и вывести на печать одномерный массив из тех чисел, в записи которых нет цифр 3 и 7.

Задача 44. Подсчитать количество шестизначных чисел, у которых все цифры разные. Вывести эти числа на печать.

Задача 45. Найти и вывести на экран все двузначные числа, сумма цифр которых не меняется при умножении числа на 2, 3, 4, 5, 6, 7, 8, 9.

Задача 46. Составить программу вычисления цифр корня натурального числа CH , введенного с клавиатуры.

Задача 47. Вычислить и вывести на печать цифровые корни простых чисел, принадлежащих диапазону [100, 10000].

Примечание: Если сложить все цифры какого-либо числа, затем все цифры найденной суммы и повторять аналогичные вычисления до получения однозначного числа (цифры), то получим значение цифрового корня данного числа.

Задача 48. Вывести на печать все натуральные числа из диапазона [100,99999], цифровой корень которых:

а) кратен 3 или 5; б) является простым числом.

Задача 49. Написать программу сложения двух простых дробей (числители и знаменатели вводятся в переменные M, N, P, и Q с клавиатуры).

Результат сложения вывести на экран также в виде простой дроби.

Задача 50. Два числа A и B выдаются генератором случайных чисел [100, 300]. Число C равно наибольшему общему делителю A и B. Число D – наименьшее общее кратное A и C. Вычислить корень квадратный из D.

Задача 51. Написать программу сокращения простой дроби (числитель CH и знаменатель ZN задаются с клавиатуры).

Вывести на экран заданную и сокращенную дроби.

Задача 52. Написать программу сокращения неправильной дроби с выделением целой части (числитель и знаменатель задаются в двух переменных CH и ZN). Вывести на печать заданную и сокращенную дроби.

Задача 53. Написать программу, которая проверяла бы, являются ли три числа взаимно простыми.

Задача 54. Вывести на печать делители числа M, взаимно простые с числом N.

Задача 55. Три числа считываются из блока данных. Определить, являются или нет НОД первых двух чисел и третье число взаимно простыми.

Задача 56. С клавиатуры вводится некоторое положительное целое число A. Число B есть целая часть выражения $SQR(A^2+15)$. Число C есть наименьшее общее кратное A и B. Определить, являются ли числа A и C взаимно простыми.

Задача 57. Числа A и B считываются из блока данных. Числа X и Y выдаются генератором случайных чисел в диапазоне [20,

50]. Определить, являются или нет НОД чисел A и B и НОД чисел X и Y взаимно простыми.

Задача 58. Написать программу печати чисел Фибоначчи, являющихся простыми числами, а также их порядковых номеров в ряду Фибоначчи на интервале до 1000.

Задача 59. Дано некоторое число N . Вывести на печать те делители N , которые являются числами Фибоначчи.

Задача 60. Вывести на печать все числа, меньшие 100, не являющиеся числами Фибоначчи.

Задача 61. Составить программу печати пифагоровых троек в заданном с клавиатуры диапазоне $[L,R]$.

ГЛАВА 6. РЕШЕНИЕ ФИЗИЧЕСКИХ ЗАДАЧ МЕТОДОМ КОМПЬЮТЕРНОГО МОДЕЛИРОВАНИЯ

6.1. МАТЕМАТИЧЕСКОЕ (КОМПЬЮТЕРНОЕ) МОДЕЛИРОВАНИЕ И ЭТАПЫ ВЫЧИСЛИТЕЛЬНОГО ЭКСПЕРИМЕНТА

Модели и моделирование играют чрезвычайно важную роль в деятельности человека. В сущности, всю совокупность знаний человека можно представить как модель материального мира, в котором отражен сам субъект моделирования.

Физические модели могут обеспечивать либо полное пространственно-временное подобие оригиналу (изучение движения корабля, самолета с помощью их уменьшенных моделей), либо только временное подобие (изучение электромагнитных явлений в цепях с сосредоточенными параметрами), либо только пространственное, геометрическое подобие (например, объемное проектирование).

Приближенное описание какого-либо класса явлений внешнего мира, выраженное с помощью математической символики, называется математической моделью. Для того чтобы получить количественную информацию о явлении, нужно найти адекватное математическое описание всех его существенных особенностей, которое и будет представлять собой математическую модель. Обычно это уравнение или система уравнений, описывающая элементарные процессы, из которых складывается исследуемое явление. Причем математическая модель – это не только уравнения математической задачи, но и дополнительные условия, устанавливающие границы их применимости. Все полученные с помощью этой модели теоретические результаты будут справедливы только в границах применимости.

Модель – это не только и не столько внешнее сходство. Главное лежит глубже: поведение модели и реального объекта должно подчиняться одинаковым закономерностям. Изучив их на доступной для исследования модели, оказывается возможным предска-

зять свойства или прогнозировать развитие физического процесса или явления, например, проектируемой конструкции или в труднодоступной области.

Процедура построения математической модели какого-либо реального явления или процесса и численное решение его с помощью компьютеров называется математическим (компьютерным) моделированием, или вычислительным экспериментом.

Использование метода математического моделирования (или вычислительного эксперимента) как средства решения сложных прикладных проблем в случае каждой конкретной задачи имеет специфические особенности. Тем не менее всегда четко просматриваются общие характерные основные черты, позволяющие говорить о единой структуре этого процесса. Технологический процесс математического моделирования принято подразделять на ряд этапов.

Первый этап – построение математической модели

1. Для исследуемого объекта строится физическая модель с критическим анализом существующих экспериментальных данных, связанных с изучаемым процессом или явлением. В рассматриваемом явлении выделяются главные факторы, которые учитывают и второстепенные, которые на данном этапе исследования отбрасывают, т.е. проводится выделение существенных факторов и свойств объекта. Отметим, что природа гораздо богаче и разнообразнее в своих проявлениях, нежели любые модели, являющиеся лишь ее бледными копиями.

2. Формулируются допущения и определяются рамки применимости данной модели, в которых будут справедливы полученные на ее основе результаты.

3. Математическая постановка задачи, конструирование математической модели проводится физиками, математиками или другими специалистами, хорошо знающими данную предметную область. Модель записывается в математических терминах, как правило, в виде дифференциальных или интегро-дифференциальных уравнений или системы таких уравнений. Указываются дополнительные условия, представляющие собой начальные или граничные условия, которым должно удовлетворять искомое решение.

4. Проводится предварительное исследование математической модели:

- на корректность поставленной задачи;
- решаемость данной задачи;
- единственность решения .

В различных приближениях производится поиск возможных аналитических решений упрощенной математической модели.

Второй этап – выбор метода решения

На этом этапе необходимо осуществить выбор метода решения математической модели. Математические модели можно решать различными методами.

1. Метод реального эксперимента моделирования с помощью аналоговых вычислительных машин, к сожалению, в связи с бурным развитием ЭВМ отошел на второй план. Однако отметим, что появились информационные системы, в частности, Electronics Workbench, позволяющие на ЭВМ имитировать работу АВМ. В этом пакете имеется возможность решать и анализировать математические модели с помощью имитаторов электронных приборов и логических схем.

2. Метод прямого программирования. Это направление связано с разработкой метода вычисления сформулированной математической задачи (алгоритма), т.е. построение разностной модели, которая представляет собой совокупность цепочек алгебраических формул, по которым ведутся вычисления, и логических условий, позволяющих установить нужную последовательность применения этих формул.

3. Использование различных информационных технологий, например, пакетов Maple, MathCad, Mathematics, Excel и т.д. для решения математических задач, в которых можно провести компьютерное моделирование реальных процессов и явлений. Однако их возможности часто при решении сложных задач бывают ограниченными.

Если ориентируемся на решение исходной задачи методом прямого программирования, то для этого необходимо создать вычислительный алгоритм математической модели. Как правило, для одной и той же математической задачи можно предложить великое множество вычислительных алгоритмов. Каждый может разработать свой алгоритм для данной задачи. Возникает вопрос (сомнение), насколько решение задачи по разработанному алгоритму

совпадает с истинным решением задачи, т.е. каково качество алгоритма? Таким образом, возникает вопрос, как отличать хорошие алгоритмы от плохих, не тратя времени и труда на программирование и расчеты, по их внешнему виду. Определение критериев для оценки качества вычислительных алгоритмов для различных задач и составляет предмет теории численных методов – раздела вычислительной математики.

Общая цель этой теории – построение эффективных вычислительных методов, которые позволяют получить решение поставленной задачи с заданной точностью за минимальное количество действий (арифметических, логических), т.е. с минимальными затратами машинного времени.

После правильного выбора численного алгоритма, перед его реализацией на ЭВМ необходимо теоретически оценить его качество и провести исследования:

- на устойчивость алгоритма;
- на сходимость алгоритма;
- на точность вычислений;
- на экономичность вычислений.

Здесь в любом случае необходимо пройти через эти стадии для понимания сущности выбранного алгоритма.

Для реальных задач получить решение соответствующей математической модели в виде аналитической формулы, содержащей явную зависимость от параметров, чаще всего не удается. В отличие от аналитической процедуры решения, вычислительный эксперимент имеет многовариантный характер, так как при использовании методов вычислительного эксперимента каждый конкретный расчет проводится при фиксированном значении параметров. Если параметров много, то приходится проводить большое число расчетов однотипных вариантов задачи, отличающихся значениями некоторых параметров. Вот почему, проводя вычислительный эксперимент, важно опираться на эффективные численные методы.

Третий этап – разработка и применение программного обеспечения

Это создание программы для реализации разработанного алгоритма на машинных языках. Существует много языков программирования, каждый из которых ориентирован на определенный

тип машины, на свой класс задач. Можно перечислить наиболее распространенные языки программирования, это Бейсик, Фортран, Паскаль, СИ, Ассемблер.

Программное обеспечение (или математическое обеспечение) современной ЭВМ представляет собой сложное хозяйство, включающее языки, трансляторы, операционные системы, библиотеки стандартных программ, вспомогательные информационные системы, пакеты прикладных программ и прочие программные продукты.

Современное программирование – это самостоятельная наука с фундаментальными принципами, подходами и методами. Таким образом, проведение вычислительного эксперимента требует от специалиста по математическому моделированию знаний языков программирования и навыков работы на вычислительных машинах.

После создания программы согласно разработанному алгоритму необходимо отладить программу, т.е. отыскать и исправить все ошибки и опечатки, допущенные как при создании алгоритма, так и при его программной реализации. В этих предварительных расчетах тестируется также сама математическая модель, выясняется, насколько хорошо она описывает изучаемый класс явлений, в какой степени она адекватна реальности. Существуют различные методы тестовой проверки:

1) использование при математическом моделировании реальных процессов, законов сохранения (интегралов движения). Эти законы являются вспомогательными уравнениями, которые помогают при решении дифференциальных уравнений протестировать полученное численное решение, т.е. являются контрольными уравнениями, проверяющими правильность решения на ЭВМ;

2) использование сравнения результатов расчета на ЭВМ с экспериментальными надежными данными по изучаемому явлению;

3) использование сравнения результатов моделирования с возможными аналитическими решениями упрощенной модели.

Эти методы обычно используются в совокупности. Сопоставление результатов тестовой проверки позволяет уточнить математическую модель, обрести уверенность в правильности предсказаний, которые будут получены с ее помощью.

Четвертый этап – компьютерное исследование, или вычислительный эксперимент

На этом этапе проводятся расчеты на ЭВМ, где наиболее отчетливо проявляется его сходство с экспериментом натурным, и этот этап часто называют вычислительным экспериментом. Если в лаборатории экспериментатор с помощью специально построенной установки задает вопросы природе, то специалисты по вычислительному эксперименту с помощью ЭВМ ставят эти вопросы математической модели. Ответ в обоих случаях получается в виде некоторой цифровой информации, которую предстоит еще расшифровать. Причем в современных физических экспериментах со сложными объектами или процессами, протекающими в экстремальных условиях, каждое измерение температуры, плотности, скорости и т.д. дается с большим трудом. Зачастую нужную информацию приходится извлекать из косвенных данных. Точность полученных результатов, как правило, невелика.

Иное дело вычислительный эксперимент. ЭВМ в процессе расчета может выдавать любую информацию, необходимую исследователю. Конечно, точность этой информации определяется достоверностью самой модели. Именно по этой причине в серьезных прикладных исследованиях никогда не начинают вести полномасштабные или, как говорят, производственные расчеты по новой программе.

Проведение расчетов – это исследовательская работа, поиск, имеющий свою стратегию. Фактически в процессе расчетов осуществляется диалог «человек – компьютер».

После проведения длительной, кропотливой работы в вычислительном эксперименте наступает фаза прогноза, с помощью математического моделирования предсказывается поведение исследуемого объекта в условиях, где эксперименты пока не проводились или где они вообще невозможны.

Пятый этап – обработка и анализ результатов вычислительного эксперимента

Обработка результатов расчетов, их всесторонний анализ и, наконец, выводы составляет сущность пятого этапа. Здесь используются различные статистические, графические и другие информа-

ционные системы (программы, приложения Windows). В параграфе 8.1. мы приводим технологию обработки полученной информации в системе ORIGIN.

Выводы, полученные на этом этапе, бывают в основном двух типов: или возникает необходимость уточнения модели; или результаты, пройдя проверку на разумность и надежность, публикуются или передаются заказчику на использование.

Однако чаще всего эти две стороны переплетаются: выясняются какие-либо необычные формы протекания изучаемого процесса, неожиданные режимы работы проектируемой установки, в результате чего появляется желание уточнить те или иные детали процесса. Математическая модель модифицируется, как правило, усложняется, и начинается новый цикл вычислительного эксперимента со всеми его этапами. В этом случае говорят о многомодельности вычислительного эксперимента.

Таким образом, технологический цикл математического и компьютерного моделирования реальных объектов, явлений и процессов может быть представлен схематически как (см. рис. 34).

Все перечисленные этапы ориентированы на решение какой-либо физической, биологической, экономической, технической, педагогической и других задач методом математического (компьютерного) моделирования и являются условными.

Освоение численных методов решения дифференциальных уравнений или системы дифференциальных уравнений является одним из необходимых навыков решения физических задач. Однако надо осознать, что временные затраты на разработку программного обеспечения измеряются иногда не в днях, а в месяцах. Поэтому в некоторых случаях имеет смысл воспользоваться пакетами для математических расчетов типа MAPLE, MATHCAD, MATLAB и др. В пособии в краткой форме даются также технологии решения математической модели в пакетах Excel, Maple, MathCad.

После проведения длительной, кропотливой работы в вычислительном эксперименте наступает фаза прогноза, с помощью математического моделирования предсказывается поведение исследуемого объекта в условиях, где эксперименты пока не проводились или где они вообще невозможны. Эта фаза является самой интересной, творческой и сложной, так как получение совершенно новых результатов в исследуемой области может привести к новым открытиям и, самое главное, моральной удовлетворенности исследователя.

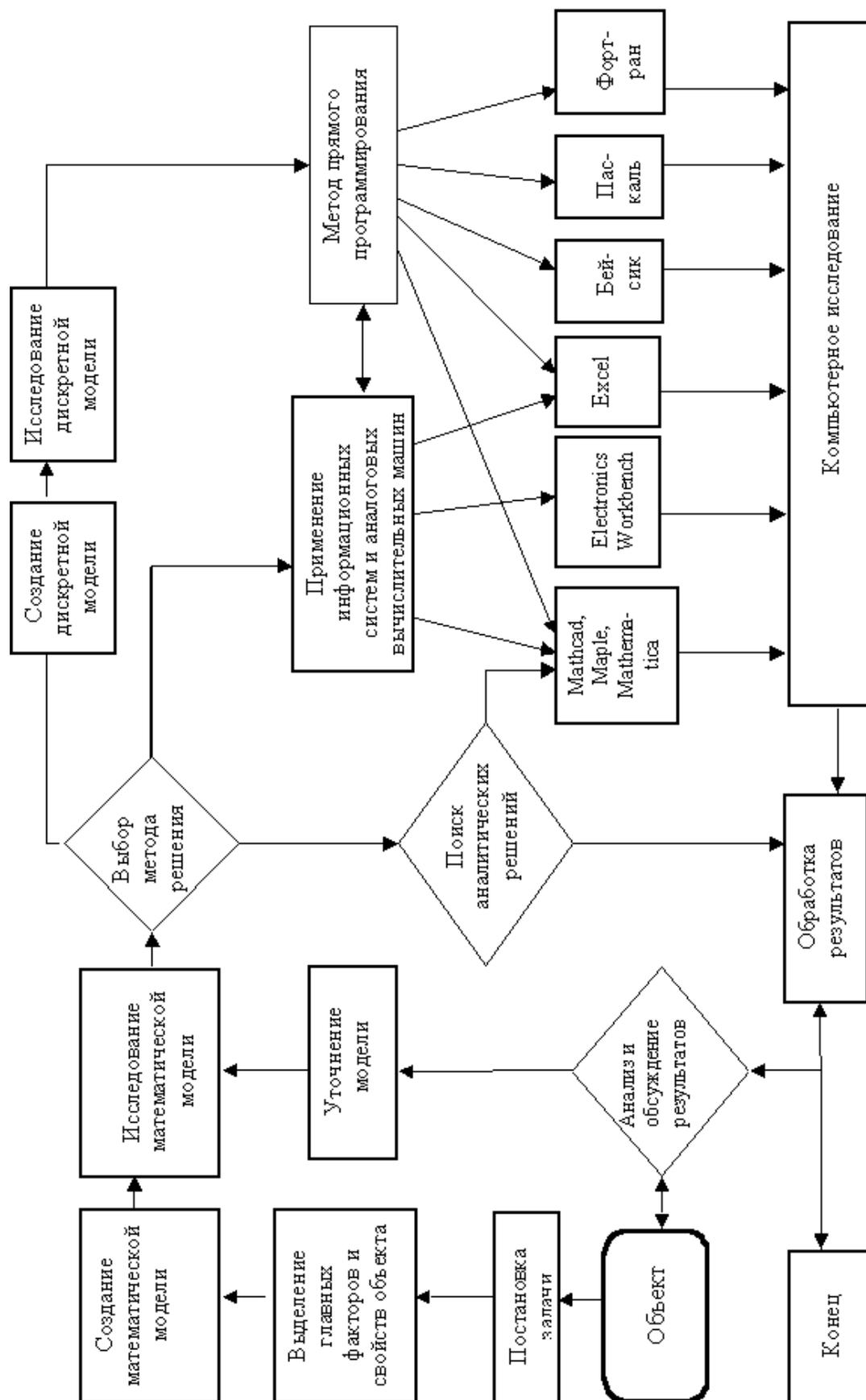


Рис. 34. Схема математического и компьютерного моделирования реальных процессов и явлений

Ниже приведены физические задачи для отработки навыков решения физических задач методом математического моделирования с использованием ЭВМ. Программы решения задач написаны на языке Паскаль. Графические результаты оформлены с помощью пакета ORIGIN.

В седьмой главе приведены технологии решения такого рода задач с помощью информационных систем Excel и MathCad.

6.2. МОДЕЛИРОВАНИЕ ДВИЖЕНИЯ ТЕЛА, БРОШЕННОГО ПОД УГЛОМ К ГОРИЗОНТУ

Рассмотрим движение материального тела, брошенного в реальной среде под углом к горизонту. Эта классическая задача приводится во многих учебниках и книгах [4-6, 15–17, 73]. Проведем математическое и компьютерное моделирование этого процесса с учетом всех этапов вычислительного эксперимента, следуя работе [73].

1. Постановка физической задачи. Процесс математического моделирования начинается с постановки физической модели, с критическим анализом существующих экспериментальных данных, связанных с изученным процессом или явлением. Пусть материальное тело массой m брошено из пункта O под углом α к горизонту с начальной скоростью v_0 (рис.35).

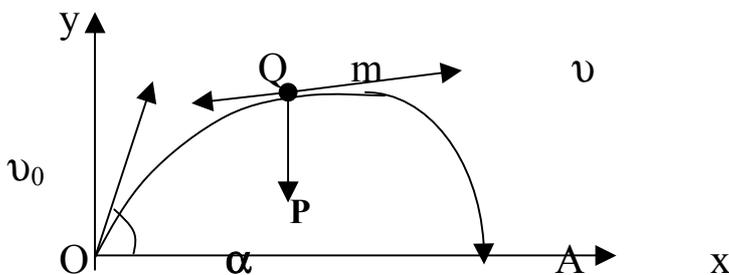


Рис.35

Точка движется в реальной среде, оказывающей сопротивление движению с силой Q , пропорциональной скорости движения. Необходимо воспроизвести на ЭВМ движение материального тела и исследовать его, считая, что место бросания O и место падения A лежат на одной высоте.

2. Конструирование математической модели (вывод дифференциальных уравнений движения). В декартовой системе координат xOy ось x проходит через точки O и A . Закон Ньютона в проекции на координатные оси можно записать в виде

$$F_x = m\ddot{x}, \quad F_y = m\ddot{y}, \quad (2)$$

где F_x, F_y – проекции всех сил на оси координат, действующих на материальное тело, \dot{x}, \dot{y} – ускорения тела вдоль осей координат. На тело, движущееся под углом к горизонту, действуют две силы. Будем считать, что сила тяжести $F = mg$ равна весу тела $F=P$, а сила сопротивления среды $Q = kmv$, где k – коэффициент пропорциональности. Дифференциальные уравнения движения тела получаются из (2) после подстановки туда выражений проекций сил P и Q на координатные оси:

$$F_x = P_x + Q_x, \quad F_y = P_y + Q_y. \quad (3)$$

Проекции сил равны:

$$F_x = 0, \quad F_y = -mg, \quad Q_x = -km\dot{x}, \quad Q_y = -km\dot{y}.$$

Подставляя в (2), получим систему дифференциальных уравнений, которая является математической моделью движения тела, брошенного под углом к горизонту.

$$\begin{aligned} \ddot{x} &= -k\dot{x}, & \dot{x}(0) &= v_0 \cos \alpha, & x(0) &= 0, \\ \ddot{y} &= -k\dot{y} - g, & \dot{y}(0) &= v_0 \sin \alpha, & y(0) &= 0. \end{aligned} \quad (4)$$

3. Законы сохранения (интегралы движения). Они помогают при решении дифференциальных уравнений протестировать полученное численное решение, т.е. являются контрольными уравнениями, проверяющими правильность решения на ЭВМ. Законы сохранения являются дополнениями к дифференциальным уравнениям. В нашей задаче для контроля правильности работы ЭВМ используется метод избыточной переменной. Введем вспомогательную избыточную переменную $z(t)$ и потребуем, чтобы она вместе с переменными $x(t), y(t), \dot{x}(t), \dot{y}(t)$ удовлетворяла некоторому контрольному соотношению. В данной задаче оно имеет вид:

$$kx + \dot{x} + ky + \dot{y} - z = 0. \quad (5)$$

Способ воспроизведения избыточной переменной $z(t)$ можно найти, дифференцируя (5) по t . Тогда для $z(t)$ определяющее дифференциальное уравнение имеет вид:

$$\dot{z} = k\dot{x} + \ddot{x} + k\dot{y} + \ddot{y}. \quad (6)$$

Действительно, если подставить (4) в (6), получим выражение для производной по Z и окончательную систему дифференциальных уравнений (математическую модель), описывающих движение тела, брошенного под углом к горизонту:

$$\begin{aligned} \dot{z} &= -g, \\ \ddot{x} &= -k\dot{x}, \\ \ddot{y} &= -k\dot{y} - g, \\ kx + \dot{x} + ky + \dot{y} - z &= 0. \end{aligned} \quad (7)$$

Начальные условия для задачи имеют вид

$$\begin{aligned} z(0) &= v_0 (\cos \alpha + \sin \alpha), \\ \dot{x}(0) &= v_0 \cos \alpha, \quad x(0) = 0, \\ \dot{y}(0) &= v_0 \sin \alpha, \quad y(0) = 0; \end{aligned}$$

4. Выбор масштабов физических величин является одним из важных моментов при моделировании на ЭВМ. Масштабы – размерные коэффициенты пропорциональности между математическими переменными и машинными. Различают масштабирование зависимых и независимых переменных. Масштабы должны быть выбраны так, чтобы машинные переменные были порядка единицы. В нашем случае, чтобы рассчитать масштабы, необходимо располагать значениями:

$$x_{\max}, \dot{x}_{\max}, y_{\max}, \dot{y}_{\max}, z_{\max}.$$

Откуда их взять, если отсутствует аналитическое решение (7)? Максимальные значения переменных нужно отыскать, минуя аналитическое решение системы (7). Здесь на помощь приходит понимание физической сущности процесса. Вместо заданного процесса, описываемого «сложными» дифференциальными уравнениями (7), выбирают другой процесс со сходным физическим содержанием, но с «простым» математическим описанием, аналитическое исследование которого позволяет оценить границы изменения переменных в моделируемом процессе. Этот вспомогательный процесс называ-

ют масштабной задачей. Масштабной задачей в нашем случае может служить процесс движения материального тела, брошенного под углом к горизонту в отсутствие сопротивления среды. В пустоте нет препятствий движению, и тело будет двигаться быстрее и перемещаться дальше, чем в сопротивляющейся среде. Поэтому найденные максимальные значения скоростей \dot{x}, \dot{y} и перемещений x, y для тела, которое движется в пустоте, будут пригодны для масштабирования переменных, описывающих движения тела в сопротивляющейся среде. Движению тела в отсутствие сопротивления внешней среды соответствует значение $k=0$, и потому система дифференциальных уравнений (7) примет вид:

$$\begin{aligned}\ddot{x} &= 0, \dot{x}(0) = v_0 \cos \alpha, x(0) = 0; \\ \ddot{y} &= -g, \dot{y}(0) = v_0 \sin \alpha, y(0) = 0.\end{aligned}\quad (8)$$

Уравнения (8) имеют простые аналитические решения:

$$x(t) = (v_0 \cos \alpha)t, \quad y(t) = (v_0 \sin \alpha)t - \frac{gt^2}{2},$$

из которых следует:

1) для любого угла бросания α величины \dot{x}, \dot{y} проекций скорости v на координатные оси x и y не превосходят величины начальной скорости v_0 ;

2) максимальная дальность полета x_{\max} материального тела соответствует углу бросания $\alpha = \frac{\pi}{4}$ и равна $\frac{v_0^2}{g}$;

3) максимальная высота подъема y_{\max} тела соответствует углу бросания $\alpha = \frac{\pi}{2}$ и равна $\frac{v_0^2}{2g}$;

4) максимальное значение $z_{\max} < 2v_0$.

Расчет масштабов выполнен в таблице.

Переменная	Максимальное значение переменной	Величина масштаба	Обозначение масштаба
x, y	$\frac{v_0^2}{g}$	$\frac{100g}{v_0^2}$	m_0
\dot{x}, \dot{y}	v_0	$\frac{100}{v_0}$	m_1
z	$2v_0$	$\frac{100}{2v_0}$	m_z

В результате масштабирования (применения теории подобия) мы имеем такую же систему, как (7), но вместо натуральных величин x, \dot{x}, k, g, v имеем масштабированные величины. Обычно масштабированные величины имеют порядок единицы.

5. Алгоритм решения. Метод численного решения уравнений (7) заключается в замене их разностными аналогами. Разностная система уравнений

$$\begin{aligned} \frac{x_{i+1} - 2x_i + x_{i-1}}{h^2} &= -kx_{i-1}, & x_1 &= v_0 \cos \alpha, & x_0 &= 0; \\ \frac{y_{i+1} - 2y_i + y_{i-1}}{h^2} &= -k \frac{y_i - y_{i-1}}{h} - g, & y_1 &= v_0 \sin \alpha, & y_0 &= 0 \end{aligned} \quad (9)$$

является дискретным аналогом математической модели (7) движения тела, брошенного под углом к горизонту. Решая их, мы на самом деле решаем совершенно другие уравнения. Однако говорим, что решения этих уравнений являются решениями исходного уравнения. Насколько они удовлетворяют, точнее, приближаются (аппроксимируются) к решению исходной системы уравнений (7), определяется выбранным методом численного решения. Эта задача решается в теории разностных численных методов [8, 38, 41, 46, 73]. Из уравнений (9) получаем алгоритм численного решения математической задачи (7):

$$\begin{aligned} x_{i+1} &= 2x_i - x_{i-1} - h^2 kx_{i-1} \\ y_{i+1} &= 2y_i - y_{i-1} - kh(y_i - y_{i-1}) - h^2 g \end{aligned} \quad (10)$$

6. Программа численного решения.

```
PROGRAM GORIZ;
const n=1000;g=9.8;
var i:integer;
    h,k:real;
    x,y: array [0..n] of real;
    gor:text;
begin h:=0.05;k:=1;
x[0]:=1;x[1]:=2;
y[0]:=1;y[1]:=2;
for i:=1 to (n-1) do begin
    x[i+1]:=2*x[i]-x[i-1]-k*h*h*x[i];
    y[i+1]:=2*y[i]-y[i-1]-k*h*(y[i]-y[i-1])-g*h*h
end;
assign(gor,'d:\gorizon.dat');
rewrite(gor);
for i:=0 to n do begin
writeln(gor,x[i],', ',y[i]);end;
close(gor);
end.
```

7. Результаты моделирования. На рис.36 приведены результаты моделирования движения тела, брошенного под углом к горизонту при разных значениях сопротивления среды.

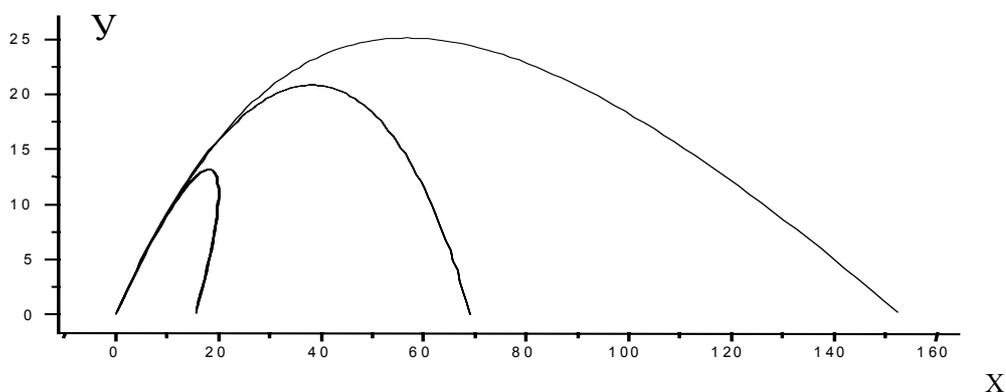


Рис. 36

Контрольные вопросы и задания к лабораторной работе.

1. Какие этапы математического моделирования вы знаете?
2. Что такое масштабирование уравнений?
3. Как получить разностные уравнения (10)?
4. Провести вычислительный эксперимент по исследованию траектории тела в зависимости от начальной высоты, скорости, угла полета и коэффициента сопротивления среды.
5. Промоделировать задачу попадания мяча в баскетбольное кольцо.

6.3. ДВИЖЕНИЕ ТЕЛА С УЧЕТОМ СОПРОТИВЛЕНИЯ СРЕДЫ

1. Постановка задачи. В реальных ситуациях любое тело движется в газовой или жидкой среде и, следовательно, испытывает сопротивление среды. Требуется выявить закономерности движения тела в реальной среде.

2. Построение математической модели. Будем считать, что на движущееся тело в газовой или жидкостной среде действует только сила трения.

Закономерности, которые проявляются в таких системах, носят эмпирический характер. Сила сопротивления среды зависит от скорости: при малых скоростях величина силы сопротивления пропорциональна скорости и имеет вид

$$F_{\text{сопр}} = k_1 v, \quad (11)$$

где k_1 – определяется свойствами среды и формой тела. Для шарика $k_1 = 6\pi\mu r$, где μ – динамическая вязкость среды, r – радиус шарика. Для воздуха при $t = 20^\circ \text{C}$ и давлении 1 атм $\mu = 0.0182 \text{ Н с/м}^2$, для воды $\mu = 1.002 \text{ Н с/м}^2$, для глицерина $\mu = 1480 \text{ Н с/м}^2$.

При высоких скоростях сила сопротивления становится пропорциональной квадрату скорости:

$$F_{\text{сопр}} = k_2 v^2, \quad (12)$$

где $k_2 = 0.5 c S \rho_{\text{среды}}$, здесь S – площадь сечения тела, поперечного по отношению к потоку, $\rho_{\text{среды}}$ – плотность среды, c – безразмерный коэффициент лобового сопротивления.

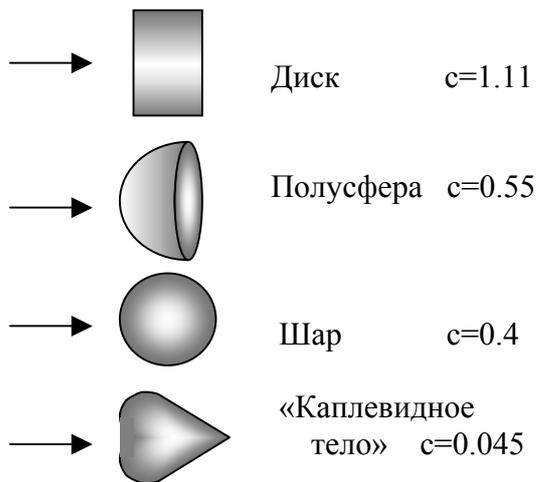


Рис. 37

ны значения коэффициента лобового сопротивления для некоторых тел.

Из рассмотрения сил, действующих на тело, получим уравнения движения для тела, падающего с большой высоты:

$$\begin{aligned} m dv/dt &= mg - k_2 v^2 - k_1 v, \\ v &= dx/dt. \end{aligned} \quad (13)$$

Эти уравнения составляют математическую модель движения тела в газообразной или жидкой среде.

3. Алгоритм и программа решения.

Заменим систему уравнений (13) разностной системой уравнений:

$$\frac{v_{i+1} - v_i}{h} = g - (k_2 v_i^2 + k_1 v_i) / m, \quad v_0 = 0; \quad (14a)$$

$$\frac{x_{i+1} - x_i}{h} = v_i \quad x_0 = H_0; \quad \dots (14b)$$

Система уравнений (14) является дискретным аналогом математической модели (13) движения тела, падающего под действием силы тяжести. Из уравнений (14) получаем алгоритм численного решения математической задачи (13).

$$\begin{aligned} x_{i+1} &= x_i + h v_i \\ v_{i+1} &= v_i + h (g - (k_1 v_i + k_2 v_i^2) / m) \end{aligned} \quad (15)$$

4. Программа численного решения.

```
Program a;                               Begin
Const                                     v[i+1]:=h*g-k2*h*v[i]*v[i]/m+v[i];
n=5000;g=9.8;M=60;h=0.2;                 x[i+1]:=-h*v[i]+x[i];
Label 1;                                  if x[i]<200 then begin
Var c,S,q,k2:real;                         s:=20;c:=0.4;q:=1.;k2:=0.5*c*s*q;en
  i,p,r:integer;                            d;
  X,V:array [0..n] of real;                 if x[i]<0 then begin
  Dnt:text;                                  x[i]:=0;v[i]:=0;end;
Begin                                         if x[i]=0 then begin
Assign(dnt,'c:\ayrat.dat');                 x[i+1]:=0;v[i+1]:=0;end;
Rewrite(dnt);                                end;
x[0]:=n*h;v[0]:=0;c:=0.44;S:=0.9;          For i:=0 to n do
q:=1.29;                                       begin
k2:=0.5*c*S*q;                               writeln(dnt,i*h,' ',x[i],' ',v[i]);
for i:=0 to n-1 do                           {writeln(dnt,i,' ',R[i]);}
end;
close(dnt);
end.
```

5. Задания к лабораторной работе.

Эту задачу можно провести либо методом прямого программирования, либо в пакете MathCAD. Ниже даны задания для проведения исследований по первому методу.

1. Построить дискретную модель уравнений и разностную схему. Написать алгоритм решения дифференциальных уравнений (13).

2. Составить программу согласно алгоритму. Выходные данные представить в графической форме. Рекомендуется использовать графический пакет ORIGIN.

3. Задания для вычислительного эксперимента.

3.1. Вычислить, на какой высоте (в какой момент времени) парашютисту надо открыть парашют, чтобы иметь к моменту приземления безопасную скорость (не более 10 м/с). Построить график зависимости скорости и пути от времени.

3.2. Установить, как должна быть связана высота прыжка и рабочая площадь парашюта, чтобы скорость приземления была безопасной.

3.3. Рассмотреть падение тел с заданными характеристиками (масса, форма) в средах разной плотности.

6. Результаты моделирования. На рис.38 представлены два графика прыжка парашютиста с разных высот и с разными площадями парашюта.

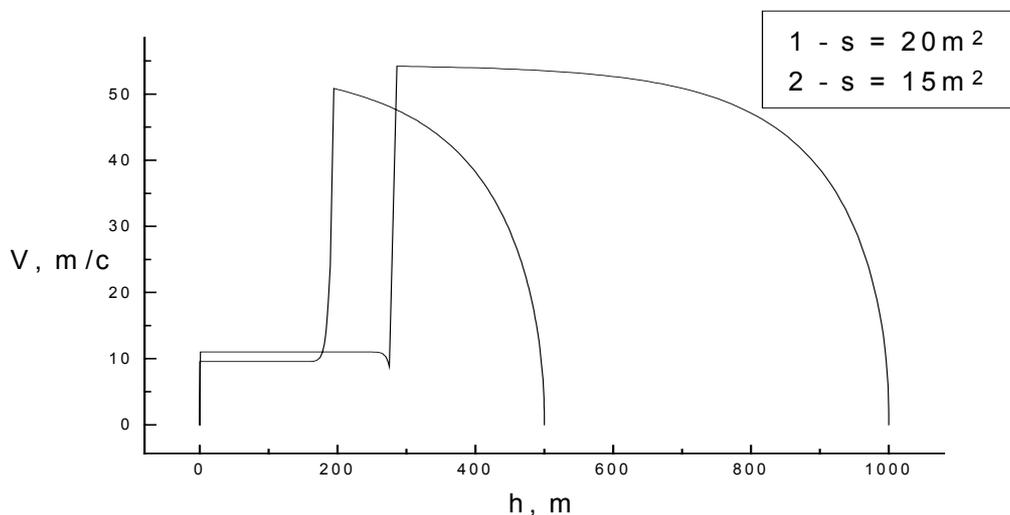


Рис. 38

6.4. ДВИЖЕНИЕ ШАРИКА В ВЯЗКОЙ СРЕДЕ

1. Постановка задачи. Рассмотреть задачу падения шарика под действием силы тяжести в вязкой среде (глицерин, масло). Требуется выявить закономерности движения шарика в реальной вязкой среде.

2. Математическая модель. Будем считать, что на движущийся шарик в газовой или жидкой среде действуют только сила трения и сила тяжести. Величина силы сопротивления среды при малых скоростях пропорциональна скорости и имеет вид

$$F_{\text{сопр}} = k_1 v, \quad (16)$$

Для шарика $k_1 = 6\pi\mu r$, где μ – динамическая вязкость среды, r – радиус шарика.

Из рассмотрения сил, действующих на шарик, можно получить следующие уравнения движения шарика в вязкой среде:

$$\begin{aligned} m dv/dt &= mg - k_1 v, \\ v &= dx/dt. \end{aligned} \quad (17)$$

Эти уравнения и составляют математическую модель движения тела в вязкой среде.

3. Задания к лабораторной работе.

3.1 Привести уравнения к безразмерному виду.

3.2. Записать дискретную модель уравнений и программу решения дифференциальных уравнений (17). Дискретная модель аналогична уравнениям (14).

3.3. Составить программу согласно алгоритму. Выходные данные представить в графической форме. Рекомендуется использовать графический пакет ORIGIN.

3.4. Задания для вычислительного эксперимента.

Свинцовый шарик массой $m = 0,54$ г падает в глицерине (плотность свинца $\rho_c = 11 \cdot 10^3 \text{ кг/м}^3$, плотность масла $\rho_m = 0,92 \cdot 10^3 \text{ кг/м}^3$, вязкость масла $\mu = 950 \text{ Н с/м}^2$, вязкость глицерина $\mu = 1480 \text{ Н с/м}^2$) с начальной скоростью v_0 , направленной по вертикали. Постройте графики зависимости скорости, ускорения и перемещения шарика от времени. Варьируя v_0 , установите с помощью компьютерной модели характерные особенности движения шарика (решение этой задачи с помощью системы Excel см. в гл. 7).

6.5. ЗАДАЧА ДВУХ ТЕЛ

1. **Постановка задачи.** Рассмотрим движение космического тела (планеты, кометы, спутника) в гравитационном поле, создаваемом телом большой массы.

2. **Построение математической модели.** Движение любого тела в гравитационном поле описывается с помощью второго закона Ньютона

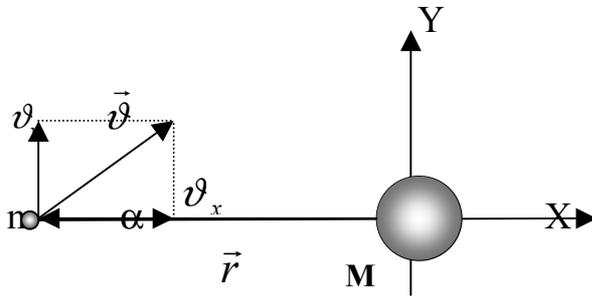


Рис.39

Отметим, что знак «минус» в уравнении движения возникает потому, что сила стремится уменьшить расстояние между телами. В системе координат, начало которой привязано к большому телу, уравнения модели имеют вид:

$$\begin{aligned} mdv_x/dt &= -GMm x/(x^2+y^2)^{3/2} \\ dx/dt &= v_x \\ mdv_y/dt &= -GMm y/(x^2+y^2)^{3/2} \\ dy/dt &= v_y \end{aligned} \quad (20)$$

Начальные условия определяются двумя параметрами: начальной скоростью и углом α .

$$\begin{aligned} x(0) &= x_0, \quad y(0) = 0 \\ v_{x0} &= v_0 \cos\alpha, \quad v_{y0} = v_0 \sin\alpha \end{aligned} \quad (21)$$

3. Дискретная модель.

Заменим систему уравнений (18) разностной системой уравнений:

$$\begin{aligned} \frac{v1_{i+1} - v1x_i}{h} &= -GM \frac{x_i}{(x_i^2 + y_i^2)^{3/2}} \\ \frac{x_{i+1} - x_i}{h} &= v1_i \end{aligned} \quad (22a)$$

$$\frac{v_{2_{i+1}} - v_{2_i}}{h} = -GM \frac{y_i}{(x_i^2 + y_i^2)^{3/2}}$$

$$\frac{y_{i+1} - y_i}{h} = v_{2_i}$$
(22б)

$$x(0) = x_0, \quad y(0) = 0,$$

$$v_{1_0} = v_0 \cos \alpha, \quad v_{2_0} = v_0 \sin \alpha$$
(22в)

Система уравнений (22) является дискретным аналогом математической модели (20). Ниже приведена программа расчета, где использован алгоритм по методу прогноза и коррекции.

4. Программа численного решения.

```

program aa;
const n=2000;G=6.67;M=2;h=0.033;
var i:integer;
    f1,f2,v11,v21,x1,y1,a,v0,alfa:real;
    X,Y,V1,V2,R:array [0..n] of real;
    dnt:text;
Begin
assign(dnt,'c:\ast1.dat');
rewrite(dnt);
a:=30;
alfa:=a*3.14/180; v0:=0.7;
x[0]:=-10;y[0]:=0;v1[0]:=v0*cos(alfa);v2[0]:=v0*sin(alfa);
for i:=0 to n-1 do
begin
f1:=G*M*x[i]/(sqrt(x[i]*x[i]+y[i]*y[i]))*(x[i]*x[i]+y[i]*y[i]));
v11:=v1[i]-h*f1;
f2:=G*M*y[i]/(sqrt(x[i]*x[i]+y[i]*y[i]))*(x[i]*x[i]+y[i]*y[i]));
v21:=v2[i]-h*f2;
x1:=x[i]+h*v11;
y1:=y[i]+h*v21;
v1[i+1]:=v1[i]-0.5*h*(f1+G*M*x1/(sqrt(x1*x1+y1*y1)*(x1*x1+y1*y1)));
v2[i+1]:=v2[i]-0.5*h*(f2+G*M*y1/(sqrt(x1*x1+y1*y1)*(x1*x1+y1*y1)));
x[i+1]:=x[i]+0.5*h*(v11+v1[i]);
y[i+1]:=y[i]+0.5*h*(v21+v2[i]);
end;
For i:=0 to n do
begin

```

```

        writeln(dnt,x[i], ' ',y[i]);
    end;
close(dnt);
end.

```

5. Задания к лабораторной работе.

1. Написать алгоритм решения дифференциальных уравнений (20). Рекомендуется использовать графический пакет ORIGIN.

2. Задания для вычислительного эксперимента.

2.1. Найти траекторию движения кометы, залетевшей в солнечную систему, находящейся на расстоянии от Солнца в 100 астрономических единиц ($1 \text{ а.е.} = 1,5 \cdot 10^{11} \text{ м}$ – расстояние от Солнца до Земли). Скорость кометы равна 2 км/с и направлена под углом 30° к оси «Комета-Солнце». Изменяя скорость, определить, при каком условии траектория будет замкнутой. Определить период полета кометы.

2.2. Проверить в компьютерном эксперименте выполнимость трех законов Кеплера.

6. Результаты моделирования. На графиках рис. 40, 41, 42, 43 представлены траектории движения космического тела (кометы) в зависимости от его начальной скорости в гравитационном поле, создаваемом телом с многократно большей массой (Солнца), при заданных начальных параметрах: проекций радиус-вектора на оси X и Y ($x[0]$, $y[0]$), начальной скорости космического тела (v_0), массы тела, создающего гравитационное поле (M). Точка пересечения линий, параллельных осям координат, показывает положение тела с многократно большей массой (Солнца).

Из рис. 40 и 41 видно, что по мере уменьшения начальной скорости космического тела (кометы) его траектория движения все больше закручивается. При начальной скорости 1.2 км/с космическое тело движется по траектории, представляющей собой эллипс, в одном из фокусов которого находится тело с многократно большей массой (Солнце). Это видно из рис.41.

Из сравнения рис. 42 и 43 видно, что когда начальная скорость космического тела меньше начальной скорости при движении по эллиптической орбите, то оно падает на тело с многократно большей массой, приближаясь к нему по спирально-эллиптической траектории. Это объясняется тем, что начальная скорость косми-

ческого тела недостаточна для того, чтобы преодолеть гравитационное поле тела многократно большей массы (Солнца).

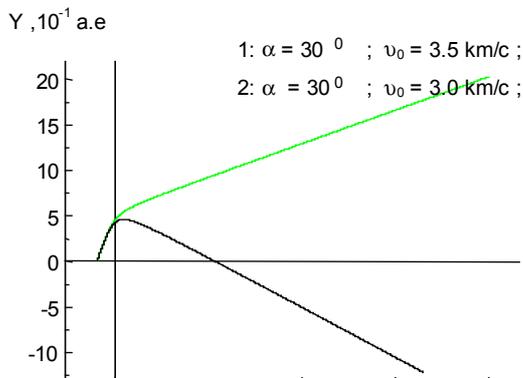


Рис. 40

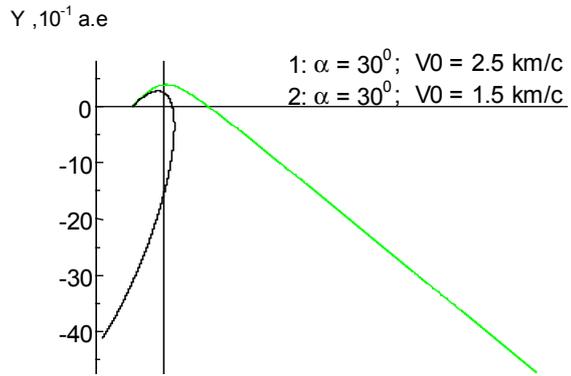


Рис. 41

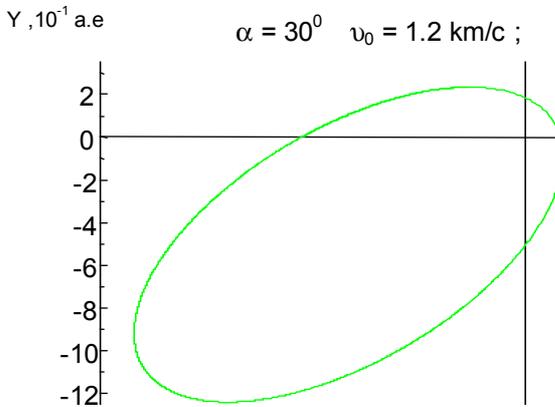


Рис. 42

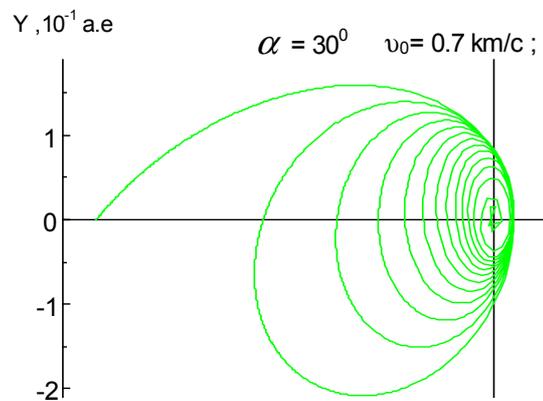


Рис. 43

6.6. ОДНОСТУПЕНЧАТАЯ РАКЕТА

1. **Постановка задачи.** Построить модель вертикального взлета одноступенчатой ракеты (рис. 44).

2. **Построение математической модели.** Математическую модель движения ракеты можно получить на основе закона сохранения импульса. Рассмотрим закон сохранения импульса. За промежуток времени dt между моментами времени t и $t + dt$ часть топлива выгорела, и масса ракеты уменьшилась на dm , изменился импульс ракеты. Однако суммарный импульс ракеты сократился (ракеты + продукт сгорания).

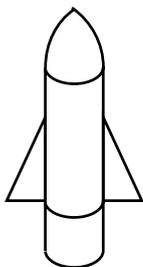


Рис. 44

$$m(t)v(t) = m(t + dt)v(t + dt) - dm[v(t + dt) - u], \quad (23)$$

где $v(t)$ – скорость ракеты, $v(t + dt) - u$ – средняя за промежуток времени dt скорость истекающих из сопл газов (относительно земли). Первое слагаемое в правой части уравнения (23) – импульс ракеты в момент $t + dt$, второе – импульс, переданный истекающим газом за время dt .

Подставляя

$$\begin{aligned}v(t + dt) &= v(t) + dv, \\m(t + dt) &= m(t) + dm\end{aligned}$$

в (23) и разделив обе части уравнения на dt , получим:

$$m \frac{dv}{dt} = -\frac{dm}{dt} u, \quad (24)$$

где $\frac{dm}{dt}$ – скорость изменения массы ракеты. При выводе уравнений (24) мы считали, что ускорение ракеты намного больше, чем ускорение свободного падения ($\frac{dv}{dt} \gg g$). Если взять простую зависимость массы ракеты от времени $m(t) = m_0 - \alpha t$, то

$$m \frac{dv}{dt} = \alpha u = F_{\text{тяги}}.$$

Для современных ракет скорость истечения газов из сопла принимает значения $u = (3 \div 5) \text{ km/s}$, а скорость изменения массы $\alpha = (2 \div 5) 10^5 \text{ кг/с}$.

Уравнение (24) является математической моделью движения тела с переменной массой без учета сил сопротивления. Это уравнение имеет аналитическое решение. Решение можно найти, преобразуя (24) к виду

$$\frac{dv}{dt} = -\frac{d \ln m}{dt} u.$$

Интегрируя это уравнение, получим

$$v(t) = v_0 + u \ln \frac{m_0}{m(t)}, \quad (25)$$

где v_0, m_0 – скорость и масса в начальный момент.

В общем случае математическая модель движения тел с переменной массой описывается уравнением Мещерского-Циолковского:

$$m(t) \frac{dv}{dt} = \alpha u - F_{\text{сопр}} - G \frac{M \cdot m(t)}{(R+h)^2}, \quad (26)$$

$$\frac{dh}{dt} = v.$$

Здесь

$$F_{\text{сопр}} = k_1 v + k_2 v^2, \quad k_1 = 6\pi\mu r, \quad k_2 = 0.5 cS\rho_{\text{среды}}, \quad (27)$$

где μ – динамическая вязкость среды, r – радиус цилиндра ракеты.

Для воздуха при $t = 20^0$ С и давлении 1 атм $\mu = 0.0182$ Н с/м², будем считать, что при взлете ракеты масса ее изменяется согласно закону

$$m(t) = \begin{cases} m_0 - \alpha t, & \text{если } m(t) \leq m_{\text{кон}} \\ m_{\text{кон}}, & \text{если } m(t) = m_{\text{кон}} \end{cases}, \quad (28)$$

и плотность воздуха, входящая в коэффициент сопротивления, убывает по закону

$$\rho = \rho_0 10^{-\beta h}, \quad \text{здесь } \beta = 5,6 * 10^{-5} \text{ м}^{-1},$$

m_0 – начальная масса ракеты, заправленной топливом. Под $m_{\text{кон}}$ понимается обычно полезная масса (масса спутника) и структурная масса (масса топливных баков, двигателей, систем управления и т.д.), т.е. та масса, которая остается после полного выгорания топлива, α – расход топлива.

3. Задания к лабораторной работе

1. Построить дискретную модель уравнений и разностную схему. Написать алгоритм решения дифференциальных уравнений (26).

2. Составить программу согласно алгоритму. Выходные данные представить в графической форме. Рекомендуется использовать графический пакет ORIGIN или GRAFER.

3. Задания для вычислительного эксперимента.

3.1. Провести моделирование взлета ракеты при значениях параметров $m_0=2\cdot 10^7$ кг, $m_{\text{кон}} = 2\cdot 10^5$ кг, $\alpha = 2\cdot 10^5$ кг/с, $u = (3 \div 5)$ км/с. Выяснить условия достижения первой космической скорости.

3.2. Провести исследование соотношения двух из входных параметров, при которых ракета достигнет первой космической скорости и в этот момент исчерпает горючее.

3.3. Построить фазовые диаграммы в переменных $(m(t), F)$.

4. Результаты моделирования.

На рис.45 показано изменение скорости ракеты от высоты. Программа масштабирована так, что скорость v измеряется в км/с, высота в км, масса измеряется в единицах: $m_m = 10^7$ кг, $m = 2 m_m$, $g=0,0098$ кг/с², $s=0,0001$ км², $u=5$ км/с, $\rho_0 = 129 m_m / \text{км}^3$ – плотность воздуха на поверхности Земли при нормальных условиях. Как видно из рисунка, на высоте 159 км топливо закончилось, максимальная скорость при этом $v = 7,6$ км/с. Затем ракета начинает терять высоту и в конечном итоге падает на землю.

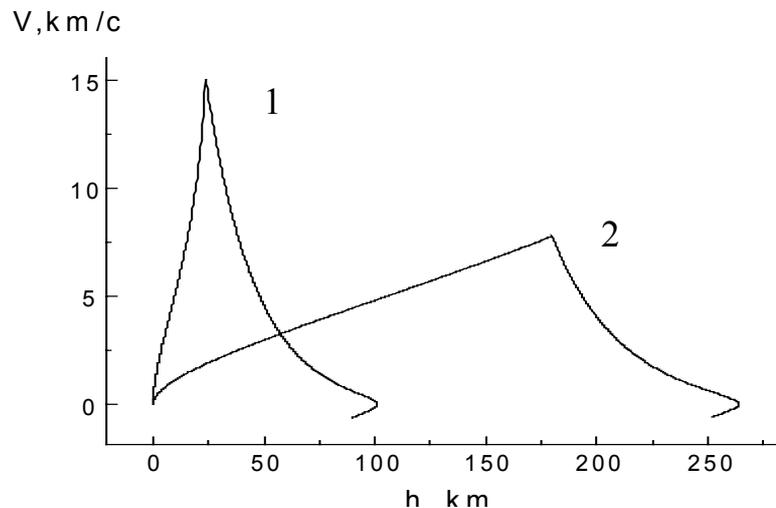


Рис. 45. Скорость корабля при разных расходах горючего:
1) $\alpha=0.25 m_m/c$; 2) $\alpha=0.3 m_m /c$

6.7. ДВИЖЕНИЕ ЗАРЯЖЕННЫХ ЧАСТИЦ В КУЛОНОВСКОМ ПОЛЕ

1. Постановка задачи. Рассмотреть движение частицы с зарядом q и массой m в кулоновском поле другой частицы с зарядом Q , положение которой зафиксировано.

2. Построение математической модели. Движение заряженной частицы в кулоновском поле другой частицы описывается с помощью закона Ньютона и закона Кулона:

$$F = \frac{1}{4\pi\epsilon_0} \frac{Qq}{r^2} . \quad (29)$$

В векторной форме

$$\vec{F} = -\frac{1}{4\pi\epsilon_0} \frac{Qq}{r^3} \vec{r} . \quad (30)$$

Знак «минус» возник потому, что сила стремится уменьшить расстояние между телами.

В системе координат, начало которой привязано к телу с большой массой и зарядом Q , уравнения модели в системе СГС имеют вид:

$$\begin{aligned} mdv_x/dt &= -Qq x / (x^2+y^2)^{3/2} \\ dx/dt &= v_x, \\ mdv_y/dt &= -Qq y / (x^2+y^2)^{3/2} \\ dy/dt &= v_y. \end{aligned} \quad (31)$$

Начальные условия определяются двумя параметрами: начальной скоростью и углом α .

$$\begin{aligned} x(0) &= x_0, \quad y(0) = 0, \\ v_{x0} &= v_0 \cos \alpha, \quad v_{y0} = v_0 \sin \alpha. \end{aligned} \quad (32)$$

3. Задания к лабораторной работе.

1. Построить дискретную модель уравнений и разностную схему. Написать алгоритм решения дифференциальных уравнений (31).

2. Разобраться в нижеприведенной программе. Выходные данные представить в графической форме. Рекомендуется использовать графический пакет ORIGIN.

3. Задания для вычислительного эксперимента. В эксперименте Резерфорда α -частица движется с некоторой скоростью на атом золота, в результате кулоновского взаимодействия происходит изменение траектории α -частицы. Построить траектории движения α -частицы тела в зависимости от прицельного параметра $p=y$. Заряд α -частицы: $q = 2e$, масса 4 а.е.м., заряд ядра золота $Q = 79e$.

Начальное расстояние между зарядами 10^{-14} см, кинетическая энергия α -частицы равна $E_k = \frac{m_\alpha v^2}{2} = qU = 4\text{Мэв}$.

4. Программа численного решения.

```

program SG;
const N=1200;
var H,A,V,ALFA:real;
    i:integer;
    X,y,vx,vy:array [0..N] of real;
    RR:TEXT;
begin
    assign (RR,'C:\M NB4.DAT');rewrite(RR);
A:=60; V:=1.4;
ALFA: =3.14*A/180;
h:=0.001;
x[0]:=1.0;y[0]:=0.0;VX[0]:=V*COS(ALFA);
VY[0]:=V* SIN(ALFA);
    for i:=0 to N-1 do
        begin
            vx[i+1]:=vx[i]-h*ABS(x[i])/((x[i]*x[i]+y[i]*y[i])*
                (sqrt(x[i]*x[i]+y[i]*y[i])));
            vy[i+1]:=vy[i]-h*y[i]/((y[i]*y[i]+x[i]*x[i])*
                (sqrt(x[i]*x[i]+y[i]*y[i])));
            x[i+1]:=x[i]+vx[i]*h;
            y[i+1]:=y[i]+vy[i]*h;
        end;
        for i:=0 to n do begin
            writeln (RR,x[i],',',y[i])
        end; Close (RR);
    end.

```

5. Результаты моделирования. На рис. 46, 47 показаны траектории движения заряженной частицы в кулоновском поле другой частицы в зависимости от угла α . Угол α – угол между направлением начальной скорости частицы и линией, соединяющей заряды. Прицельный параметр вычисляется по формуле $p = x_0 \sin \alpha$. Из графика видно, что чем больше угол α , тем меньше угол отклонения в кулоновском поле ядра золота.

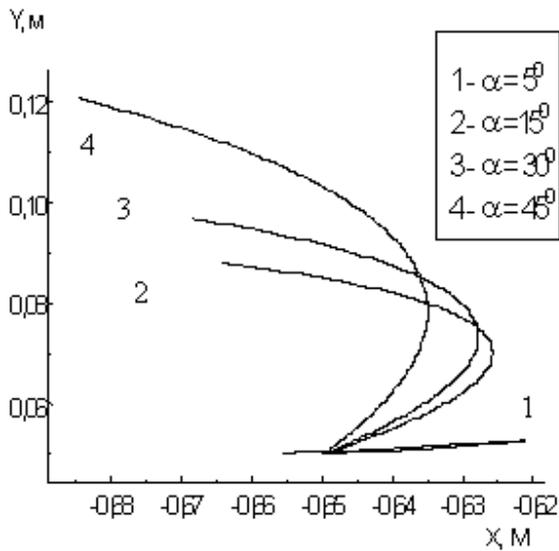


Рис. 46. Траектория движения пробной частицы в кулоновском поле. $Q = 0,07\text{Кл}$, $q = 0,001\text{Кл}$. $R = 1\text{м}$, $v_0 = 0,1\text{ м/с}$

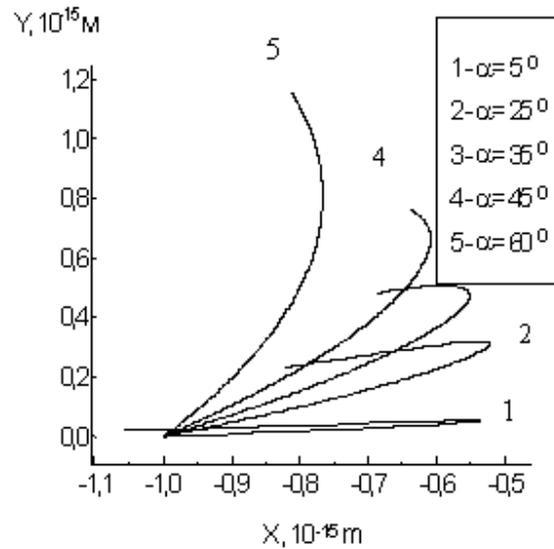


Рис. 47. Траектории движения α -частицы в кулоновском поле ядра золота. $Q = 79e$, $q = 2e$, $R = 10^{-15}\text{ м}$

6.8. ПАРАМЕТРИЧЕСКИЙ МАЯТНИК

1. Постановка задачи. Пусть имеется маятник, подвешенный на растяжимой упругой нити (рис.48). Задача заключается в исследовании колебаний маятника с периодически меняющейся длиной

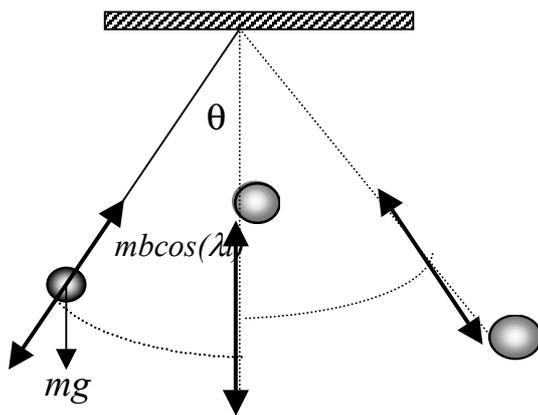


Рис. 48

нити подвеса. Такой маятник называется параметрическим.

2. Математическая модель. Колебания математического маятника описываются следующим дифференциальным уравнением:

$$\frac{d^2\theta}{dt^2} = -\frac{g \sin\theta}{l + l_1 \cos(\lambda t)} \quad (33)$$

Траектория движения маятника может быть описана как

$$x(t) = (l + l_1 \cos(\lambda t)) \sin\theta, \quad (34a)$$

$$y(t) = (l + l_1 \cos(\lambda t)) \cos\theta, \quad (34b)$$

где λ – частота колебаний маятника по длине подвеса. Одно из интересных явлений, связанных с этими колебаниями – это появление так называемого параметрического резонанса при некоторых соотношениях частот λ и ω_0 :

$$\lambda = \omega_0 / 2, \lambda = \omega_0, \lambda = 3\omega_0 / 2 . \quad (35)$$

Параметрический резонанс проявляется в резком нарастании амплитуды колебаний маятника при совпадении значений частоты λ параметрического воздействия со значением указанных частот собственных колебаний маятника.

3. Задания к лабораторной работе.

1. Построить дискретную модель уравнений и разностную схему. Написать алгоритм решения дифференциальных уравнений (33).

2. Разобраться в программе, приведенной ниже. Выходные данные представить в графической форме. Рекомендуются использовать графический пакет ORIGIN.

3. Задания для вычислительного эксперимента.

3.1. Найти траекторию движения маятника.

3.2. Построить график зависимости угла θ от времени.

3.3. Построить (без учета трения) на фазовой плоскости (λ/ω_0 , l_1) границы нескольких зон параметрического резонанса.

4. Программа численного решения.

```

program PM1;
const g=9.81;l=5;n=1500;
var i:integer;
    h,w0,w1,t0,t1,m,a,l1:real;
    f,x:array [0..n] of real;
    y:array [0..n] of real;
    PARM:text;
begin
  h:=0.1; a:=0.5; t0:=1; t1:=20; m:=30;l1:=1;
  x[0]:=1; x[1]:=0; y[0]:=l*cos(a*3.14/180); y[1]:=0;
  f[0]:=0.01; f[1]:=0.01;
  w0:=sqrt(g/l);w1:=2*w0;
  assign (PARM,'d:\maitnik.DAT'); rewrite (PARM);
  for i:=1 to (N-1) do

```

```

begin
  f[i+1]:=2*f[i]-f[i-1]-h*h*g*sin(f[i])/(l+l1*cos(w1*i*h));
  x[i]:=sin(f[i])*(l+l1*cos(w1*i*h));
  y[i]:=cos(f[i])*(l+l1*cos(w1*i*h));
end;
for i:=1 to N do
  begin
    writeln (PARM,i*h,' ',x[i],' ',y[i]);
  end;
close (PARM);
end.

```

5. Результаты моделирования. На рис.49 представлены результаты моделирования колебания параметрического маятника. Как видно из графиков, происходит нарастание колебаний маятника по оси X, аналогичное нарастание происходит в направлении оси Y.

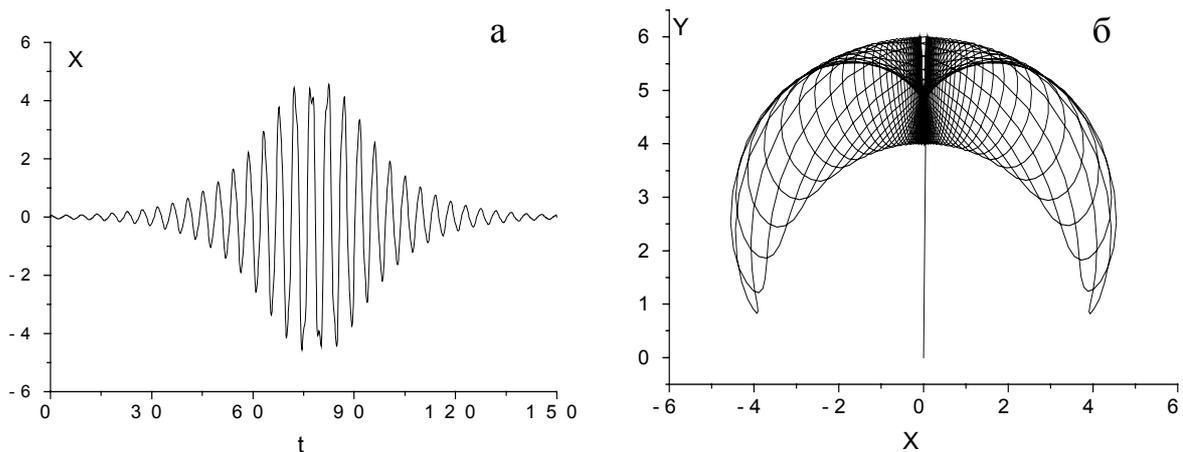


Рис. 49. Зависимость координаты X параметрического маятника от времени (а) и его траектории (б): $\lambda/\omega=2$, $l/l_1=5$, $\theta_0=0$

6.9. ВЫПРЯМЛЕНИЕ С ФИЛЬТРАЦИЕЙ

1. Постановка задачи. Выпрямление – способ получения постоянного тока, например, для питания радиотехнических устройств. Объект выпрямления – электрическое напряжение переменного тока. Цель фильтрации при выпрямлении – ослабить переменную составляющую выпрямленного напряжения.

Здесь необходимо промоделировать работу (рис.50) фильтра выпрямителя. На входе фильтра действует выпрямленное напряжение $E(t)$ одно- или двухполупериодное, полученное из переменного синусоидального напряжения. Для ослабления переменной составляющей в схеме фильтра имеется реактивный элемент: конденсатор емкостью C . Фильтр работает на нагрузку сопротивлением R_H .

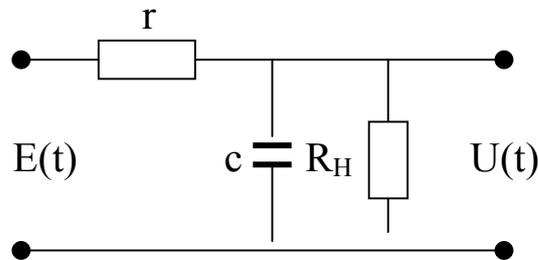


Рис. 50

2. Математическая модель. Вывод дифференциального уравнения, описывающего работу фильтра, производится согласно законам Кирхгофа. В законченном виде уравнение будет:

$$\frac{du(t)}{dt} = \frac{1}{rC} \left[E(t) - \left(1 + \frac{r}{R_H} \right) u(t) \right]. \quad (36)$$

$$u(0) = 0.$$

Входное переменное поле на фильтр можно задать в виде

$$E(t) = \begin{cases} E_0 \cos(\omega t), & \text{если } \cos(\omega t) > 0 \\ 0, & \text{если } \cos(\omega t) < 0 \end{cases}$$

или

$E(t) = E_0 \text{abs}(\cos(\omega t))$, которое означает, что на вход фильтра подается соответственно одно- или двух полупериодное выпрямленное напряжение.

3. Задания к лабораторной работе.

1. Провести масштабирование переменных и привести дифференциальные уравнения к безразмерному виду.

2. Написать алгоритм решения дифференциальных уравнений (36). Построить разностную схему и разностные уравнения.

3. Составить программу согласно алгоритму. Выходные данные представить в графической форме. Рекомендуется использовать графический пакет ORIGIN.

4. Провести вычислительный эксперимент.

4. Результаты моделирования. На рис.51 изображены входное $E(t)$ и выходное $u(t)$ напряжение фильтра для одно- и двухполупериодного выпрямителя. Выбор определенного напряжения $E(t)$ осуществляется путем моделирования входного поля на входе фильтра. С помощью изменения параметров фильтра r , C и нагрузки R_H можно оценить качество фильтрации.

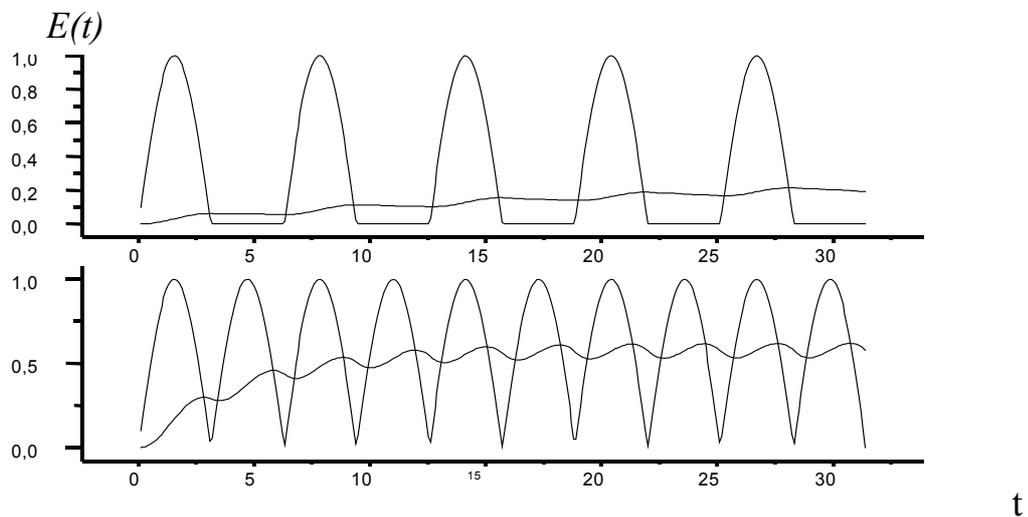


Рис. 51

ГЛАВА 7. ПРИМЕНЕНИЕ ИНФОРМАЦИОННЫХ ТЕХНОЛОГИЙ ПРИ РЕШЕНИИ ФИЗИЧЕСКИХ И МАТЕМАТИЧЕСКИХ ЗАДАЧ

7.1. ИСПОЛЬЗОВАНИЕ ЭЛЕКТРОННЫХ ТАБЛИЦ ДЛЯ ГРАФИЧЕСКОГО ПРЕДСТАВЛЕНИЯ ЧИСЛОВОЙ ИНФОРМАЦИИ

Понимание зависимости между величинами (числовыми данными), сравнение, анализ, применение таких зависимостей – важный компонент знаний школьников. Примеры конкретных зависимостей рассматриваются в различных школьных дисциплинах. Мы ограничимся графическим представлением данных (числовой информации) с помощью диаграмм, гистограмм, линейных графиков. Определимся в терминологии.

Диаграмма – графическое изображение, наглядно показывающее линейными отрезками или геометрическими фигурами соотношение между различными величинами.

Гистограмма – диаграмма, один из видов графического изображения величин по количественному признаку. Гистограммы имеют различные модификации, бывают столбчатые, круговые и т.д. Столбчатая гистограмма представляет собой совокупность смежных прямоугольников, построенных на прямой линии. Площади прямоугольников пропорциональны соответствующим им величинам.

В круговой диаграмме значения величин отображаются в виде секторов круга, углы которых пропорциональны значениям отдельных элементов данных. Секторы штрихуются (закрашиваются) так, чтобы их можно было отличить друг от друга.

В **линейном графике** происходит отображение исходных величин в виде точек, соединенных отрезками прямых линий.

Характер изучения данного вопроса в курсе математики средней школы недостаточно учитывает потребности других дисциплин и практическую деятельность учащихся.

Наличие компьютера и программ «Электронные таблицы» позволяют эффективно решить данный вопрос. Работа с электронными таблицами дает возможность графического представления данных, содержащихся в таблице. Сущность работы с графическими средствами электронной таблицы в том, что пользователь выделяет в таблице данные, которые он хотел бы представить в графическом виде. Обычно эти данные содержатся в смежных столбцах и строках. Один из столбцов или одна из строк выбирается в качестве основы для разметки. Значения других столбцов (строк) используются для построения графиков. Графики могут иметь различный вид: столбики разной высоты (гистограммы), линейные графики, точки на координатной плоскости, круговые диаграммы.

Средства графики электронной таблицы повышают наглядность отображения числовой информации. Теперь по вопросу, где и как использовать те или иные графические представления числовой информации. Круговую диаграмму целесообразно использовать, когда нужно выделить для сравнения долю числовой информации как часть целого. Гистограммы – для количественного сравнения числовой информации. Графики – для выделения динамики числовой информации.

Приведем несколько примеров с использованием компьютера и электронной таблицы.

Пример 1. В таблице показано распределение ПЭВМ в 1988–1990 гг. по основным региональным рынкам в млн штук.

Таблица

Страны	1988	1989	1990
США	9,5	9,8	10,4
Зап. Европа	6,6	7	7,8
Канада	0,62	0,72	0,8
Япония	1,28	1,37	1,5
Австралия	0,49	0,55	0,6
Прочие	2,42	2,72	3,1
Всего	20,91	22,16	24,2

Задание 1. Показать удельный вес распределения ПЭВМ в отдельных ведущих странах в общем мировом выпуске ПЭВМ за 1990 г.

Результат представлен в виде круговой диаграммы на рис. 52.

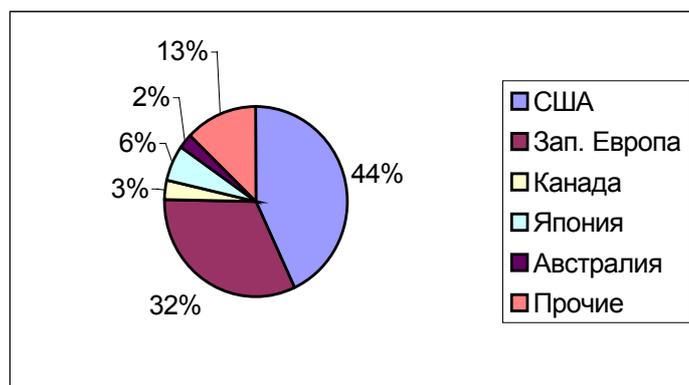


Рис. 52

Задание 2. Показать рост распределения ПЭВМ в США с 1988 по 1990 год. Результат на рис. 53 представлен в виде линейной графики.

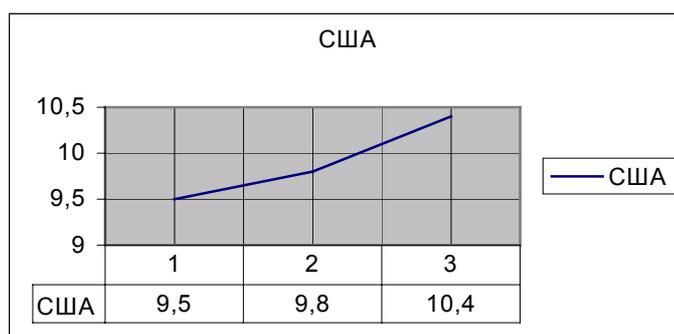


Рис.53

Пример 2. В таблице приведены характеристики четырех крупнейших американских фирм, производящих программное обеспечение для персональных вычислительных средств по итогам 1989 г.

Фирма	Год основ.	Число сотруд.	Рост %	Доходы млрд \$
Microsoft Corp.	1975	4789	33	952,8
Lotus Development Corp.	1982	2806	19	556
Word Perfect Corp.	1979	1612	57	281,4
Ashton Tate	1980	1450	8	265,3
Всего		10657	117	2055,5

Задание 3. Показать количественное соотношение доходов приведенных выше фирм. Результат на рис.54 представлен в виде гистограммы.

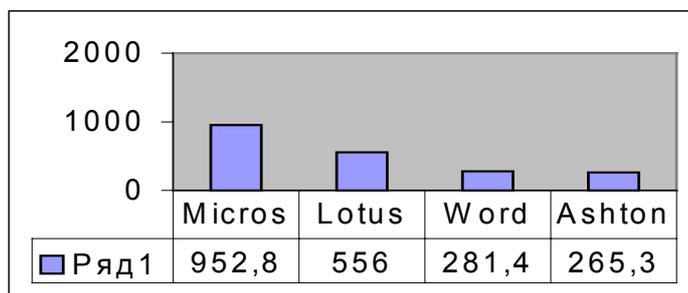


Рис. 54

7.2. ЭЛЕКТРОННАЯ ТАБЛИЦА EXCEL КАК ИНСТРУМЕНТ РЕШЕНИЯ ФИЗИЧЕСКИХ ЗАДАЧ

Электронные таблицы представляют собой класс специальных программ для ведения документации. Документ изображается на экране в виде таблицы, у которой именованы строки и столбцы. Каждая клетка может содержать текст, числа или формулу. С содержимым клеток можно производить арифметические, алгебраические и логические операции. Изменение содержимого одной из клеток автоматически ведет к изменению содержимого других клеток, связанных с ним логически или формулой. Таким образом, обработка данных происходит автоматически, результат получаем в виде готовых таблиц. При необходимости результат можно получить в виде графиков или же диаграмм.

В качестве примера возьмем классическую задачу: моделирование движения небесного тела под действием сил тяготения.

Задача 1. Построить орбиту малой планеты по ее координатам, рассчитанным с интервалом в 5 суток, если в перигелии она находится на расстоянии 0,5 а.е. от Солнца и имеет скорость 0,026 а.е./сут. Считать, что планета движется под действием притяжения только со стороны Солнца. Влияние других планет не учитывать.

Решение. Решение данной задачи средствами программирования приведено в [29]. Применение электронных таблиц Excel дает возможность решить данную задачу без программирования в традиционном понимании.

Ниже приведены общие формулы, позволяющие производить вычисления.

$$T_I = T_{I-1} + DT;$$

$$A_{I,X} = -GM * X_I / R_I^3;$$

$$A_{I,Y} = -GM * Y_I / R_I^3;$$

$$V_{I,X} = V_{I-1,X} - A_{I-1,X} * DT;$$

$$V_{I,Y} = V_{I-1,Y} - A_{I-1,Y} * DT;$$

$$X_I = X_{I-1} + V_{I-1,X} * DT;$$

$$Y_I = Y_{I-1} + V_{I-1,Y} * DT;$$

$$R_I = \text{КОРЕНЬ}(X_I^2 + Y_I^2).$$

Приведенные формулы представляют собой готовую схему алгоритма для вычислений.

Введем свою систему единиц. Время измеряется в сутках, расстояние в а.е. (астрономических единицах), за единицу массы принята масса Солнца. При таком выборе единиц числовые значения исходных данных таковы:

$$X_0 = 0,5; Y_0 = 0; V_{0,X} = 0; V_{0,Y} = 0,026; -GM = -1/58^2; DT = 5.$$

Заполняем электронную таблицу.

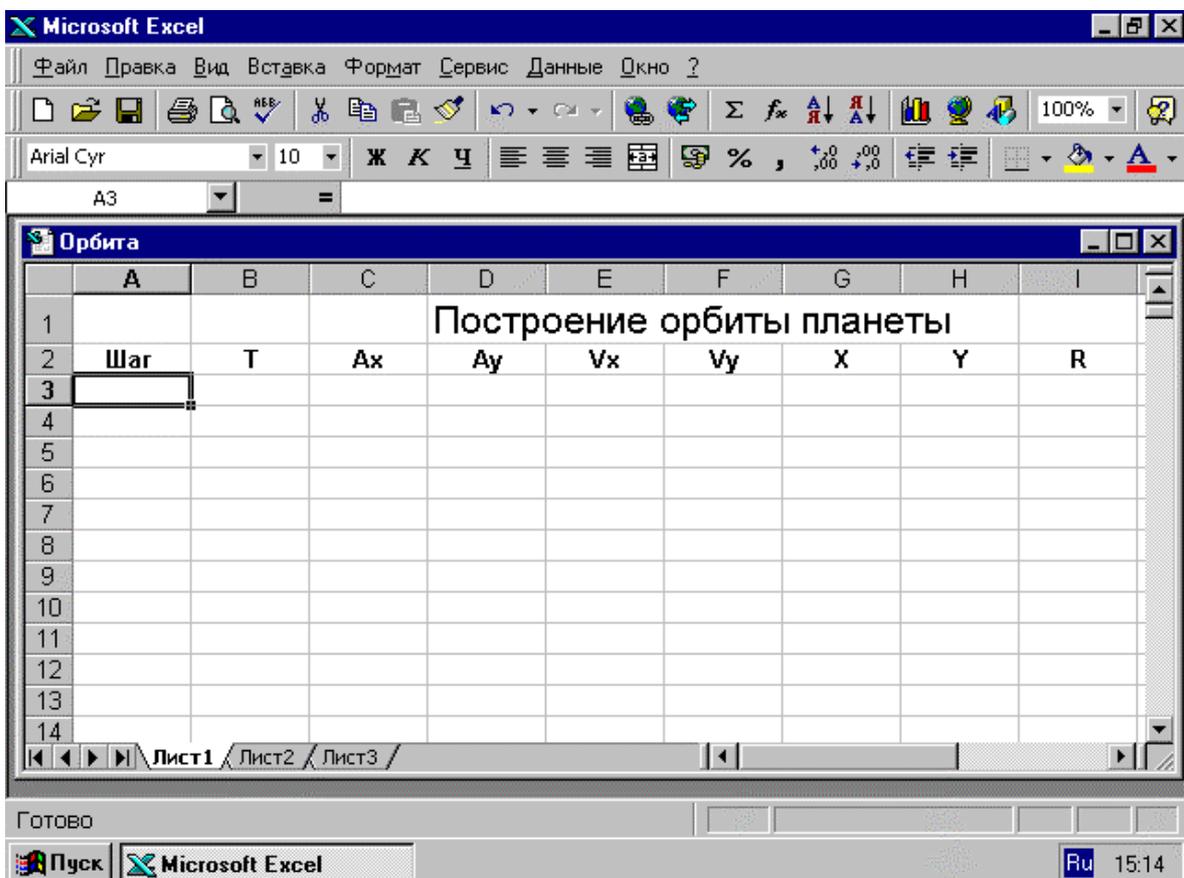


Рис. 55

Первоначальный вид электронной таблицы приведен на рис. 55.

Пояснения к заполнению электронной таблицы

1. В ячейку A3 внесено 1.
2. В ячейку A4 введена формула =A3+1, которая затем скопирована в ячейки A5:A20.
3. В ячейку B3 внесено 0.
4. В ячейку B4 введена формула =B3+5, которая затем скопирована в ячейки B5:B20.
5. В ячейку C3 введена формула $=(-1/(58^2))*G3/(I3^3)$, которая скопирована в ячейки C4:C20.
6. В ячейку D3 введена формула $=(-1/(58^2))*H3/(I3^3)$, которая скопирована в ячейки D4:D20.
7. В ячейку E3 внесено 0.
8. В ячейку E4 введена формула =E3+C3*5, которая затем скопирована в ячейки E5:E20.
9. В ячейку F3 внесено 0,026.
10. В ячейку F4 введена формула =F3+D3*5, которая скопирована в ячейки F5:F20.
11. В ячейку G3 внесено 0,5.
12. В ячейку G4 введена формула =G3+E4*5, которая затем скопирована в ячейки G5:G20.
13. В ячейку H3 внесено 0.
14. В ячейку H4 введена формула =H3+F4*5, которая скопирована в ячейки H5:H20.
15. В ячейку I3 введена формула =КОРЕНЬ(G3^2+H3^2), которая скопирована в ячейки I4:I20.
16. В ячейку J3 введена формула =КОРЕНЬ(C3^2+D3^2), которая скопирована в ячейки J4:J20.

Результаты вычислений приведены в таблице 1.

Таблица 1

Шаг	T	Ax	Ay	Vx	Vy	X	Y	R	A
1	2	3	4	5	6	7	8	9	10
1	0	-0,00119	0	0	0,026	0,5	0	0,5	0,001189
2	5	-0,0012	-0,00033	-0,00595	0,026	0,470273	0,13	0,487911	0,001249
3	10	-0,00109	-0,00067	-0,01196	0,024336	0,410458	0,251682	0,481476	0,001282
4	15	-0,00086	-0,00095	-0,01743	0,020985	0,323313	0,356607	0,481352	0,001283
5	20	-0,00055	-0,00112	-0,02174	0,016232	0,214624	0,437769	0,48755	0,001251

1	2	3	4	5	6	7	8	9	10
6	25	-0,00022	-0,00117	-0,02449	0,010618	0,092172	0,49086	0,499439	0,001192
7	30	7,75E-05	-0,00111	-0,02559	0,004762	-0,03578	0,514669	0,515911	0,001117
8	35	0,000313	-0,00099	-0,0252	-0,00081	-0,16179	0,510624	0,535643	0,001036
9	40	0,000481	-0,00083	-0,02364	-0,00575	-0,27998	0,481887	0,557319	0,000957
10	45	0,000589	-0,00066	-0,02123	-0,00989	-0,38615	0,432462	0,579772	0,000884
11	50	0,000651	-0,0005	-0,01829	-0,01318	-0,4776	0,366546	0,602041	0,00082
12	55	0,000678	-0,00035	-0,01504	-0,01568	-0,55277	0,288146	0,623368	0,000765
13	60	0,000683	-0,00022	-0,01164	-0,01745	-0,61099	0,200905	0,643177	0,000719
14	65	0,000671	-0,00011	-0,00823	-0,01857	-0,65215	0,108054	0,66104	0,00068
15	70	0,000649	-1,2E-05	-0,00488	-0,01913	-0,67652	0,012422	0,676639	0,000649
16	75	0,00062	7,56E-05	-0,00163	-0,01919	-0,68467	-0,08351	0,689745	0,000625
17	80	0,000587	0,000154	0,001472	-0,01881	-0,67731	-0,17755	0,700196	0,000606
18	85	0,000549	0,000224	0,004404	-0,01804	-0,65529	-0,26774	0,707877	0,000593

Для построения орбиты планеты выделяем ячейки G2:H20. Из мастера диаграмм выбираем тип диаграммы «Точечная», далее «Точечная диаграмма со значениями, соединенными сглаживающими линиями без маркеров». В заголовке указываем название графика «Траектория орбиты». Полученная траектория орбиты приведена на рис.56.

Чтобы получить график удаления планеты в различные моменты итерации, выделяем ячейки I2:I20. Из мастера диаграмм выбираем тип диаграммы «Лепестковая», далее «Лепестковая диаграмма является аналогом графика в полярной системе координат ...». В заголовке указываем название диаграммы «Удаление планеты от Солнца». Полученный график приведен на рис. 57.

Для построения графика зависимости ускорения от времени выделяем ячейки C2:D20. Из мастера диаграмм выбираем тип диаграмм «График», далее «График отображает развитие процесса во времени или по категориям». Выбираем «далее», выпадает меню «диапазон данных». В меню «диапазон данных» выбираем «ряд». В «подписи оси X» набираем =Лист1!\$B\$2:\$B\$20. Указатель мыши подводим к «добавить» и щелкаем левой кнопкой мышки, в «имя» набираем А, в «значения» набираем =Лист1!\$I\$2:\$I\$20. Указатель мыши подводим к «далее» и щелкаем левой кнопкой мышки. В подписи данных вводим название диаграммы «График зависимости ускорения от времени».

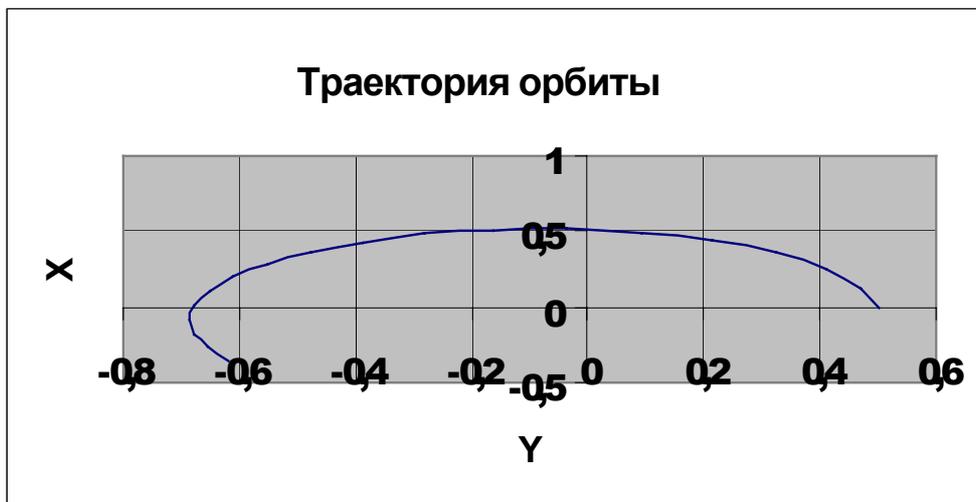


Рис. 56



Рис. 57



Рис. 58

Указатель мышки подводим к «готово», щелкаем левой кнопкой мышки. График готов (рис. 58). Увеличение количества шагов итерации дает более полное решение задачи.

Использование электронных таблиц при моделировании физических процессов не единственная область. Применение электронных таблиц при проведении расчетов данных лабораторных работ позволяет сделать основной упор на постановку опыта (эксперимента) и анализ полученных результатов, т.е. формирование навыков, необходимых физику-экспериментатору.

Задача 2. Определить длину световой волны. Лабораторная работа за 11 класс «Определение длины световой волны» (Физика: Учеб. за 11 кл. сред. шк./ Н.М. Шахмаев, Д.Ш. Шодиев. М.: Просвещение, 1991.)

Решение. Для определения длины световой волны с помощью дифракционной решетки используется следующая формула:

$$\lambda = d \cdot h / l,$$

где d – постоянная дифракционной решетки,

l – расстояние между дифракционной решеткой и экраном,

h – расстояние между центром дифракционной картины и наблюдаемым спектром,

λ – длина световой волны.

Учащиеся, предварительно ознакомившись с условиями проведения лабораторной работы и собрав указанную в описании установку, зная значение d , измерив l , поочередно определяя положения цветных полос в спектрах первого порядка, данные заносят в электронную таблицу. Пример подготовки электронной таблицы приведен на рис. 59.

Пояснения к заполнению электронной таблицы

1. Числовые данные вводим в соответствующие ячейки.

2. В ячейку D4 введена формула $= (B4 + C4) / 2$, которая затем скопирована в ячейки D5:D10.

3. В ячейку G4 введена формула $= F4 * D4 / E4$, которая затем скопирована в ячейки G5:G10.

Результаты выполненной лабораторной работы для наглядной количественной оценки можно представить в виде графиков или гистограмм.

В качестве примера приведем разработку доцента Бийского Госпедуниверситета, кандидата физико-математических наук М. Старовикова «Движение тела в поле силы тяжести Земли».

1	Определение длины световой волны						
3	Цвет полосы	h слева (см)	h справа (см)	h среднее (см)	l (см)	d (см)	λ (см)
4	Фиолетовая	2	2	2	50	0,001	0,00004
5	Синяя	2,2	2,2	2,2	50	0,001	0,000044
6	Голубая	2,4	2,4	2,4	50	0,001	0,000048
7	Зеленая	2,6	2,7	2,65	50	0,001	0,000053
8	Желтая	2,8	2,9	2,85	50	0,001	0,000057
9	Оранжевая	3,3	3,2	3,25	50	0,001	0,000065
10	Красная	3,5	3,5	3,5	50	0,001	0,00007

Рис. 59

Задача 3. Свинцовый шарик массой $m = 0,54$ г падает в масло (плотность свинца $\rho_c = 11,3 \cdot 10^3$ кг/м³, плотность масла $\rho_m = 0,92 \cdot 10^3$ кг/м³, вязкость масла $\text{ДинВяз} = 0,95$ кг/(мс)) с начальной скоростью V_0 , направленной по вертикали. Постройте графики зависимости скорости, ускорения и перемещения шарика от времени. Варьируя V_0 , установите с помощью компьютерной модели характерные особенности движения шарика.

Решение. Оформление и ввод данных проведем по аналогии с задачами 1 и 2. Введем обозначения и присвоим им значения и вычислим некоторые вспомогательные величины. Заполним нулевой и первый строку таблицы данных. Получив первые значения скорости и ускорения, остальные значения получаем, растягивая мышью выделенную строку первых значений данных по вертикали.

Обозначение величин и присваивание им значений

$P_c =$	1,13E+04	кг/м ³	ДинВязк=	0,95	кг/(мс)	$m =$	5,40E-04 кг
$P_m =$	9,50E+02	кг/м ³	$V_0 =$	-0,5	м/с		
$g =$	9,81	м/с ²	$dt =$	0,001	с		

Вычисление вспомогательных величин

$r =$	Рис. 60. Графики зависимости ускорения и скорости от времени					
$k =$	0,040311934	кг/с				

Таблица данных

N	t, с	V, м/с	dV, м/с	y, м	dy, м/с	a, м/с ²
0	0	-0,5	0,047052	0	-0,0005	4,63E+01
1	0,001	-0,45295	0,043539	-0,0005	-0,00045	4,28E+01
2	0,002	-0,40941	0,040289	-0,000952948	-0,00041	3,95E+01
3	0,003	-0,36912	0,037281	-0,001362357	-0,00037	3,65E+01
4	0,004	-0,33184	0,034498	-0,001731477	-0,00033	3,38E+01
5	0,005	-0,29734	0,031923	-0,002063316	-0,0003	3,12E+01
6	0,006	-0,26542	0,02954	-0,002360656	-0,00027	2,88E+01
7	0,007	-0,23588	0,027335	-0,002626073	-0,00024	2,66E+01
8	0,008	-0,20854	0,025294	-0,002861951	-0,00021	2,46E+01
9	0,009	-0,18325	0,023406	-0,003070494	-0,00018	2,27E+01
10	0,01	-0,15984	0,021658	-0,003253743	-0,00016	2,09E+01
11	0,011	-0,13818	0,020042	-0,003413586	-0,00014	1,93E+01
-	-	-	-	-	-	-
45	0,045	0,111085	0,001433	-0,002323035	0,000111	6,93E-01
46	0,046	0,112519	0,001326	-0,002211949	0,000113	5,86E-01
47	0,047	0,113845	0,001227	-0,002099431	0,000114	4,87E-01
48	0,048	0,115072	0,001136	-0,001985586	0,000115	3,95E-01
49	0,049	0,116208	0,001051	-0,001870514	0,000116	3,10E-01
50	0,05	0,117258	0,000972	-0,001754306	0,000117	2,32E-01

Аналогично можно использовать электронные таблицы при выполнении и других работ по математическому моделированию.

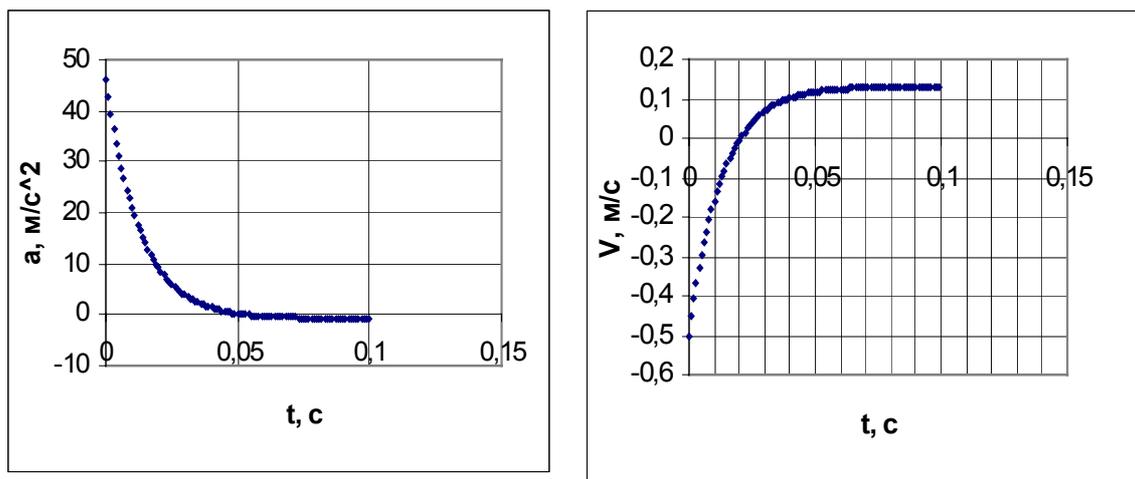


Рис. 60. Графики зависимости ускорения и скорости от времени

7.3. РЕШЕНИЕ МАТЕМАТИЧЕСКИХ И ФИЗИЧЕСКИХ ЗАДАЧ В СИСТЕМЕ MATHCAD 8.1

На персональном компьютере сегодня можно решать задачи научно-технических расчетов, не прибегая к их кодированию на каком бы ни было алгоритмическом языке (Бейсик, Паскаль, СИ). Использование интегрированных программных систем автоматизации математических расчетов (Eureka, MatLab, Maple, Mathematica, MathCAD и др.) позволяет решать поставленные задачи на входном языке, который максимально приближен к естественному математическому языку.

В MathCAD описание решения математических задач дается с помощью привычных математических формул и знаков. Такой же вид имеют и результаты вычислений. Так что системы MathCAD вполне оправдывают аббревиатуру CAD (Computer Aided Design), говорящую о принадлежности к сложным и продвинутым системам автоматизированного проектирования – САПР. MathCAD своего рода САПР в математике.

К задачам, решаемым в системах MathCAD, можно отнести:

- подготовку научно-технических документов, содержащих текст и формулы, записанные в привычной для специалистов форме;

- вычисление результатов математических операций, в которых участвуют числовые константы, переменные и размерные физические величины;

- операции с векторами и матрицами;

решение уравнений и систем уравнений (неравенств);
статистические расчеты и анализ данных;
построение двумерных и трехмерных графиков;
тождественные преобразования (в том числе упрощение),
аналитическое решение уравнений и систем;
дифференцирование и интегрирование, аналитическое и численное;
решение дифференциальных уравнений;
проведение серий расчетов с различными значениями начальных условий и других параметров.

Подробно изучить приемы работы с системой MathCAD можно по специальным руководствам. Мы же остановимся на применении системы при решении некоторых школьных математических задач. Надо отметить, что технология работы проста и очень нравится ребятам. «Общение» с MathCAD повышает интерес школьников к математике, информатике. Учащиеся получают начальные профессиональные знания и привыкают к профессиональному языку. Надеемся, что приведенные примеры решения типовых математических задач заинтересуют учителей информатики, математики, физики в использовании системы MathCAD.

Знакомство с системой MathCAD 8.1

Пользовательский интерфейс системы создан так, что пользователь, имеющий элементарные навыки работы с Windows-приложениями, может сразу начать работать с MathCAD.

Под интерфейсом понимается не только легкое управление системой как с клавишного пульта, так и с помощью мыши, но и просто набор необходимых символов, формул, текстовых комментариев с последующим запуском документов (Worksheets) в реальном времени.

Запустив систему MathCAD из Windows, вы увидите на экране диалоговое окно, первоначально пустое (рис. 61). Над ним видна строка с основными элементами интерфейса. Опции главного меню, содержащиеся в этой строке, легко изучить самостоятельно; некоторые из них очень похожи на стандартные опции, принятые в текстовых редакторах Windows.

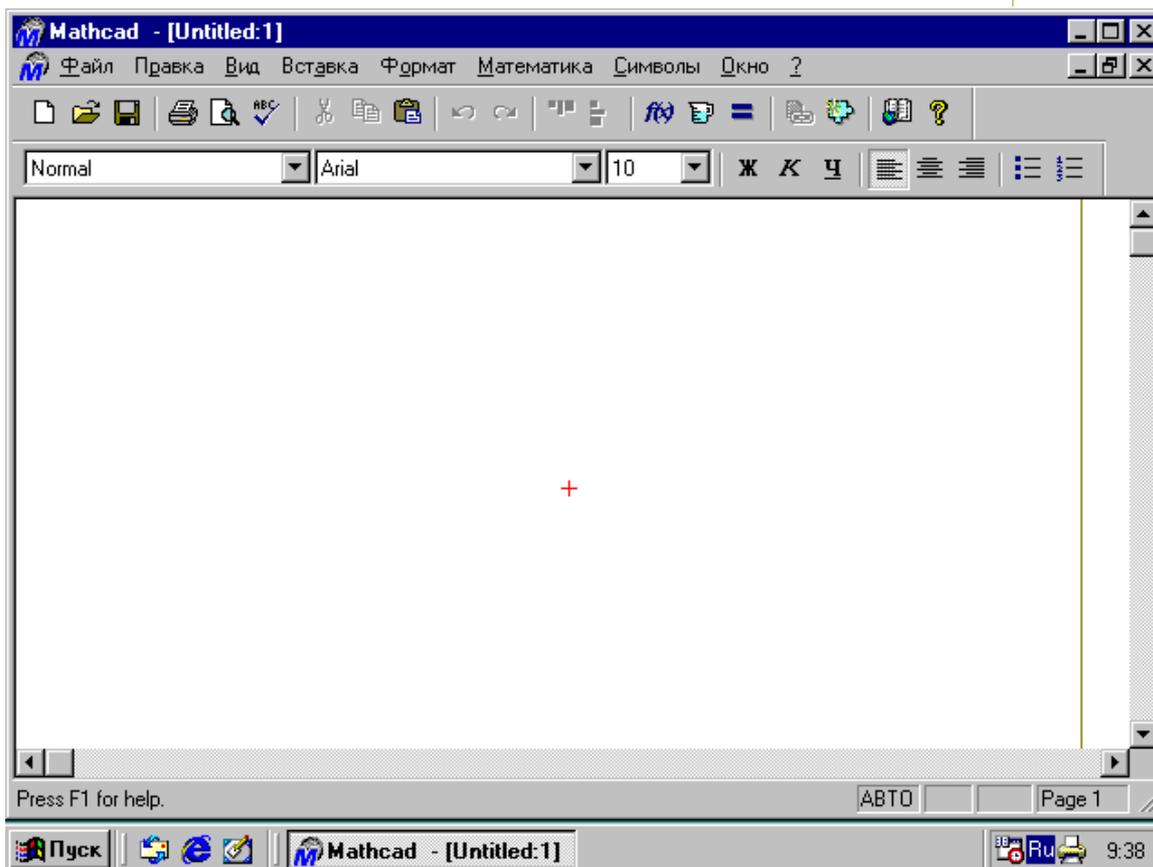


Рис. 61

Работа с документами MathCAD обычно не требует обязательного использования возможностей главного меню, так как основные из них дублируются кнопками быстрого управления, которые расположены в удобных, перемещаемых с помощью мыши наборных панелях – палитрах. Наборные панели появляются в окне редактирования документов при активизации кнопок – пиктограмм. Они служат для вывода заготовок – шаблонов математических знаков (цифр, знаков арифметических операций, матриц, знаков интеграла, производных, приделов и др.). Указатель мыши подводим к «Вид» в главном меню, щелкаем левой кнопкой мыши; указатель подводим к «Панели инструментов» и щелкаем левой кнопкой мыши; выпадает следующее меню. Указатель мыши подводим к «Математика» и щелкаем левой кнопкой мыши. Выпадают наборные панели (рис. 62).

Приведем примеры решения некоторых типовых математических задач.

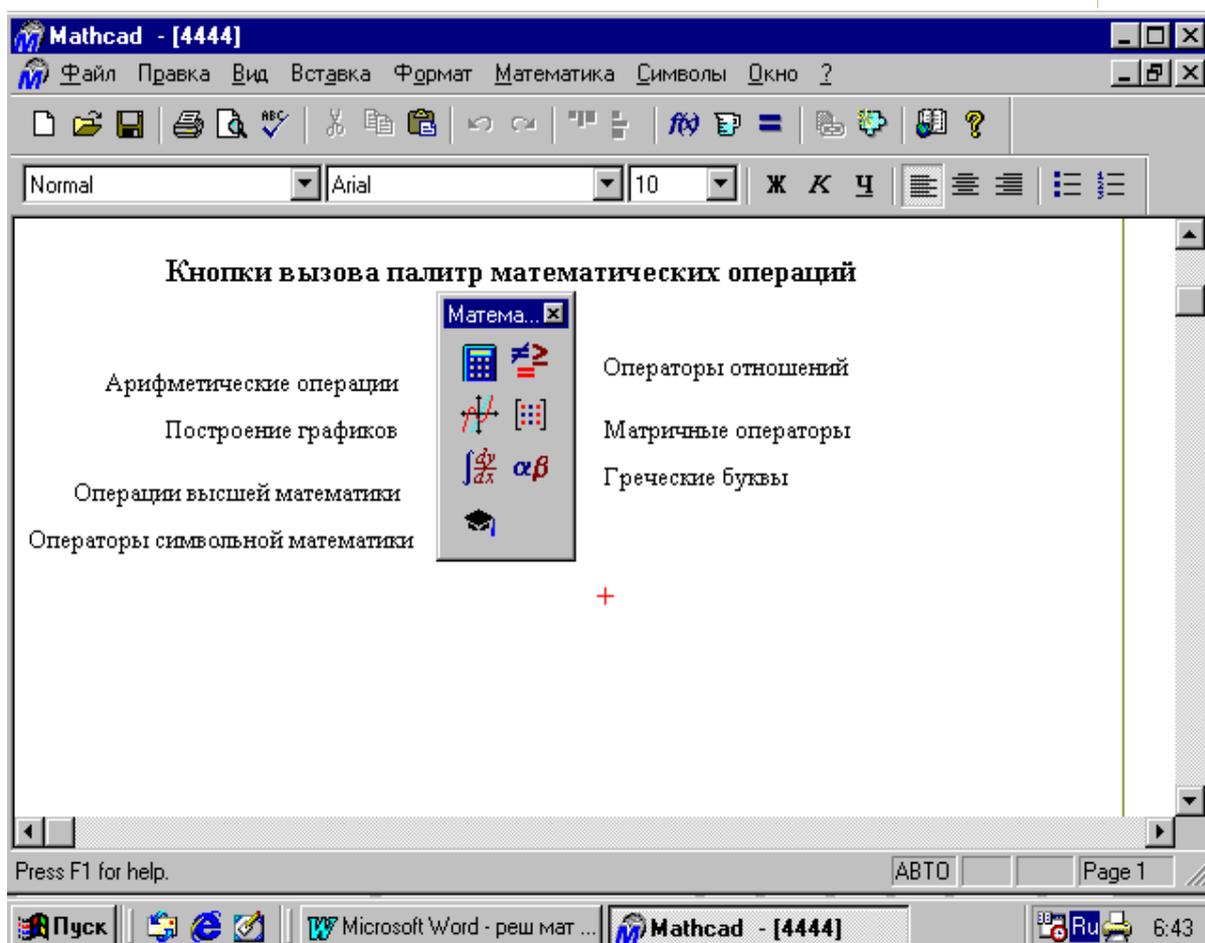


Рис. 62

Примечание. Решение завершаем щелчком левой кнопки мыши, предварительно уводя указатель мыши за пределы выделенной области набора примера.

Пример 1. Упростить выражение: $\frac{a^2 - b^2}{2a + 2b}$.

Решение. В окне редактирования (далее на экране) набираем исходное выражение $\frac{a^2 - b^2}{2a + 2b}$.

Указатель мыши подводим к опции «Символы» в главном меню и щелкаем левой кнопкой мыши один раз (далее входим в «Символы»). В выпавшем меню указатель мыши подводим к опции «Упростить» и активизируем (щелчком левой кнопкой мыши) указанную опцию. На экране видим отображение нашего выражения, но уже в выделенном виде. Повторяем наши действия: входим в «Символы» (подводим указатель мыши и щелкаем левой

кнопкой мыши) и активизируем «Упростить». На экране появляется ответ: $\frac{1}{2} a - \frac{1}{2} b$.

Пример 2. Вычислить: $10x^2 - 5y^2$, при $x=1,5$ и $y=-1,6$.

Решение. На экране с клавиатуры набираем знак =, компьютер сам поставит знак =.

$$x: =1.5 \quad y: =-1.6$$
$$10 \cdot x^2 - 5 \cdot y^2 =$$

рядом со знаком равенства читаем ответ: 9.7.

Пример 3. Преобразуйте в многочлен: $(a + 2 \cdot b) \cdot (a - 2 \cdot b) \cdot (a^2 + 4 \cdot b^2)$.

Решение. На экране набираем исходное выражение $(a + 2 \cdot b) \cdot (a - 2 \cdot b) \cdot (a^2 + 4 \cdot b^2)$.

Входим в меню «Символы», активизируем «Расширить». На экране отображается введенное нами выражение. Повторяем сделанные нами операции: входим в «Символы», активизируем «Расширить». На экране читаем ответ: $a^4 - 16 \cdot b^4$.

Пример 4. Разложите на множители: $4z^4 - 25k^2$.

Решение. На экране набираем $4 \cdot z^4 - 25 \cdot k^2$.

Входим в меню «Символы», активизируем «Фактор». На экране отображается введенное нами выражение. Повторяем сделанные нами операции: входим в «Символы», активизируем «Фактор». На экране читаем ответ: $-(5 \cdot k - 2 \cdot z^2) \cdot (5 \cdot k^2 + 2 \cdot z^2)$.

Пример 5. Разложите на множители: $12x^3 - 3x^2y - 18xy^2$.

Решение. На экране набираем $12 \cdot x^3 - 3 \cdot x^2 \cdot y - 18 \cdot x \cdot y^2$.

Входим в меню «Символы», активизируем «Фактор». На экране отображается введенное нами выражение. Повторяем сделанные нами операции: входим в «Символы», активизируем «Фактор». На экране читаем ответ: $3 \cdot (4 \cdot x^2 - x \cdot y - 6 \cdot y^2)$.

Пример 6. Сократите дробь: $\frac{x^2 - 2mx + 3x - 6m}{x^2 + 2mx + 3x + 6m}$

Решение. На экране набираем исходное выражение.

Входим в меню «Символы», активизируем «Упростить». На экране отображается введенное нами выражение. Повторяем сделанные нами операции: входим в «Символы», активизируем

«Упростить». На экране читаем ответ: $\frac{x - 2m}{x + 2m}$.

Пример 7. Вычислите: $\frac{36^{\frac{1}{2}}}{27^{1/3} - 81^{1/4} \cdot 5}$.

Решение. На экране набираем искомый пример. Ставим знак равенства и читаем ответ: -0.014.

Пример 8. Решите уравнение: $2 \cdot (5 \cdot x - 1)^2 + 35 \cdot x - 11 = 0$.

Решение. Аналитическое решение. Набираем ключевое слово **given** (дано).

Вводим уравнение $2 \cdot (5 \cdot x - 1)^2 + 35 \cdot x - 11 = 0$. Здесь при вводе знака =, мы вводим знак логическое равно из палитры, а не с клавиатуры.

Набираем **find(x)→**, рядом читаем решение:

$$\begin{array}{r} -\underline{3} \quad \underline{3} \\ 5 \quad 10 \end{array}$$

Пример 9. Решите уравнение: $y^3 + 6 \cdot y^2 - 16 \cdot y = 0$.

Решение. Численный поиск корней уравнения.

Для поиска корней искомой переменной надо присвоить начальное значение, а затем при помощи вызова функции **root(f(x),x)** находим корень.

Набираем на экране $y:=1$
 $\text{root}(y^3 + 6 \cdot y^2 - 16 \cdot y, y)=$
читаем ответ: 8.

Если в качестве начального значения возьмем $y:=-2$, то получим ответ: 0.

Пример 10. Решите систему уравнений:

$$x^2 + y + 8 = x \cdot y$$

$$y - 2 \cdot x = 0.$$

Решение. Набираем ключевое слово **given** и систему уравнений

$$x^2 + y + 8 = x \cdot y$$

$$y - 2 \cdot x = 0.$$

Между левыми и правыми частями уравнений ставим знак логическое равно =. Набираем вызов функции **find(x,y)→**, читаем на экране ответ:

$$\begin{array}{r} -2 \quad 4 \\ -4 \quad 8 \end{array}$$

Пример 11. а) Решите неравенство: $5 \cdot x - 3 \leq 4$.

Решение. На экране набираем неравенство и входим в палитру «Символические операторы», активизируем «**solve**», набираем $5 \cdot x - 3 \leq 4$ solve, $x \rightarrow$ на экране читаем ответ: $x \leq 7/5$.

б) Решите неравенство: $2 \cdot a^2 - 5 < 15$.

На экране набираем неравенство и входим в палитру «Символические операторы», активизируем «**solve**», набираем а

$2 \cdot a^2 - 5 < 15$ solve, а \rightarrow

на экране читаем ответ: $(-\sqrt{10} < a) \cdot (a < \sqrt{10})$.

Пример 12. Вычислите: $\cos 34^\circ \cdot \cos 56^\circ - \sin 34^\circ \cdot \sin 124^\circ$.

Решение. Набираем на экране

$\cos(34 \cdot \text{deg}) \cdot \cos(56 \cdot \text{deg}) - \sin(34 \cdot \text{deg}) \cdot \sin(124 \cdot \text{deg}) =$

и читаем ответ: 0.

Примечание. Набираем **deg**, если угол задан в градусах; **rad** – в радианах.

Пример 13. Построить график функции $y = 2 \cdot \sin 2 \cdot x$.

Решение. Набираем на экране $y(x) := 2 \cdot \sin(2 \cdot x)$. Отводим указатель мыши от выделенной части и щелкаем левой кнопкой мышки. Указатель мыши подводим к «Построение графиков» и входим, активизируем «Декартов график». Появляется шаблон для построения графика. На ней выделены метки. Указатель мыши подводим к нижней метке, активизируем. Набираем x . Появляются по горизонтали еще две метки, где мы должны указать интервалы построения графика. Указатель мыши подводим к левой метке, щелкая левой кнопкой мыши активизируем и вводим левую границу **0**. Указатель мыши подводим к правой границе, активизируем и вводим **5**. Уводим указатель мыши к метке оси **Y**, активизируем его и вводим $y(x)$. Появляются метки нижней и верхней границ оси **Y**. В нижней набираем **-2**, в верхней **2**. Отводим указатель мыши от шаблона для графиков, щелкаем левой кнопкой мыши. Появляется искомый график. Для форматирования графика нужно дважды щелкнуть в области графика. В выпавшем меню можно управлять отображением линий, масштабом и др.

Пример 14. Построить график функции $y = 2 \cdot x + 2$.

Решение. Решение аналогично предыдущему примеру.

Пример 15. Вычислите предел многочлена: $2 \cdot x^3 - 3 \cdot x^2 + 3$.

Решение. Из палитры «Высшей математики», активизируем **lim**, заполняем выведенный шаблон; завершаем набор знаком \rightarrow палитры «Операторы отношений». На экране читаем ответ: 7.

Пример 16. Вычислите производную: $\cos x + x \cdot \sin x$.

Решение. Из палитры «Высшей математики» активизируем $\frac{d}{dx}$, заполняем выведенный шаблон; завершаем набор знаком \rightarrow палитры «Операторы отношений». На экране читаем ответ: $x \cdot \cos(x)$.

Пример 17. Вычислите неопределенный интеграл: $\int (x^2 + \cos x) dx$.

Решение. Из палитры «Высшей математики» активизируем \int , заполняем выведенный шаблон; завершаем набор знаком \rightarrow палитры «Операторы отношений». На экране читаем ответ: $\frac{1}{3} x^3 + \sin(x)$.

Пример 18. Вычислите определенный интеграл: $\int_0^1 \sqrt{x^2 + 1} dx$.

Решение. Из палитры «Высшей математики» активизируем \int_a^b , заполняем выведенный шаблон; завершаем набор знаком \rightarrow палитры «Операторы отношений». На экране читаем ответ: $\frac{1}{2} \sqrt{2} - \frac{1}{2} \ln(\sqrt{2} - 1)$.

Пример 19. Решить задачу движения тела, брошенного под углом к горизонту, и построить его график

Система дифференциальных уравнений, описывающая данный процесс, имеет вид:

$$\ddot{x} = -k\dot{x}, \quad \dot{x}(0) = v_0 \cos \alpha, \quad x(0) = 0,$$

$$\ddot{y} = -ky - g, \quad \dot{y}(0) = v_0 \sin \alpha, \quad y(0) = 0.$$

Решение.

$$v := 0.5 \quad g := 9.8 \quad a := \frac{3.14}{4}$$

$$x_0 := 0 \quad y_0 := 0 \quad h := 0.01$$

$$i := 1 .. 10000$$

$$x_1 := v \cdot \cos(a) \quad y_1 := v \cdot \sin(a)$$

$$x_{i+1} := 2 \cdot x_i - x_{i-1} \quad y_{i+1} := 2 \cdot y_i - y_{i-1} - h^2 \cdot g$$

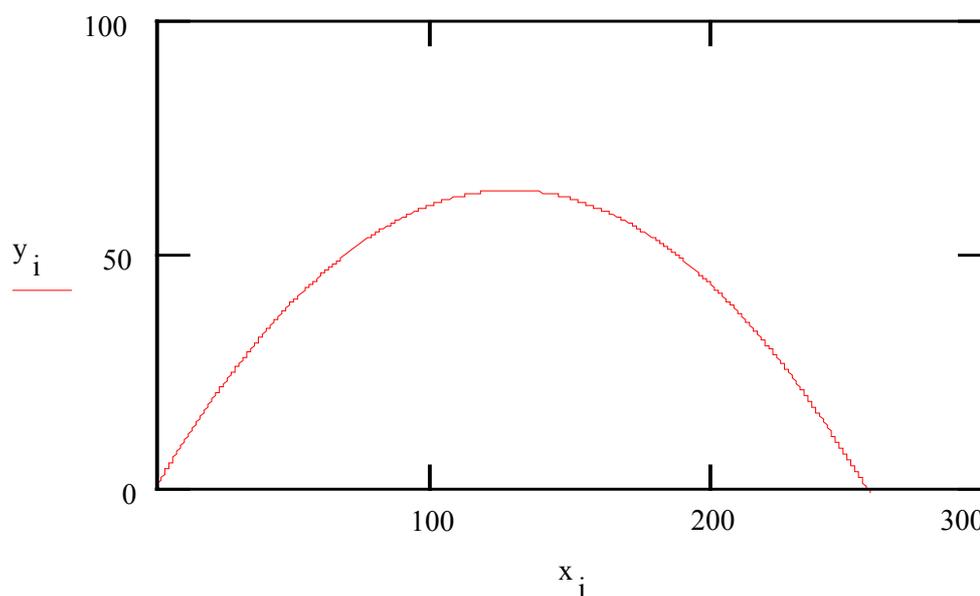


Рис. 63

Приведенные примеры далеко не полностью исчерпывают возможности системы MathCAD, но позволяют получить представление о возможностях системы и начальные навыки работы с ней.

7.4. РАЗРАБОТКА БАЗЫ ДАННЫХ «ПЛАНЕТЫ СОЛНЕЧНОЙ СИСТЕМЫ»

База данных – организованная структура, предназначенная для хранения информации. Сегодня большинство систем управления базами данных (СУБД) позволяют размещать в своих структурах не только данные, но и методы, с помощью которых происходит взаимодействие с пользователями или другими программно-аппаратными комплексами. В мире существует множество систем

управления базами данных, однако большинство из них имеют общие приемы и методы. Поэтому в качестве объекта СУБД выберем Microsoft Access, входящий в пакет Microsoft Office.

В качестве примера приведем разработку базы данных «Планеты Солнечной системы». Разработка любой базы данных начинается с проекта. Базу данных «Планеты Солнечной системы» запланируем из двух таблиц – «Планета» и «Спутники». Ниже приведены структуры указанных выше таблиц.

Структура таблицы «Планета»

Имя поля	Тип данных	Длина
Код	Счетчик	Длинное целое
Планета	Текстовый	10
Среднее расстояние от Солнца в астрономических единицах	Числовой	С запятой
Среднее расстояние от Солнца в миллионах км	Числовой	С запятой
Период обращения вокруг Солнца в земных годах и сутках	Текстовый	10
Средняя скорость движения по орбите в км/с	Числовой	С запятой
Период осевого вращения	Текстовый	15
Наклонение экватора к плоскости орбиты в градусах	Текстовый	5
Экваториальный диаметр в единицах диаметра Земли	Числовой	С запятой
Экваториальный диаметр в тысяч км	Числовой	С запятой
Объем в единицах Земли	Числовой	С запятой
Масса в единицах массы Земли	Числовой	С запятой
Плотность по сравнению с водой	Числовой	С запятой
Температура	Числовой	Целое
Число спутников	Числовой	Целое

Структура таблицы «Спутники»

Имя поля	Тип данных	Длина
Код	Счетчик	Длинное целое
Планета	Текстовый	10
Спутники	Текстовый	15
Кто и когда открыл	Текстовый	20
Расстояние от центра планеты в радиусах планеты	Числовой	С запятой
Расстояние от центра в тысяч км	Числовой	С запятой
Период обращения – звездный в сутках	Числовой	С запятой
Диаметр спутника в км	Числовой	С запятой
Масса спутника – во сколько раз она меньше массы Земли	Числовой	С запятой
Масса спутника – во сколько раз она меньше массы планеты	Числовой	С запятой

В Access под базой данных понимают отдельный файл, содержащий различные объекты Access: таблицы, запросы, макросы, отчеты, формы, модули. Рассмотрим примеры создания некоторых перечисленных объектов.

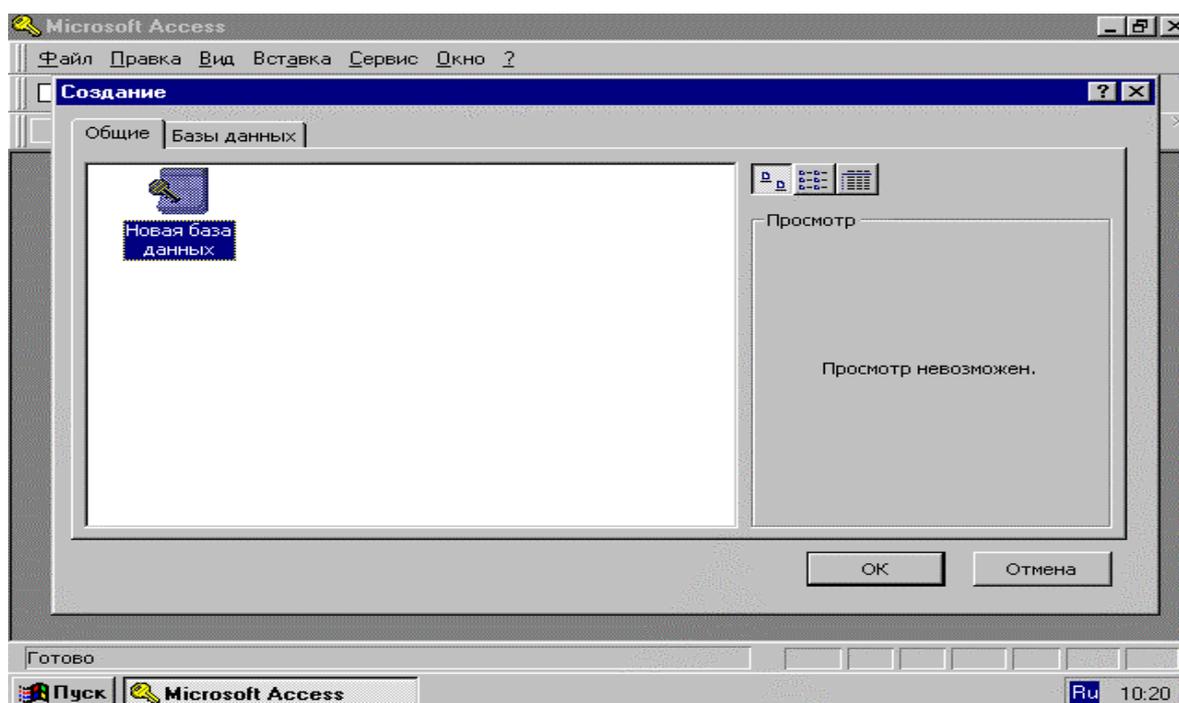


Рис. 64

Загружаем Microsoft Access. Открываем **Файл – Создать базу данных – Новая база данных – Ок** (рис.64).

В меню **Файл новой базы данных** указываем имя файла, например: **db1** и указываем создать, выпадает меню (рис. 65).

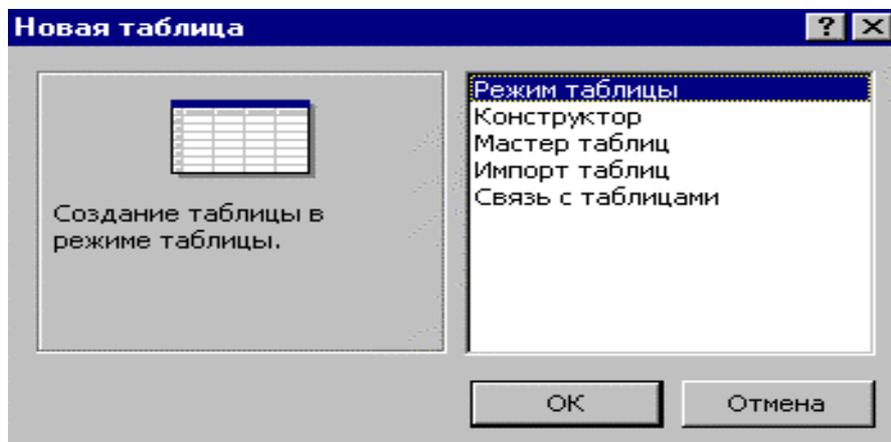


Рис. 65

Указываем **Создать**, выпадает меню (рис.66).

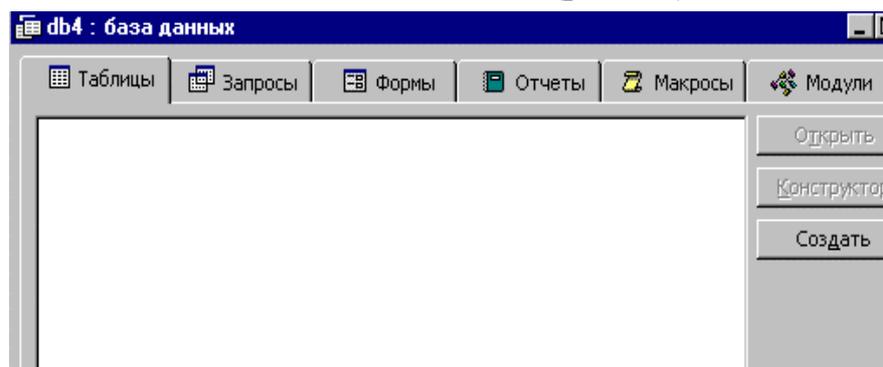


Рис. 66

Выбираем **Конструктор** и **Ок**. Выпадает следующее меню (рис.67).

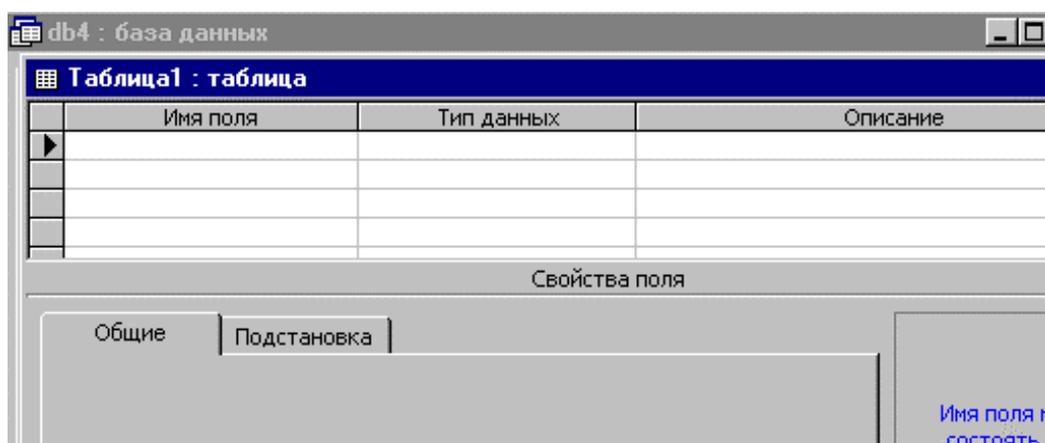


Рис. 67

Заполняем структуру таблицы «Планета». Потом закрываем указанное меню. Открываем **Таблицы** и заполняем данными. Фрагмент таблицы приведен на рис.68.

Планета					
	Код	Планета	Среднее ра	Среднее ра	Период обр
▶	1	Меркурий	0,387	57,9	88 сут.
	2	Венера	0,723	108,1	224,7 сут.
	3	Земля	1	149,6	365,3 сут.
	4	Марс	1,524	227,9	687 сут.
	5	Юпитер	5,2	778,3	11,9 лет
	6	Сатурн	9,55	1429	29,5 лет
	7	Уран	19,2	2875	84 года
	8	Нептун	30,1	4504	164,8 года
	9	Плутон	39,5	5910	247,7 года

Рис. 68

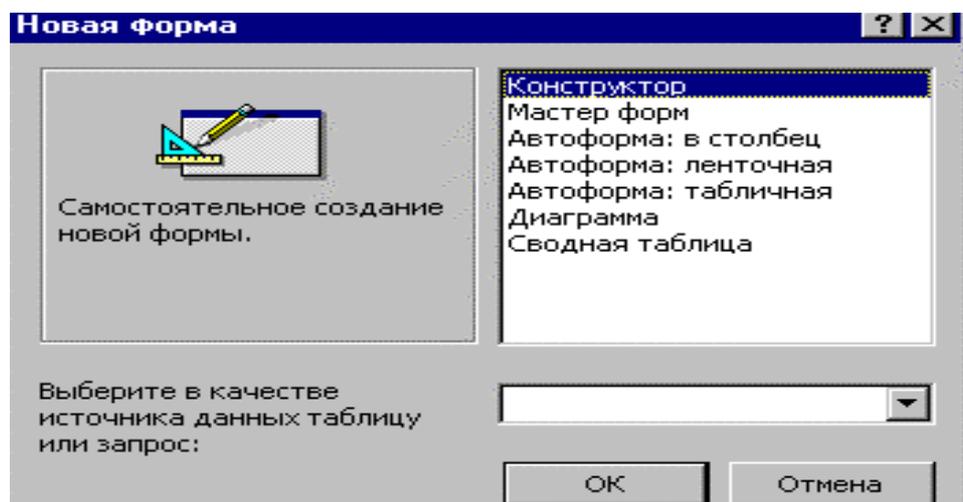


Рис. 69

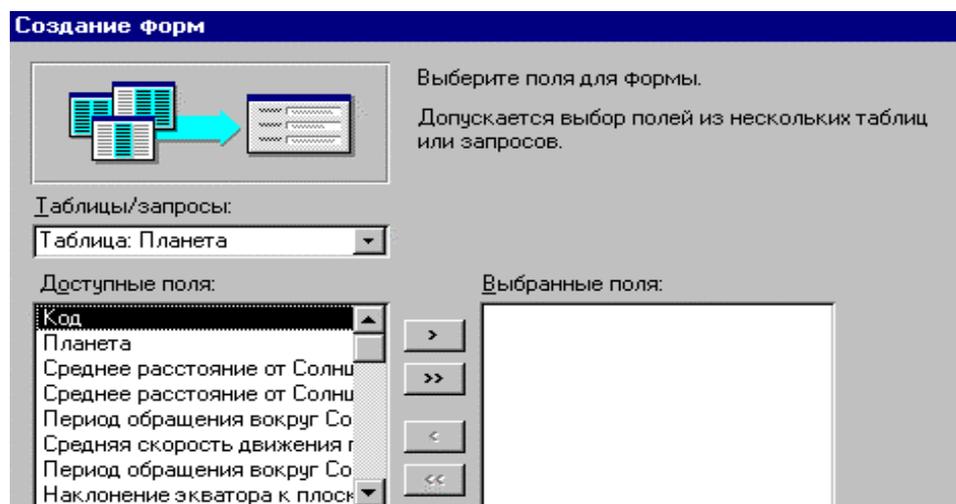


Рис. 70

Работать с таблицей не всегда удобно, а представление в ней данных не очень наглядно. Если нужно задать специальные форматы или автоматизировать работу с данными, лучше использовать формы. В меню (рис.66) активизируем **Формы – Создать**, выпадает меню (рис. 69).

Выбираем **Мастер форм** и **Ок**. Выпадает следующее меню (рис.70)

Нажимаем на >>, если выбираем все указанные поля, или выбираем необходимые поля, нажимая на >. Далее – **Готово**. (Форму можно готовить вручную, для этого нужно в предыдущем меню выбрать **Конструктор**).

Фрагмент готовой базы данных «Планета» приведен на рис.71.

Аналогично создаем таблицу «Спутники». На рис. 72 фрагмент базы данных «Спутники».

Рис. 71



Рис. 72

В Access *связь* – это ссылка между двумя таблицами, которая показывает, как соотносятся данные в этих таблицах. При создании связей в программе Access указывают, какие поля в разных таблицах содержат одинаковые данные. На рис.73 показано установление связей между нашими двумя таблицами.

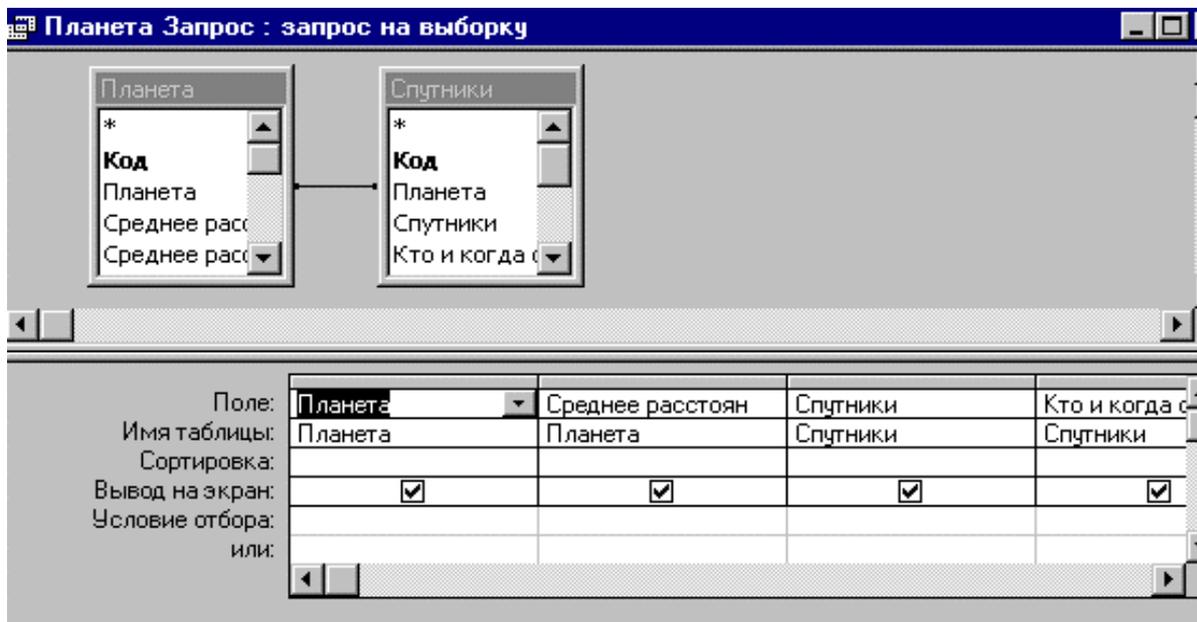


Рис. 73

Если пользователю необходимо получить часть данных из базы, он должен использовать специальные объекты – запросы. Все необходимые запросы пользователь базы должен подготовить заранее. Простейший запрос, называемый *запросом на выборку*, позволяет выбрать данные из полей таблиц, на основе которых он сформирован. На рис. 74 приведен пример организации простейшего запроса.

Использование систем управления базой данных позволяет выделить следующие виды учебной деятельности учащихся: организацию хранения и быстрого доступа к информации, что позволит использовать информационно-справочные системы, электронно-энциклопедические словари, электронные пособия по изучению различных тем и т.д.

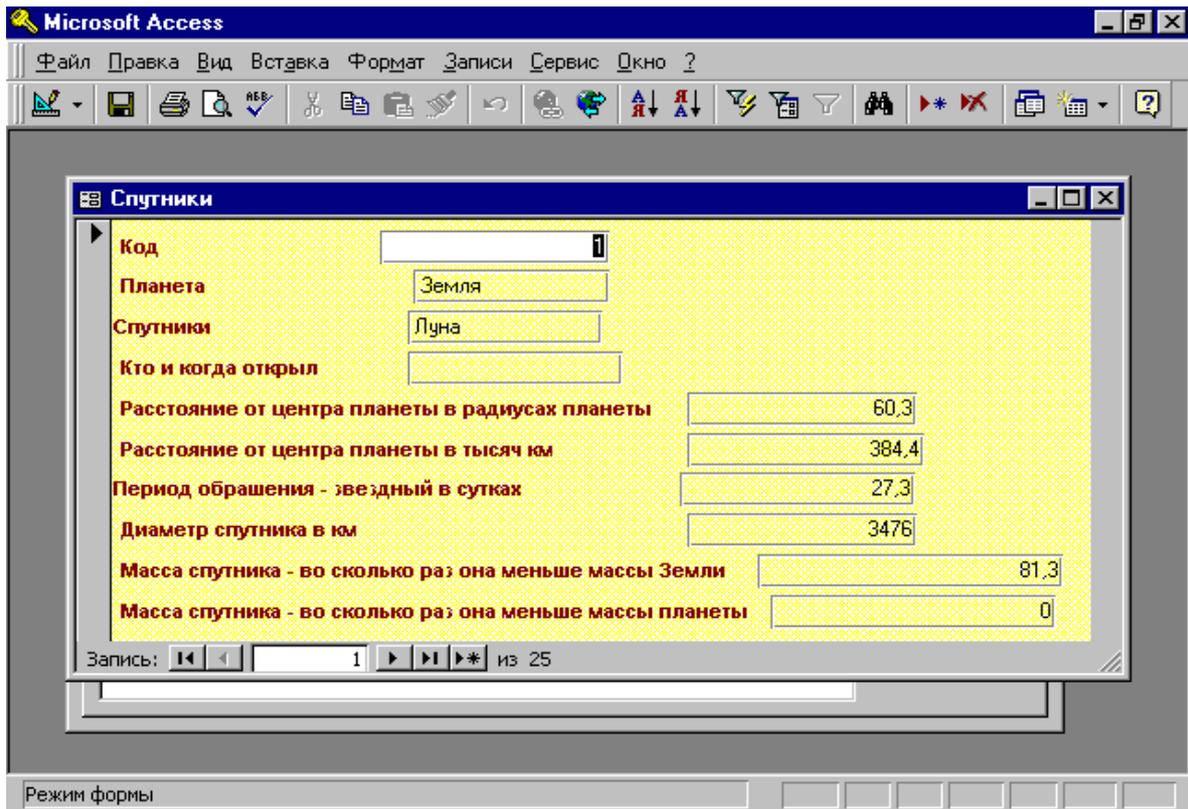


Рис. 74

7.5. МОДЕЛИРОВАНИЕ ЭЛЕКТРИЧЕСКИХ СХЕМ ИЗ ШКОЛЬНОГО КУРСА ФИЗИКИ В ИНТЕГРИРОВАННОЙ СИСТЕМЕ ELECTRONICS WORKBENCH

Научно-техническое проектирование является основным в развитии науки и техники. Одним из направлений является компьютерное схемотехническое моделирование электронных устройств.

Использование интегрированных программных систем схемотехнического моделирования аналоговых и цифровых радиоэлектронных устройств (Micro-Cap V, DesignLab 8.0, APLAC 7.0, System View 1.9, Circuit Maker 6.0, Electronics Workbench) позволяет решать следующие задачи:

- создание модели принципиальной электрической схемы устройства и ее редактирование;
- расчет режимов работы модели;
- расчет частотных характеристик и переходные процессы модели;
- провести оценку и анализ модели;

- наращивать библиотеку компонентов;
- представлять данные в форме, удобной для дальнейшей работы;
- разработку печатных плат;
- подготовку научно-технических документов и др.

Этот параграф посвящен системе Electronics Workbench 5.12, разработанной фирмой Interactive Image Technologies. Особенностью системы является наличие контрольно-измерительных приборов, по внешнему виду и характеристикам приближенных к их промышленным аналогам. Система легко усваивается и достаточно удобна в работе.

Подробно изучить приемы работы с системой можно по специальным пособиям и руководствам [19, 74]. Мы ограничимся первоначальным знакомством с системой и моделированием электрических схем из школьного курса физики.

7.5.1. Технология работы в системе Electronics Workbench 5.12

1. Знакомство с системой Electronics Workbench 5.12.

Запустив интегрированный пакет Electronics Workbench 5.12, вы увидите диалоговое окно и окно редактирования (рис.75). Окно редактирования заполнено некоторыми компонентами. Диалоговое окно Electronics Workbench содержит поле меню, библиотеку компонентов и линейку контрольно-измерительных приборов, расположенных в одном поле. Поле меню аналогично многим Windows-приложениям. Опции главного меню легко изучить самостоятельно.

Более подробно остановимся на некоторых компонентах и контрольно-измерительных приборах. На рис. 75 в окне редактирования, начиная слева (сверху), приведены обозначения следующих компонентов и контрольно-измерительных приборов: заземление, батарея, источник постоянного тока, источник переменного синусоидального тока (эффективное значение тока, частота, фаза), источник переменного синусоидального напряжения (эффективное значение тока, частота, фаза), резистор, конденсатор, катушка (индуктивность), трансформатор, переключатель, электролитический конденсатор, конденсатор переменной емкости, катушка переменной индуктивности, диод, стабилитрон, светодиод, диодный

мост, диод Шокли, $n - p - n$ транзистор, $p - n - p$ транзистор, далее 4 вида полевых транзисторов, вольтметр, амперметр, лампа накаливания (напряжение, мощность), светодиод (цвет свечения), мультиметр, осциллограф, измеритель амплитудно- и фазо-частотных характеристик.

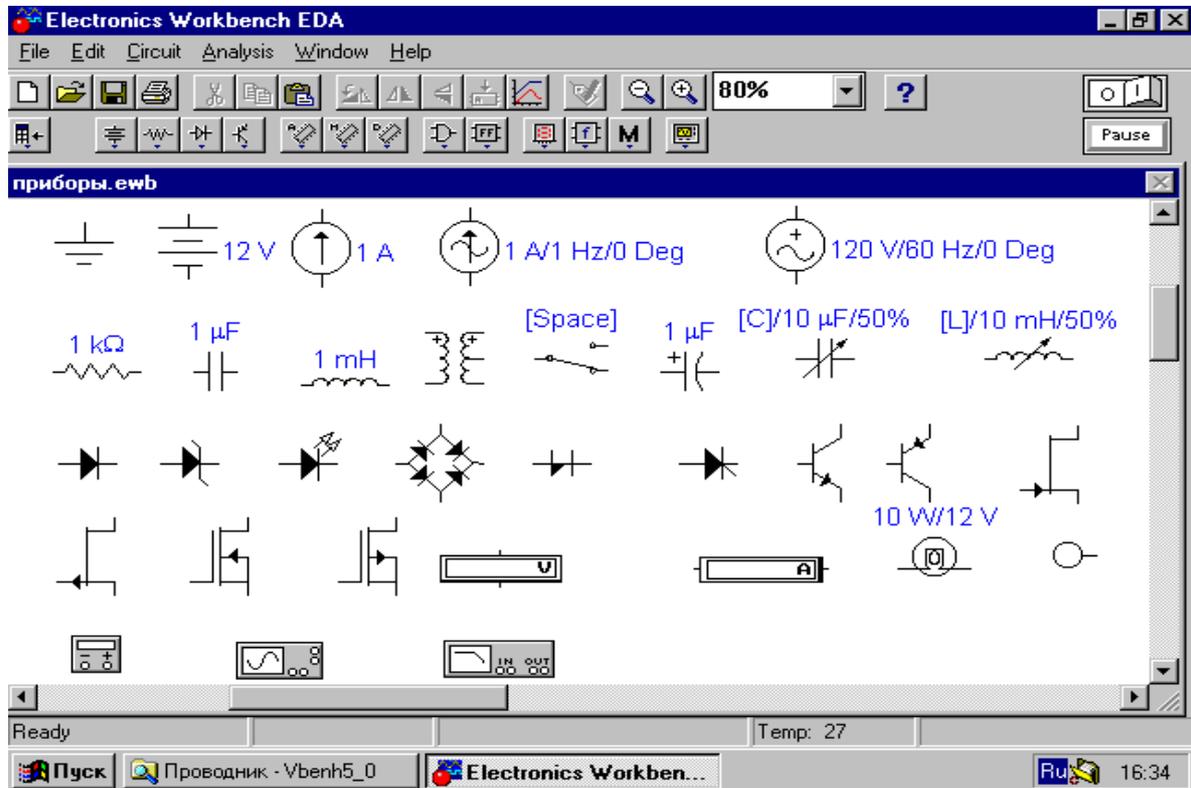


Рис. 75

2. Алгоритм технологии подготовки и запуска электрических схем.

1. Выбор необходимых компонентов электрической схемы и расположение их в окне редактирования Electronics Workbench 5.12.

Для этого подводим указатель мыши к одной из пиктограмм библиотеки компонентов или линейке контрольно-измерительных приборов и щелкаем левой кнопкой мыши. Выпадает одна из выбранных групп компонентов. Для того чтобы выбрать необходимый, подводим указатель мыши к компоненту, нажимаем левую кнопку мыши (не отпускаем кнопку), перемещаем компонент на окно редактирования, отпускаем кнопку.

2. Ввод и изменение параметров выбранных компонентов.

Подводим указатель мыши к компоненту в окне редактирования и щелкаем два раза левой кнопкой мыши. Выпадает меню, состоящее из нескольких опций. Рассмотрим два из них:

Label – необходим для обозначения компонента;

Value – необходим для простановки значений компонента.

В контрольно-измерительных приборах при необходимости, например в вольтметрах и амперметрах, при внесении параметров в опции *Label* указываем для какого тока, постоянного или переменного; в *Mode* выбираем DC – для постоянного тока, AC – для переменного.

3. Соединение компонентов электрической схемы.

После размещения компонентов и простановки параметров производится соединение их выводов проводниками. При этом необходимо учитывать, что к выводу компонента можно подключить только один проводник. Для выполнения подключения указатель мыши подводим к выводу компонента и после появления жирной точки (указатель соединения) нажимаем левую кнопку мыши, и появляющийся при этом проводник протягиваем к выводу другого компонента до появления на нем такой же жирной точки, после чего кнопку мыши отпускаем – соединение готово. Если соединение нужно разорвать, указатель мыши подводим к одному из выводов компонента или к точке соединения и при появлении указателя соединения нажимаем левую кнопку, проводник отводим на свободное место рабочего поля, после чего кнопку отпускаем. Если необходимо вывод компонента подключить к имеющемуся на схеме проводнику, то из вывода компонента проводник указателем мыши подводим к указанному проводнику и после появления точки соединения кнопку мыши отпускаем. Отметим, что прокладка соединительных проводов производится автоматически, причем препятствия – компоненты и проводники обгибаются по ортогональным направлениям (по горизонтали или вертикали).

4. Подключение электрической схемы к питанию.

В правом верхнем углу диалогового окна расположена пиктограмма

0	1
---	---

 :

0 – отключено питание; 1 – включено питание. После включения питания на контрольно-измерительных приборах регистрируются характеристики и значения собранной модели электрической схемы.

5. Контрольно-измерительные приборы. Панель контрольно-измерительных приборов находится под полем меню рабочего экрана и содержит:

1. Цифровой мультиметр;
2. Функциональный генератор;
3. Двухканальный осциллограф;
4. Измеритель амплитудно-частотных и фазо-частотных характеристик;



Рис. 76

5. Генератор слов (кодový генератор);
6. Восьмиканальный логический анализатор;
7. Логический преобразователь.

Общий порядок работы с приборами:

иконка прибора курсором переносится на рабочее поле и подключается к исследуемой схеме. Для приведения прибора в рабочее (развернутое) состояние необходимо дважды щелкнуть курсором на его иконке (рис. 76).

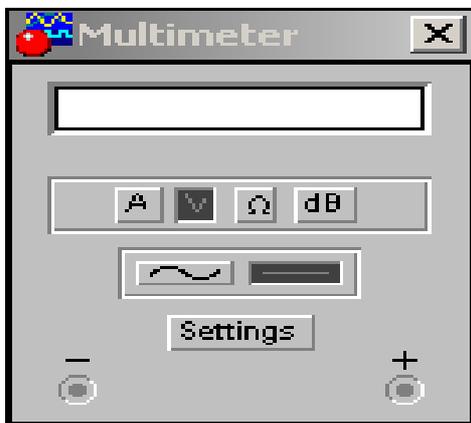
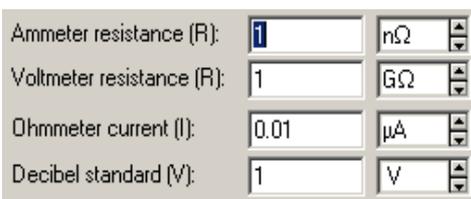


Рис. 77. Лицевая панель мультиметра

Мультиметр. На лицевой панели мультиметра (рис. 77) расположен дисплей для отображения результатов измерения, клеммы для подключения к схеме и кнопки управления:



 – выбор режима измерения тока, напряжения, сопротивления и ослабления (затухания);

 – выбор режима измерения переменного и постоянного тока;

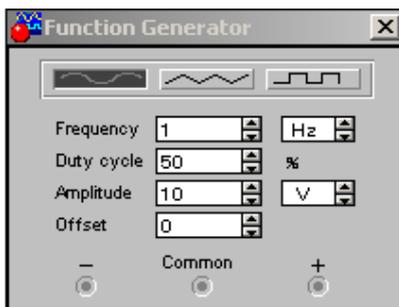


Рис. 78. Лицевая панель генератора

• – режим установки параметров мультиметра. После нажатия на эту кнопку открывается диалоговое окно, на котором обозначено:

Ammeter resistance – внутреннее сопротивление амперметра;

Voltmeter resistance – входное сопротивление вольтметра;
 Ohmmeter current – ток через контролируемый объект;
 Decibel standard – установка эталонного напряжения V_1 при измерении ослабления или усиления в децибелах (по умолчанию $V_1 = 1 \text{ В}$).

Функциональный генератор. Лицевая панель генератора показана на рис. 78. Управление генератором осуществляется следующим образом:

 – выбор формы выходного сигнала: синусоидальный (выбран по умолчанию), треугольный и прямоугольный;

Frequency 1 Hz – установка частоты выходного сигнала.

Duty cycle 50 % – установка коэффициента заполнения в %: для импульсных сигналов это отношение длительности импульса к периоду повторения – величина, обратная скважности; для треугольных сигналов – соотношение между длительностями переднего и заднего фронта;

Amplitude 10 V – установка амплитуды выходного сигнала;

Offset 0 – установка смещения (постоянной составляющей) выходного сигнала;

 – выходные зажимы; при заземлении клеммы COM (общий) на клеммах “-” и “+” получаем парафазный сигнал.

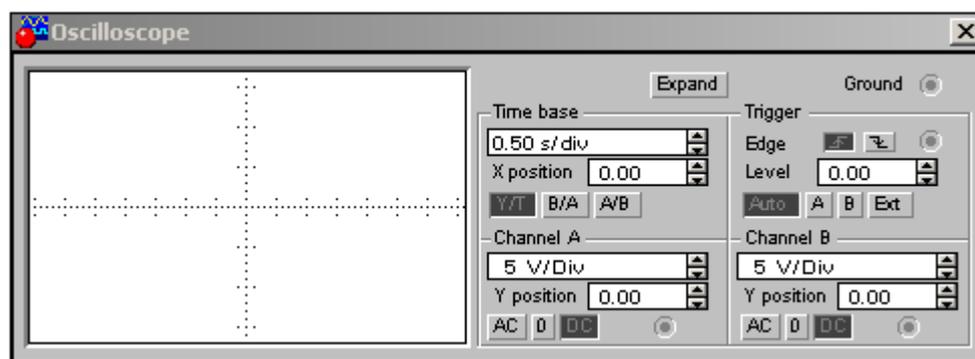


Рис. 79. Лицевая панель осциллографа

Осциллограф. Лицевая панель осциллографа представлена на рис. 79. Осциллограф имеет 2 канала А и В с разделенной регули-

ровкой чувствительности в диапазоне от 10 мкВ/дел до 5 кВ/дел и регулировкой смещения по вертикали (Y POS). Выбор режима по входу осуществляется нажатием кнопок **AC** **0** **DC**.

- Режим **AC** предназначен для наблюдения только сигналов переменного тока

- В режиме **0** входной зажим замыкается на землю.

- В режиме **DC** (включено по умолчанию) можно проводить измерения как постоянного, так и переменного тока.

Режим развертки выбирается кнопками **Y/T** **B/A** **A/B**.

- В режиме **Y/T** (обычный режим, включен по умолчанию) реализуются следующие режимы развертки: по вертикали – напряжение сигнала, по горизонтали – время;

- В режиме **B/A**: по вертикали – сигнал канала В, по горизонтали – сигнал канала А;

- В режиме **A/B**: по вертикали – сигнал канала А, по горизонтали – сигнал канала В.

В режиме **Y/T** длительность развертки (TIME BASE) может быть задана в диапазоне от 0.1 нс/дел до 1 с/дел с возможностью установки смещения в тех же единицах по горизонтали, т.е. по оси X (X POS).

При нажатии кнопки **EXPAND** лицевая панель осциллографа существенно меняется – увеличивается размер экрана (рис.80), появляется возможность прокрутки изображения по горизонтали и его сканирования с помощью вертикальных линий (синего и красного цвета), которые за треугольное ушко могут быть установлены в любое место экрана. При этом в индикаторных окошках под экраном приводятся результаты измерения напряжения, временных интервалов и их приращений.

Изображение можно инвертировать нажатием клавиши **REVERSE**, и записать данные в файл нажатием кнопки **SAVE**. Возврат к исходному состоянию осциллографа – нажатием кнопки **REDUCE**.

Измеритель АЧХ и ФЧХ. Измеритель предназначен для анализа амплитудно-частотных (при нажатой кнопке **MAGNITUDE**, включена по умолчанию) и фазо-частотных (при нажатой кнопке **PHASE**) характеристик при логарифмической (кнопка **LOG**, включена по умолчанию) или линейной (кнопка **LIN**) шкале по осям Y и X (рис.81). Настройка измерителя заключается в выборе

пределов измерения коэффициента передачи и вариации частоты с помощью кнопок в окошках F – максимальное и I – минимальное значение.

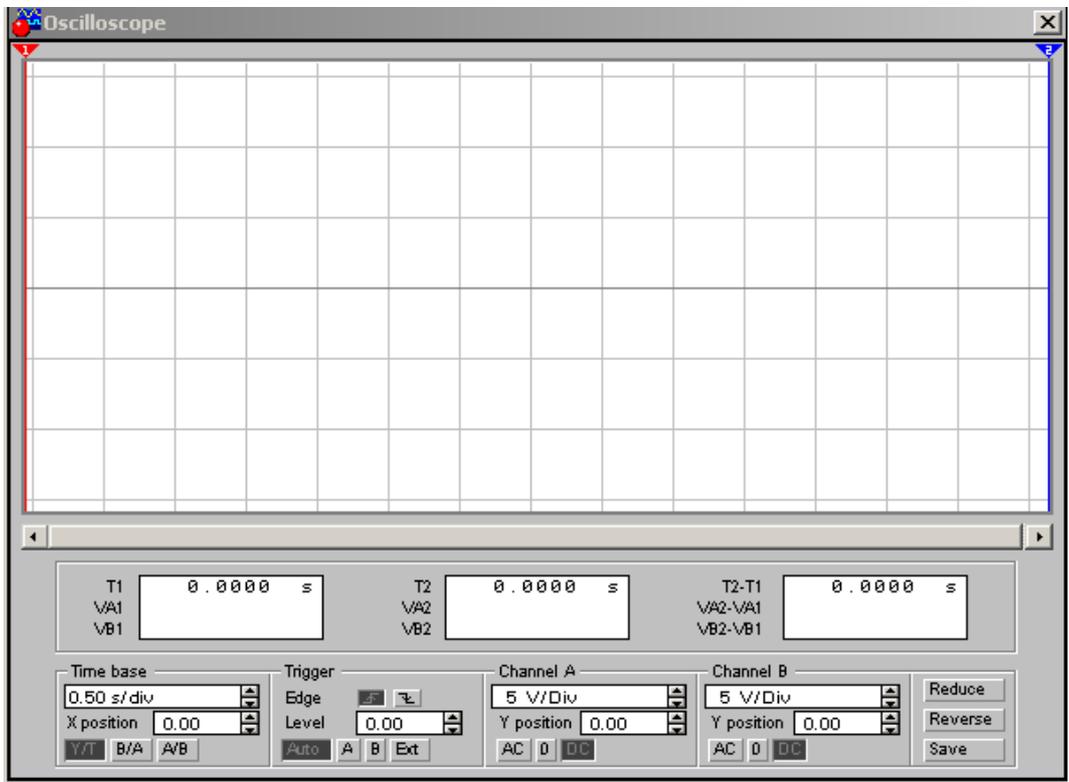


Рис. 80. Лицевая панель осциллографа в режиме EXPAND

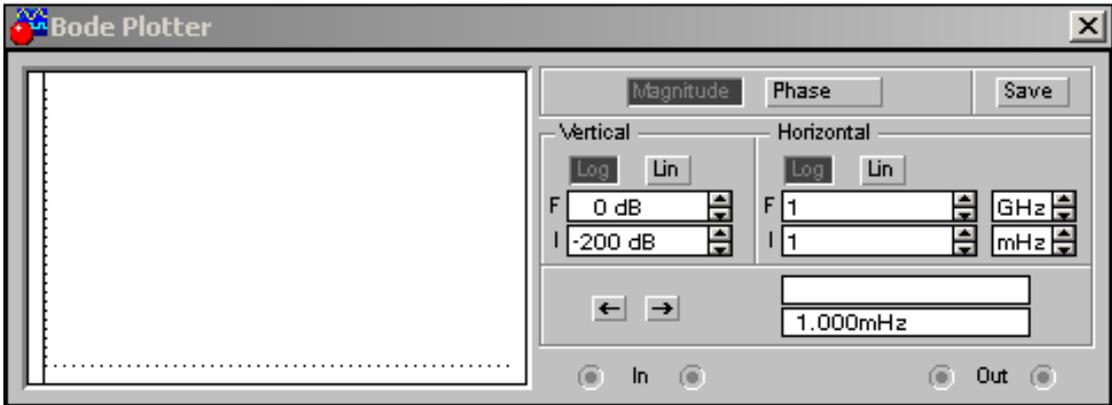


Рис. 81. Лицевая панель АЧХ и ФЧХ

Подключение прибора к исследуемой схеме осуществляется с помощью зажимов IN (вход) и OUT (выход). Левые клеммы зажимов подключаются соответственно к входу исследуемого устройства, а правые – к общей шине. К входу устройства необходи-

мо подключить функциональный генератор или другой источник переменного напряжения, при этом каких-либо настроек в этих устройствах не требуются

6. Некоторые возможности электронной лаборатории. Прежде чем приступать к разработке или к выполнению лабораторных работ в системе Electronworkbench, необходимо приобрести навыки работы с этой системой. Поэтому рассмотрим несколько заданий.

Задание 1. Создать электронную модель однородного участка электрической цепи в электронной лаборатории Electronics Workbench 5.12 (рис. 82). Проверить справедливость закона Ома для этой цепи.

Решение представлено на рис. 82.

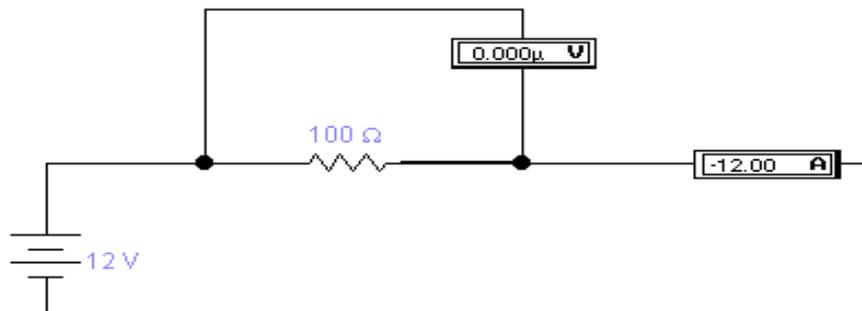


Рис. 82

Параметры компонентов цепи редактируются в окнах, открывающихся двойным щелчком мыши на элементах схемы.

Задание 2. Изучить прибор «осциллограф». В режиме В/А осциллографа получить фазовую диаграмму (рис. 83)

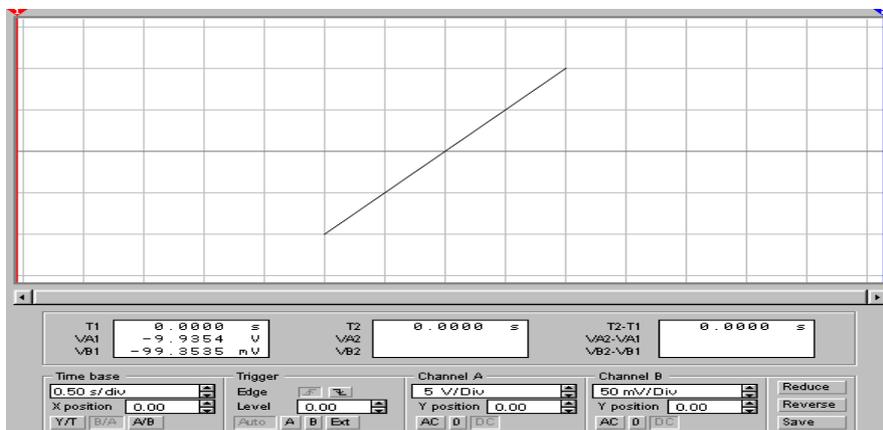


Рис. 83

Решение. Для этого собираем схему, показанную на рис.84

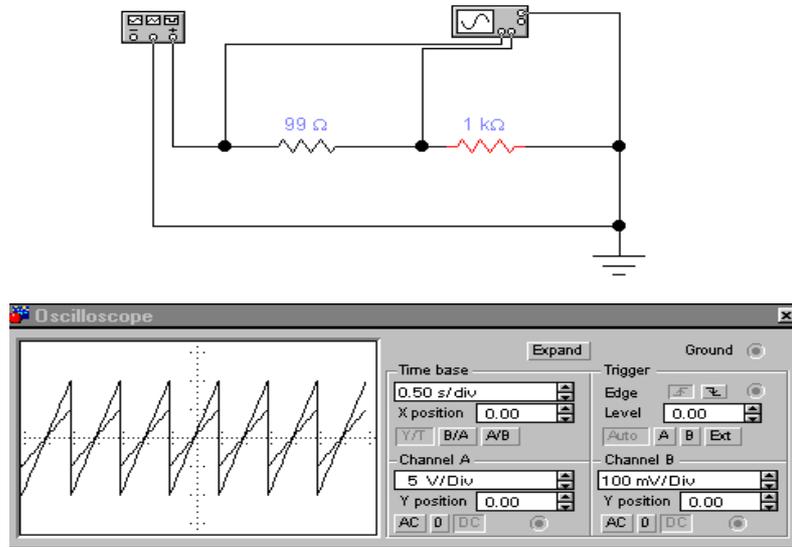


Рис. 84

Воспользуемся функциональным генератором для получения вольтамперной характеристики резисторов, подавая на резистор R1 пилообразное напряжение. Длительность первого полупериода установим равной 99% периода. Напряжение генератора будем подавать на канал А осциллографа с резистора R1. Напряжение будем подавать на канал В осциллографа с дополнительного резистора R2.

Задание 3. Получить диаграмму в графическом формате.

Решение. Сохранение диаграммы в графическом формате и ее редактирование осуществляется с помощью программы Аналитические графики (Analysis Graphs) (рис.85).

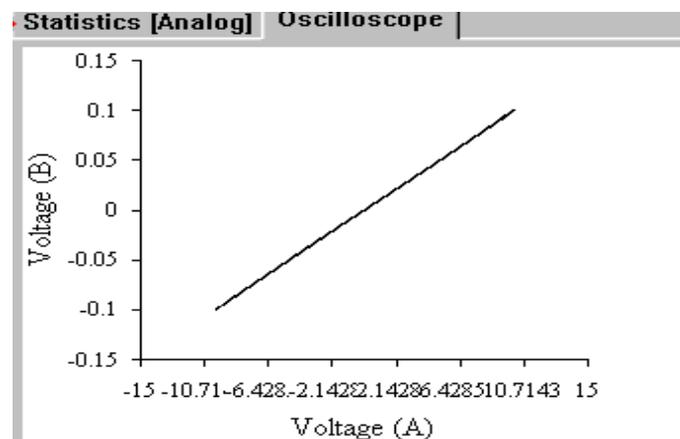


Рис. 85

Для установки параметров осей, графиков и линий сетки щелчком правой кнопки мыши вызывается контекстное меню, в котором надо выбрать команду Свойства.

Задание 4. Создать текстовый файл.

Решение. Для этого найти устройство записи данных Write Data в приложении Electronics Workbench (рис.86). Это устройство позволяет сохранить в текстовом файле (данные из текстового файла приведены ниже) уровни потенциала в 8 точках измерения. Эти данные можно вставить в окно Описаний электронной лаборатории Electronics Workbench .

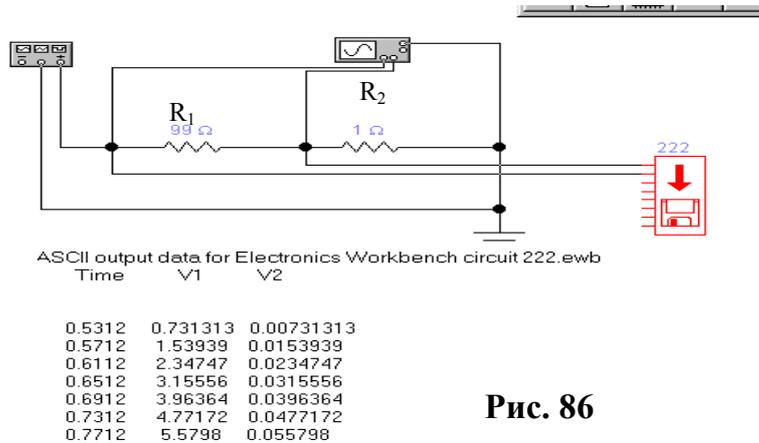


Рис. 86

Результаты, полученные из текстового файла типа txt.

ASCII output data for Electronics Workbench circuit 222.ewb		
Time	V1	V2
0.5312	0.731313	0.00731313
0.5712	1.53939	0.0153939
0.6112	2.34747	0.0234747
0.6512	3.15556	0.0315556
0.6912	3.96364	0.0396364
0.7312	4.77172	0.0477172
0.7712	5.5798	0.055798
0.8112	6.38788	0.0638788
0.8512	7.19596	0.0719596
0.8912	8.00404	0.0800404
0.9312	8.81212	0.0881212
0.9712	9.6202	0.096202
0.99	10	0.1

Задание 5. Копировать любую область окна Electronics Workbench.

Решение. В Electronics Workbench можно имитировать и ручной режим проведения опыта: увеличивать напряжение на резисторе 100 Ом, уменьшая сопротивление переменного резистора нажатием клавиши R (или любую, назначенную пользователем) согласно определенному декременту. Показания приборов в этом случае просто вводятся с клавиатуры в окно Description. На рис. 87 изображен скопированный в буфер обмена результат такого эксперимента.

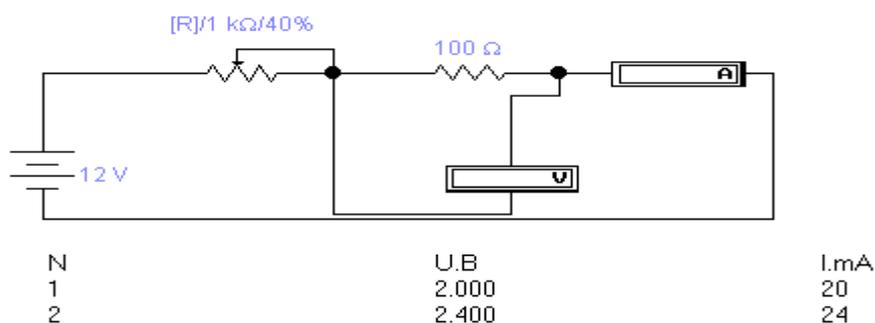


Рис. 87

Копирование любой области окна Electronics Workbench 5.12 производится с помощью команды Копировать как точечный рисунок (Copy as Bitmap), просмотр содержимого буфера обмена – с помощью команды Показать буфер обмена (Show Clipboard) в меню Правка (Edit). Копированный таким образом рисунок также можно вставить в текст Word.

Приведем примеры для самостоятельной лабораторной работы [19].

7.5.2. Цепи постоянного тока

Цель работы: изучить закон Ома для участка цепи.

Оборудование: электронная лаборатория Electronic Workbench 5.12.

Закон Ома для участка цепи: ток в проводнике I равен отношению падения напряжения U на участке цепи к ее электрическому сопротивлению R : $I=U/R$.

Закон Ома иллюстрируется схемой на рис.88, из которой видно, что на участке цепи с сопротивлением $R_2 = 100$ Ом создается

падение напряжения $V_2=9,09$ В, измеряемое вольтметром V_2 . Согласно закону Ома, ток в цепи $I=9,09/100 = 90,9$ мА, что измеряет последовательно включенный в цепь амперметр. Отметим, что в рассматриваемой схеме внутреннее сопротивление амперметра выбрано равным 10^{-12} Ом, очень малым, а входное сопротивление вольтметра – 10^{12} Ом, т.е. очень большим, чтобы подключение измерительных приборов к цепи не оказывало сколько-нибудь заметного влияния на результаты измерений.

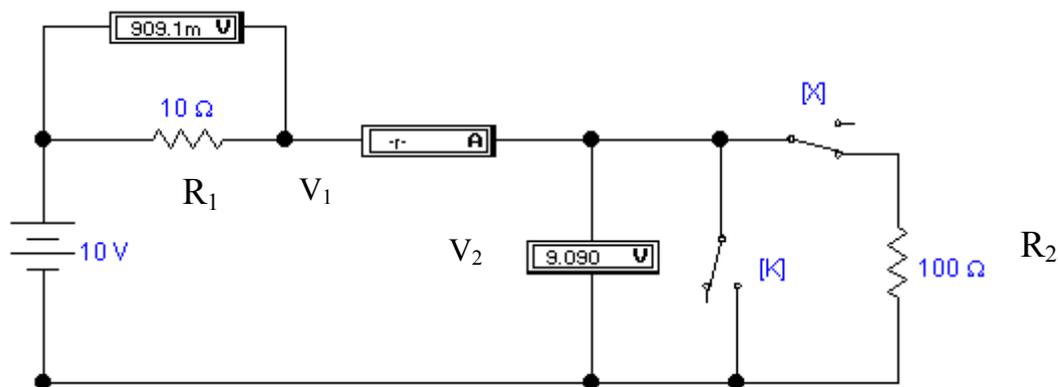


Рис. 88. Простейшая цепь постоянного тока

Отметим назначение ключей K и X на рис.88, управляемых нажатием клавиш клавиатуры. При размыкании ключа X и замыкании K в схеме вольтметр V_1 измеряет ЭДС источника: $E=10$ В, а вольтметр V_2 имеет нулевые показания. При размыкании ключа K в схеме ток $I=E/R_1=10/10=1$ А. При этом вольтметр V_1 измеряет падение напряжения $U=I_0R_1=10$ В.

Закон Ома для полной цепи: ток в замкнутой электрической цепи равен ЭДС источника E , деленной на сопротивление всей цепи. Применительно к цепи на рис.88 ее полное сопротивление равно R_1+R_2 , и на основании закона Ома получаем: $I = E / (R_1 + R_2) = 90,9$ мА, что и измеряет амперметр.

Задание для выполнения работы. Для схемы на рис.88 проведите моделирование режимов холостого хода и короткого замыкания и сравните полученные результаты с расчетными значениями.

Контрольные вопросы

1. Чем отличается закон Ома для участка цепи от закона Ома для полной цепи?
2. Что представляют собой режим холостого хода и режим короткого замыкания?

7.5.3. Источники вторичного питания. Выпрямление с фильтрацией

1. Постановка задачи.

Рассмотреть выпрямление электрического напряжения переменного тока с помощью системы Electronics Workbench 5.0 С. Цель фильтрации при выпрямлении – ослабить переменную составляющую выпрямленного напряжения.

Здесь необходимо промоделировать работу (рис.89, 90) фильтра выпрямителя. На входе фильтра действует выпрямленное напряжение $E(t)$ одно- или двухполупериодное, полученное из переменного синусоидального напряжения с помощью генератора сигналов. Для ослабления переменной составляющей в схеме фильтра имеется реактивный элемент: конденсатор емкостью $C = 800$ мкф. Фильтр работает на нагрузку сопротивлением $R_H = 0,5$ Ком.

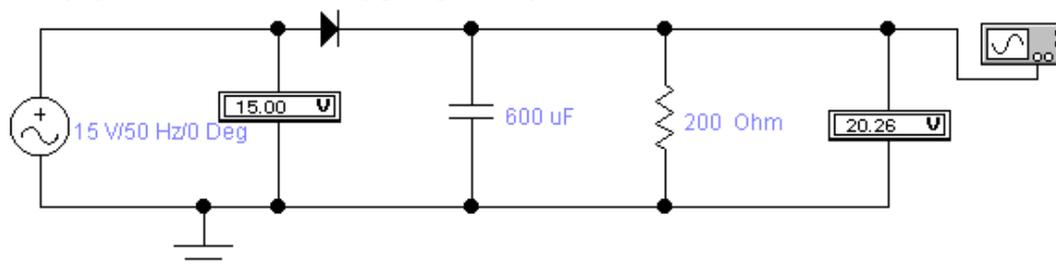


Рис. 89. Однополупериодный выпрямитель

Однополупериодную схему обычно применяют при выпрямленных токах до нескольких десятков миллиампер и в тех случаях, когда не требуется высокой степени сглаживания выпрямленного напряжения. Эта схема характеризуется низким коэффициентом использования мощности трансформатора.

Однофазная мостовая схема характеризуется высоким коэффициентом использования мощности и поэтому может быть рекомендована для использования в устройствах повышенной мощ-

ности при выходных напряжениях от десятков до сотен вольт; пульсации такие же, как и в предыдущей схеме.

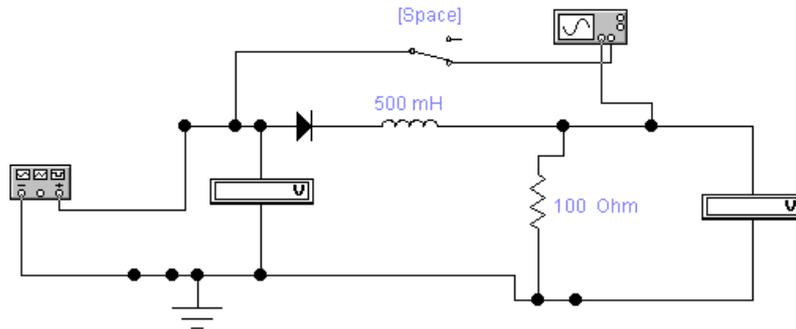


Рис. 90

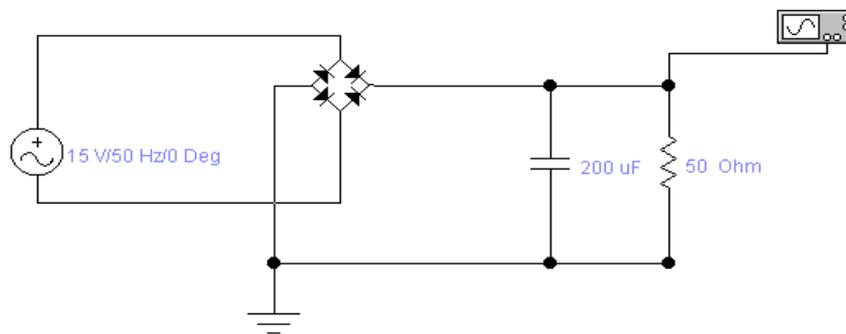


Рис. 91. Мостовая схема

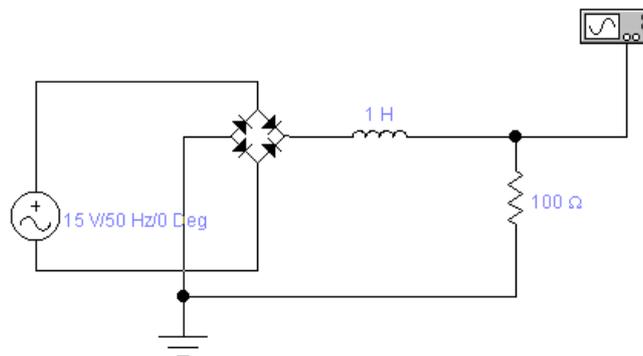


Рис. 92

Симметричная схема удвоения напряжения представляет собой последовательное соединение двух однополупериодных выпрямителей и применяется при высоких напряжениях (до 1...2 кВ) при небольших токах нагрузки (рис. 91). Пульсации на каждом конденсаторе схемы удвоения в 2 раза больше пульсаций на ее входе.

2. Задания к лабораторной работе.

2.1. Собрать схему выпрямления и фильтрации тока по приведенным схемам на рис. 89, 90. Провести моделирование вы-

прямления тока. Исследовать в зависимости от параметров фильтра как изменяется значение выходного выпрямленного напряжения.

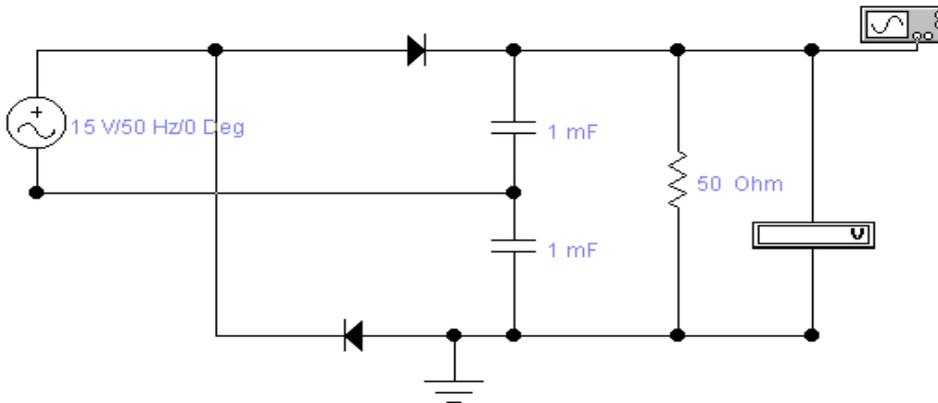


Рис. 93. Схема удвоения напряжения

2.2. Заменить выпрямитель двухполупериодным (рис. 91, 92).. Рассмотреть, насколько отличаются результаты моделирования.

2.3. Найти оптимальные параметры сопротивления и емкости, при котором получается полное выпрямление.

2.4. Рассмотреть, насколько отличаются результаты предыдущих схем выпрямления от схемы удвоения напряжения (рис. 93).

2.5 Зарисовать изображение выходного напряжения $u(t)$ для приведенных схем выпрямления.

3. Результаты моделирования. На рис. 94 изображено выходное $u(t)$ напряжение фильтра для однополупериодного выпрямителя. Выбор определенного напряжения $E(t)$ на вход выпрямителя осуществляется с помощью генератора переменного напряжения. С помощью изменения параметров фильтра r , C и нагрузки R_H можно оценить качество фильтрации.

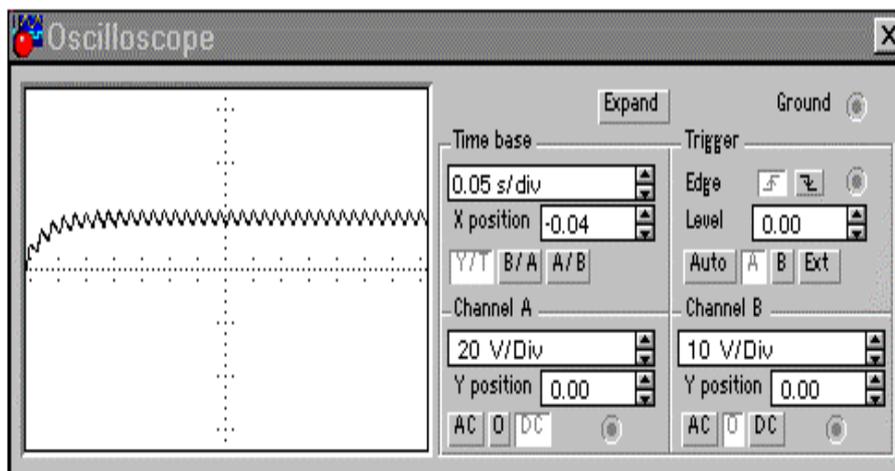


Рис. 94

7.6. ИЗУЧЕНИЕ ОСНОВ ЭЛЕКТРОНИКИ И ВЫЧИСЛИТЕЛЬНОЙ ТЕХНИКИ В ELECTRONICS WORKBENCH

Основная трудность в изложении элементов цифровой техники состоит в существенном разрыве между уровнем знаний студентов I, II курсов, школьников и современным состоянием ЭВТ. Учащимся необходимо преодолеть дистанцию огромного размера – от двоичной арифметики и простейших логических элементов до архитектуры микропроцессора и ЭВМ. Многообразие элементной базы, ее миниатюризация, отсутствие наглядности, необходимость использования различных кодов, синтез многополюсников требуют у учащихся высокого уровня абстрактного мышления. Изучение базовых логических элементов, элементов памяти, операционных элементов, комбинационной и последовательной логики на физическом уровне становится невозможным из-за громоздкости и отсутствия наглядности. Они рассматриваются схемотехнически: зависимость между входными и выходными сигналами описывается таблицами истинности или функциями на языке алгебры логики.

В качестве компьютерной среды изучения основ электроники и вычислительной техники нами выбрана система Electronics Workbench, разработанная фирмой Interactive Image Technologies. Особенностью системы является наличие контрольно-измерительных приборов, по внешнему виду и характеристикам приближенных к их промышленным аналогам. Система легко усваивается и достаточно удобна в работе. Ниже рассмотрены основные понятия булевой алгебры, примеры решения задач с помощью логических схем и лабораторные работы по основам электроники и вычислительной техники, составленных на основе работы [19].

7.6.1. Основные понятия булевой алгебры

Логические операции:

- \wedge , X , \cdot , $\&$, «и», and – конъюнкция.
- \vee , $+$, «или», or – дизъюнкция.
- \neg , $-$, not – отрицание.

Таблицы истинности

<i>A</i>	<i>B</i>	$A \wedge B$
0	0	0
0	1	0
1	0	0
1	1	1

	<i>B</i>	$A \vee B$
0	0	0
0	1	1
1	0	1
1	1	1

<i>A</i>	\bar{A}
0	1
1	0

Основные формулы алгебры-логики:

Законы коммутативности: $A \wedge B = B \wedge A$
 $A \vee B = B \vee A$

ассоциативности: $(A \vee B) \vee C = A \vee (B \vee C)$
 $(A \wedge B) \wedge C = A \wedge (B \wedge C)$.

идемпотентности: $A \vee A = A$
 $A \wedge A = A$.

дистрибутивности: $A \wedge (B \vee C) = (A \wedge B) \vee (A \wedge C)$
 $A \vee (B \wedge C) = (A \vee B) \wedge (A \vee C)$.

Формулы, позволяющие упрощать логические выражения:

$\overline{\overline{A}} = A$	$\overline{\overline{A \vee B}} = \overline{\overline{A} \wedge \overline{B}}$	$\overline{\overline{A \wedge B}} = \overline{\overline{A} \vee \overline{B}}$
$A \vee B = \overline{\overline{A} \wedge \overline{B}}$	$A \wedge \overline{A} = 0$	$A \wedge 0 = 0$
$A \wedge \overline{A} = 0$	$A \wedge 1 = A$	$A \vee 1 = 1$

В логических функциях скобки указывают последовательность выполнения операций. При отсутствии скобок первой выполняется операция отрицания, затем конъюнкция, а потом дизъюнкция.

На рис. 95 приведены логические элементы – конъюнкция, дизъюнкция, отрицание, построенные на базе транзисторов.

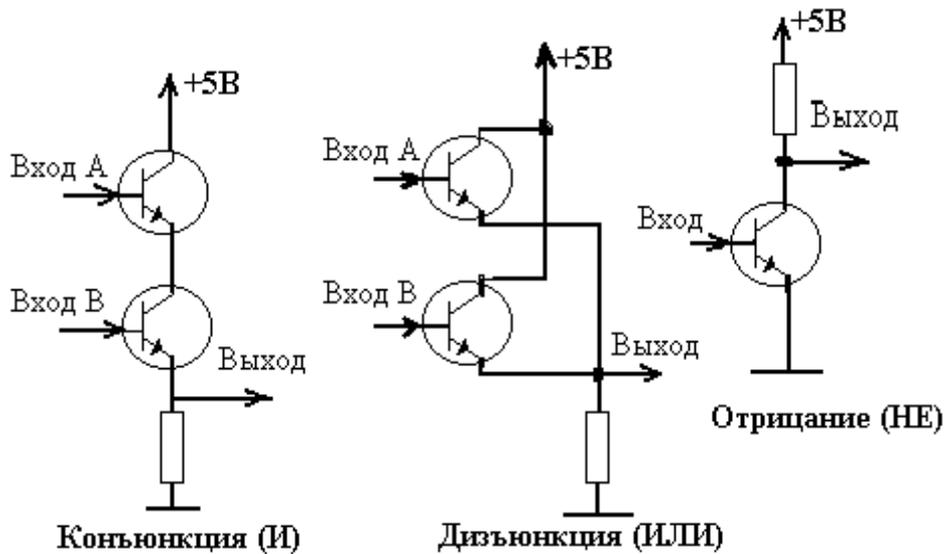


Рис. 95

Промышленность выпускает сотни типов электронно-логических элементов, в интегральном исполнении представляющих собой сочетание элементов «И», «ИЛИ», «НЕ». В виде примера рассмотрим один из самых распространенных типов логических микросхем типа К155ЛА3 (рис. 96), представляющее собой сочетание в одном корпусе четырех двухходовых схем «И»—«НЕ». Каждая логическая схема «И», «НЕ» имеет два входа (выводы 1 и 2, 4 и 5, 9 и 10, 12 и 13) и один выход (выводы 3, 6, 8, 11).

Таблица истинности для микросхемы К155ЛА3.

A	B	$A \wedge B$	$\overline{A \wedge B}$
0	0	0	1
0	1	0	1
1	0	0	1
1	1	0	0

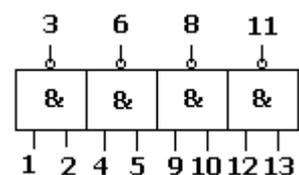


Рис. 96

Принятые обозначения логических элементов в электрических схемах приведены на рис. 97.

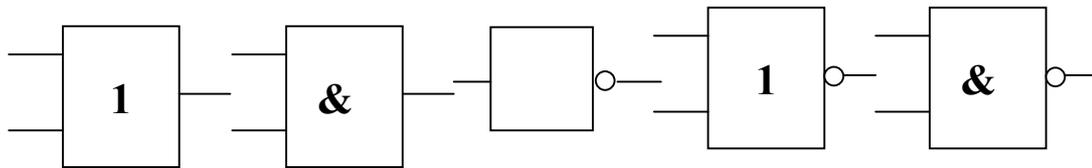


Рис. 97

Логическое сложение (дизъюнкция) – ИЛИ, логическое умножение (конъюнкция) – И, отрицание – НЕ, логический элемент «2-ИЛИ – НЕ», логический элемент «2-И – НЕ». Обозначения логических операций: * – конъюнкция, + – дизъюнкция, ‘ (апостроф) – отрицание.

7.6.2. Примеры решения задач на тему «Логические схемы»

• **Задача 1.** Проведите анализ логического устройства (рис. 98): по функциональной схеме составьте структурную формулу, упростите ее, если это возможно.

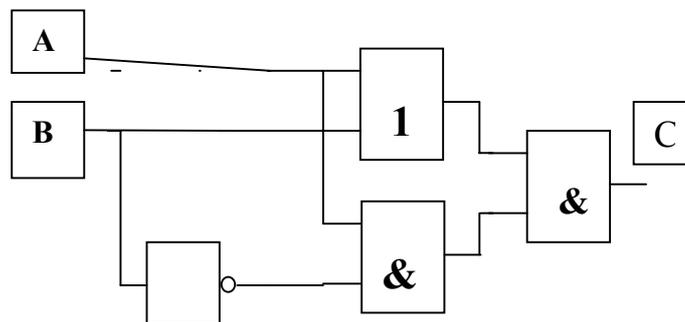


Рис. 98

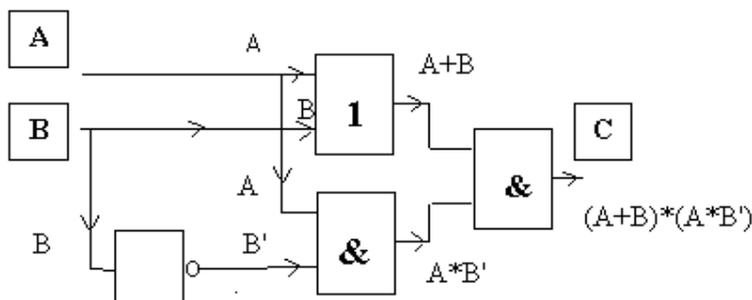


Рис. 98а

Решение. 1. Составление логической функции для функциональной (логической) схемы. При составлении логической функции необходимо проследить пути движения потоков сигналов (рис. 98а).

Ответ: $(A+B)*(A*B')$.

2. Проверка на избыточность функциональной схемы (упростить логическую функцию, т. е. преобразовать с помощью законов алгебры-логики).

$(A+B) \cdot (A \cdot B')$ = / Скобки для $A \cdot B'$ опускаем, так как перед скобками тоже знак \cdot / $= (A+B) \cdot A \cdot B' =$ / Для $A \cdot B'$ применяем закон коммутативности / $= (A+B) \cdot B' \cdot A =$ / Для $(A+B) \cdot B'$ применяем закон дистрибутивности / $= ((A \cdot B') + (B \cdot B')) \cdot A =$ / $B \cdot B' = 0$ / $= ((A \cdot B') + 0) \cdot A =$ / Поглощение 0 при дизъюнкции / $= (A \cdot B') \cdot A =$ / Скобки опускаем, применяем закон коммутативности / $= A \cdot A \cdot B' =$ / $A \cdot A = A$ / $= A \cdot B'$.

3. Проверяем справедливость логических преобразований. Для этого составляем таблицу истинности. В общем случае составляем две таблицы: для исходной и конечной логических функций. В данной задаче достаточно одной. Значения таблиц истинности $A \cdot B'$ и $(A+B) \cdot (A \cdot B')$ равны, что доказывает справедливость логических преобразований.

A	B	A+B	B'	$A \cdot B'$	$(A+B) \cdot (A \cdot B')$
0	0	0	1	0	0
0	1	1	0	0	0
1	0	1	1	1	1
1	1	1	0	0	0

4. По полученной логической функции составляем функциональную схему (рис.99).

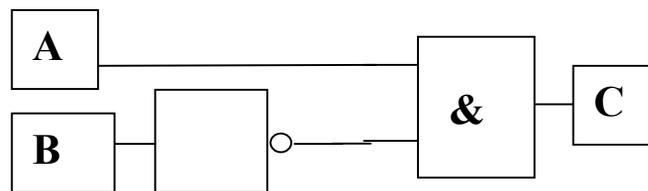


Рис. 99

Задача 2. Проведите синтез трехвходового логического устройства с выходной комбинацией 00110010 в таблице истинности.

Решение. 1. Составим таблицу истинности для данного логического устройства.

A	B	C	F(A,B,C)	
0	0	0	0	
0	0	1	0	
0	1	0	1	$A' * B * C' = 1$
0	1	1	1	$A' * B * C = 1$
1	0	0	0	
1	0	1	0	
1	1	0	1	$A * B * C' = 1$
1	1	1	0	

Так как в таблице в графе F единиц меньше, чем нулей, то построим СДНФ: $(A' * B * C') + (A' * B * C) + (A * B * C')$.

2. Используя правила алгебры-логики, попробуем его упростить:

$$\begin{aligned} (A' * B * C') + (A' * B * C) + (A * B * C') &= [((A' * B) * C') + ((A' * B) * C)] + \\ &+ (A * B * C') = [(A' * B) * (C' + C)] + (A * B * C') = / C' + C = 1 / = \\ &= [(A' * B) * 1] + (A * B * C') = (A' * B) + (A * B * C') = (A' * B) + ((A * C') * B) = \\ &= (A' + (A * C')) * B = B * (A' + A * C'). \end{aligned}$$

Упрощаем дальше, используя закон де Моргана:

$$\begin{aligned} B * (A' + A * C') &= B * (A' * (A * C')')' = B * (A * (A' + C'))' = \\ &= B * (A * (A' + C))' = B * ((A * A') + (A * C))' = / A * A' = 0, \\ 0 + (A * C) &= A * C = B * (A * C)' = B * (A' + C') = B * A' + B * C'. \end{aligned}$$

(Проверку можно осуществить с помощью таблиц истинности).

Ответ: $B * A' + B * C'$.

3. По полученной структурной формуле построим функциональную схему (рис.100).

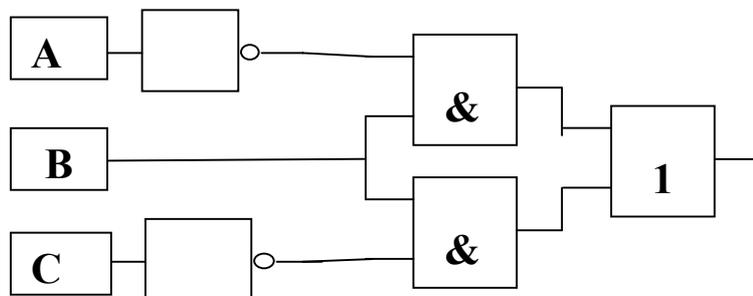


Рис. 100

7.6.3. Виртуальный логический конвертор (Logic converter)

Цель работы: Изучение назначения и принципа работы виртуального устройства логического конвертора (преобразователя). Знакомство с базовыми функциями логического конвертора.

Оборудование: Электронная лаборатория Electronics Workbench.

Краткая теория

Теперь для решения предложенных выше задач воспользуемся программой электронной лаборатории Electronics Workbench. Ниже приведены лабораторные работы использования этой интегрированной системы. Для построения логических схем в библиотеке Logic Gates (логические элементы) предусмотрена возможность выбора логических элементов. На рис. 101 изображен перечень выбора возможных логических элементов.

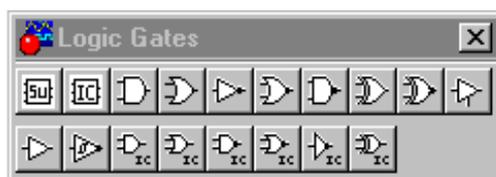


Рис. 101

На рис. 102 показаны обозначения логических элементов, используемые в Electronics Workbench: конъюнкции – И, дизъюнкции – ИЛИ, отрицания – НЕ, 2 – И – НЕ, 2 – ИЛИ – НЕ.

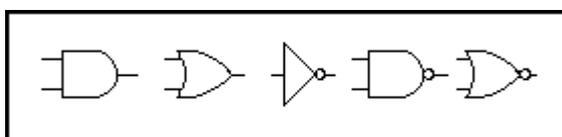


Рис. 102

В электронной лаборатории Electronics Workbench имеется виртуальное устройство. Логический конвертор (Logic Converter) позволяет осуществлять шесть логических преобразований для логической функции с числом переменных от 1 до 8: представление таблицы истинности, собранной из логических элементов схемы; об-

ращение таблицы истинности в логическую формулу (СДНФ); минимизацию СДНФ; обращение формулы в таблицу истинности; представление формулы в виде схемы в логическом базисе 2-И–НЕ. Логический конвертор выбирается из меню Instruments (рис. 103).



Рис. 103

Приведем описание технологии исследования логических схем с помощью логического конвертора (преобразователя).

1. Собираем логическую схему.
2. Подключаем исследуемую логическую схему к логическому конвертору (входов восемь, выход один – расположен справа).
3. Открываем логический конвертор щелчком левой кнопки мыши по иконке конвертора. На экране появляется меню Logic Converter (рис. 104).
4. Для получения таблицы истинности нажимаем



5. Для получения логической функции (структурной формулы) нажимаем



С помощью логического конвертора можно проводить не только анализ логических устройств, но их синтез.

Приведем описание технологии синтеза логического устройства по выходной комбинации с помощью логического конвертора (преобразователя).

1. Раскрываем лицевую панель логического конвертора (рис. 104).

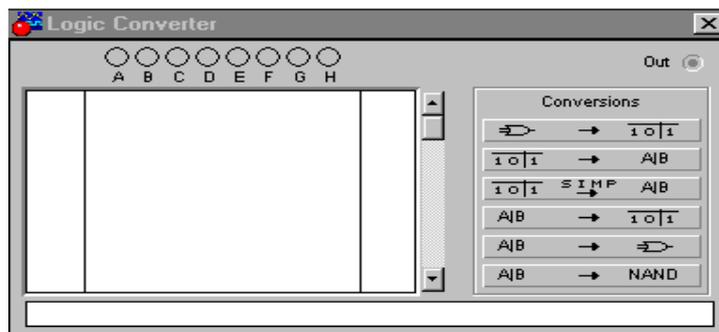


Рис. 104

2. Активизируем курсором клеммы-кнопки А, В, ... Н (начиная с F), количество которых равно количеству входов синтезируемого устройства (количеству логических переменных).

3. Вносим необходимые изменения в столбец OUT и после нажатия клавиши на панели преобразователя получаем результат в виде схемы на рабочем поле программы и логическую функцию в дополнительном дисплее.

Задача 3. Проведите анализ логического устройства (рис. 105) по функциональной схеме с помощью Electronics Workbench.

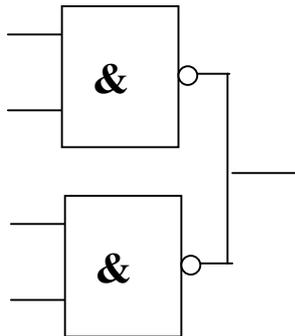


Рис. 105

На рис. 106 – решение задачи в Electronics Workbench.

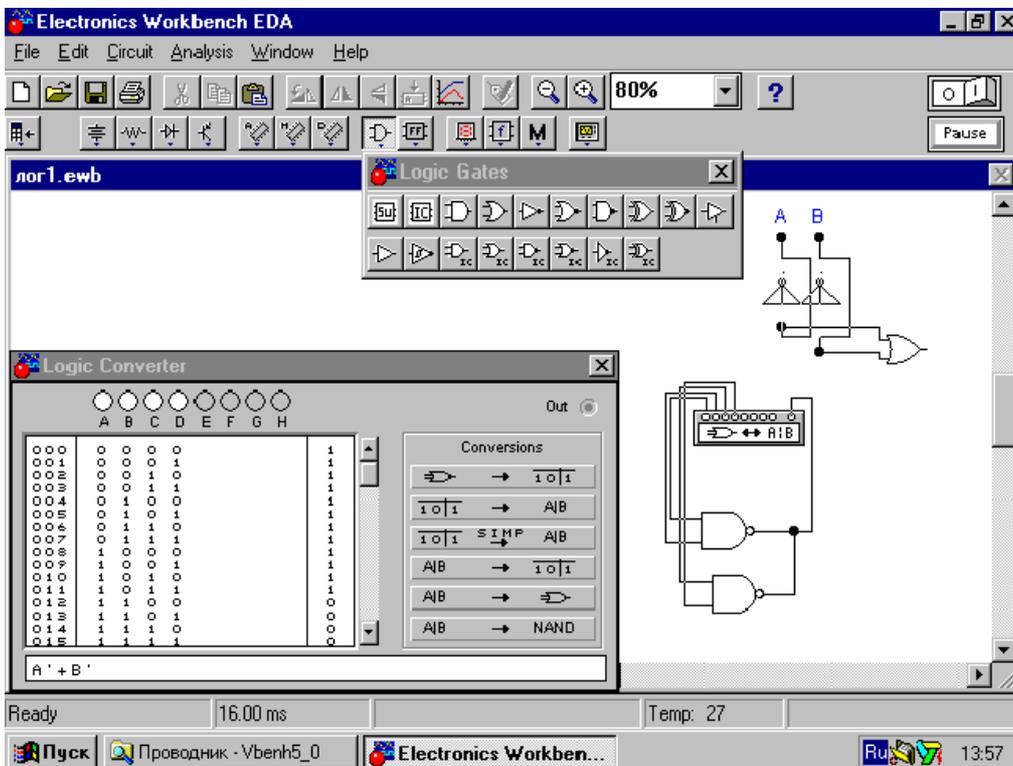
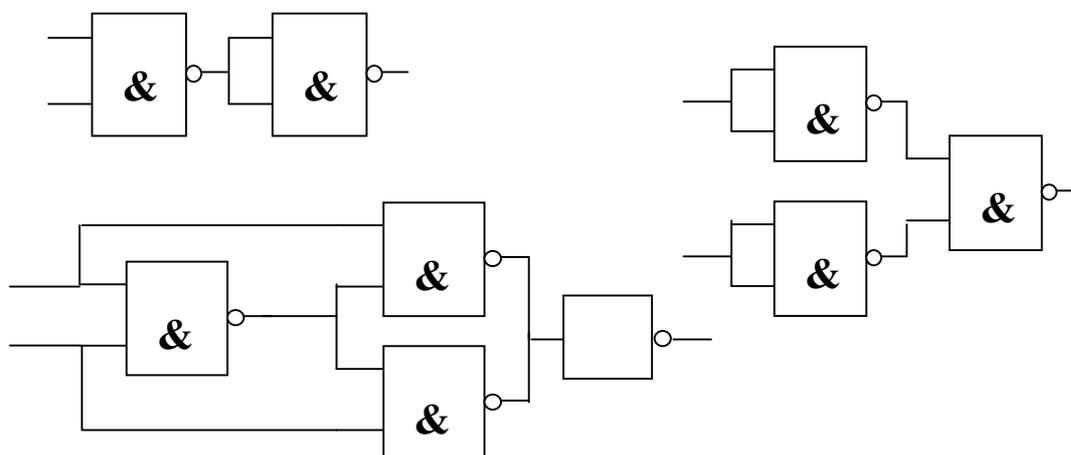


Рис. 106

Контрольные вопросы и задания.

1. Объяснить назначение и принцип работы логического конвертора. Решить следующие задания с использованием логического конвертора.

2. Исследуйте следующие функциональные схемы.



3. Исследуйте логическую схему и постройте функциональную логическую схему:

a) $B * C' + A * C$.

b) $A * B' * C + A * B' * C' + A' * B' * C$.

c) $A * (B + C) * (D + C)$.

4. Проведите синтез логического устройства с выходной комбинацией:

a) 00100111.

b) 01101001.

c) 0110100110010110.

7.6.4. Цифровой компаратор

Цель работы: Изучение назначения устройства и принцип работы цифрового компаратора.

Оборудование: Электронная лаборатория Electronics Workbench.

Краткая теория

Цифровые компараторы (от англ. *compare* – сравнивать) выполняют сравнение двух чисел – A , B – одинаковой разрядности,

заданных в двоичном или двоично-десятичном коде. В зависимости от схемного исполнения компараторы могут определять равенство $A=B$ или неравенства $A<B$, $A>B$. Результат сравнения отображается в виде логического сигнала на одноименных выходах в случае выполнения условия на выходе 1.

Цифровые компараторы применяются для выявления нужного числа (слова) в цифровых последовательностях для выполнения условных переходов.

Схемы одноразрядных компараторов приведены на рис. 107.

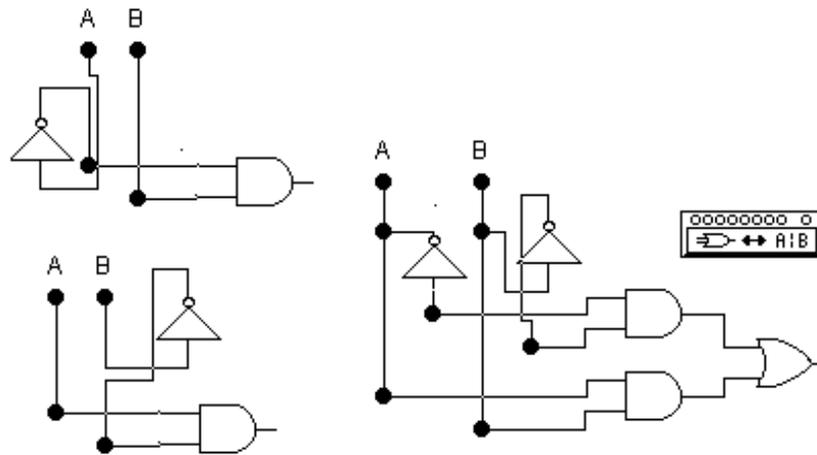


Рис. 107

Операциям сравнения ($A<B$, $A=B$, $A>B$) соответствуют структурные формулы ($A' * B$, $A' * B' + A * B$, $A * B'$).

Контрольные вопросы и задания.

1. Какие функции выполняет цифровой компаратор, в каких устройствах он может быть использован?
2. Подсоединив схемы к логическому конвектору, исследуйте приведенные схемы.
3. Составьте схему устройства, объединяющую все три компаратора.
4. Составьте схемы устройств, удовлетворяющие условиям: $A \leq B$, $A \neq B$, $A \geq B$.
5. Исследуйте составленные схемы устройств.
6. Составьте структурные формулы и таблицы истинности для приведенных выше логических схем цифровых компараторов.

7.6.5. Устройство контроля четности

Цель работы: Изучение назначения и принципа работы устройства контроля четности.

Оборудование: Электронная лаборатория Electronics Workbench.

Краткая теория

Операция контроля четности двоичных чисел позволяет повысить надежность передачи и обработки информации. Ее сущность заключается в суммировании по модулю 2 всех разрядов с целью выяснения четности числа, что позволяет выявить наиболее вероятную ошибку в одном из разрядов двоичной последовательности. Например, если при передаче кода 1001 произойдет сбой во втором разряде, то на приемном пункте получим код 1101 – такую ошибку определить в общем случае затруднительно. Если же код относится к двоично-десятичному (способ кодирования десятичных чисел, при котором каждая цифра представляется четырьмя двоичными разрядами – двоичной тетрадой), обнаружение ошибок путем введения дополнительного бита четности происходит следующим образом. На передающей стороне передаваемый код анализируется и дополняется контрольным битом до четного или нечетного числа единиц в суммарном коде. Соответственно суммарный код называется четным или нечетным. В случае нечетного кода дополнительный бит формируется таким образом, чтобы сумма всех единиц в передаваемом коде, включая контрольный бит, была нечетной. При контроле четности все наоборот. Например, в числе 0111 число единиц нечетно. Поэтому при контроле нечетности дополнительный код должен быть нулем, а при контроле четности – единицей. На практике чаще всего используется контроль нечетности, поскольку он позволяет фиксировать полное пропадание информации (случай нулевого кода во всех информационных разрядах). На приемной стороне производится проверка кода четности. Если он правильный, то прием разрешается, в противном случае включается сигнализация ошибки или посылается передатчику запрос на повторную передачу.

Схема формирования бита четности для четырехразрядного кода приведена на рис.108. Она содержит четыре элемента, исключающие ИЛИ, выполняющие функции сумматоров по модулю 2 (без переноса), и состоит из трех ступеней. На первой ступени попарно суммируются все биты исходного кода на входах А, В, С, D. На второй ступени анализируются сигналы первой ступени, и устанавливается четность или нечетность суммы входного кода. На третьей ступени полученный результат сравнивается с контрольным сигналом на входе E, задающим вид используемого контроля, в результате чего на выходе F формируется дополнительный пятый бит четности, сопровождающий информационный сигнал в канале передачи.

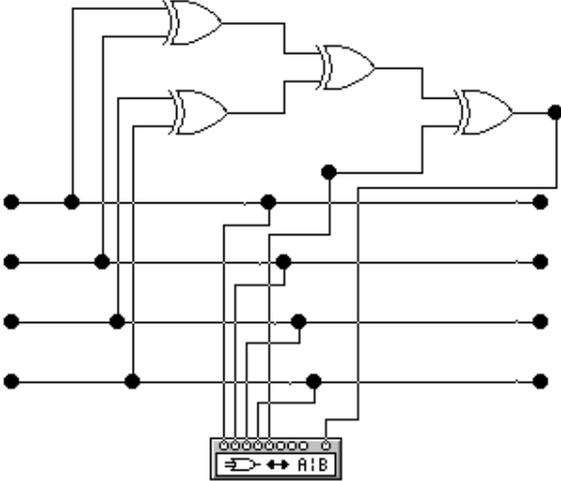


Рис. 108

Результаты моделирования приведены на рис. 109.

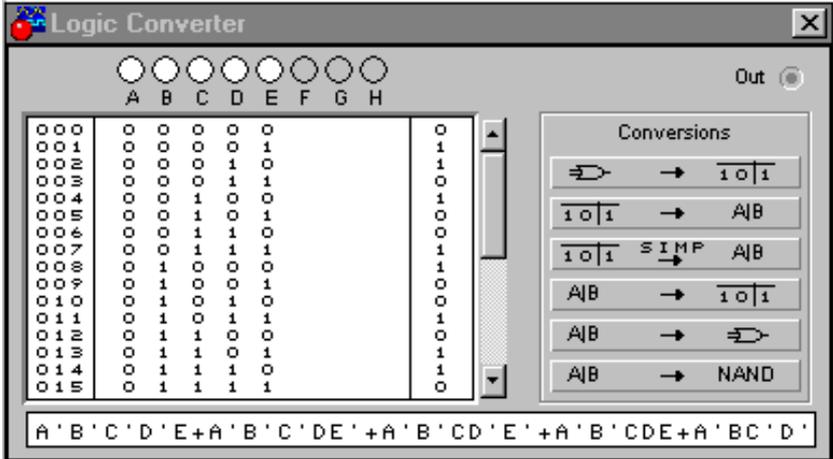


Рис. 109

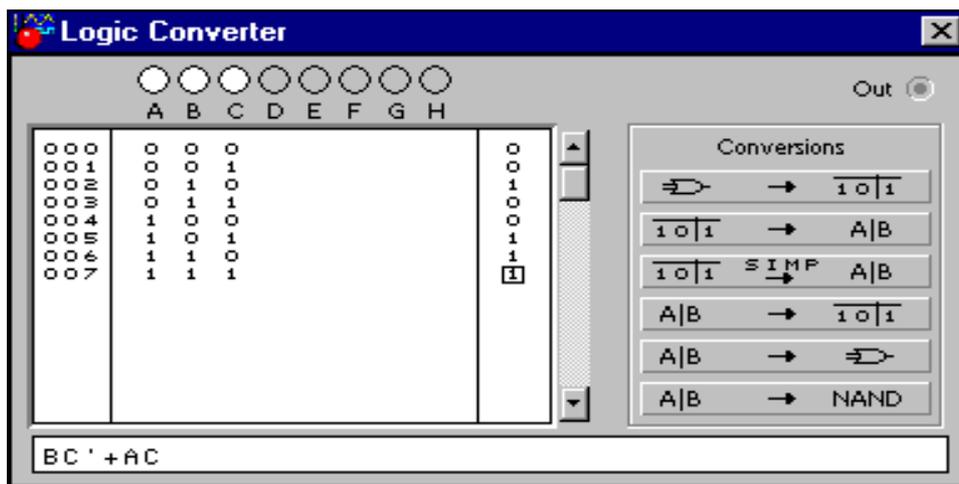


Рис. 111

Результаты моделирования двухканального мультиплексора с помощью логического конвертора показаны на рис. 111, из которого видно, что его выходной сигнал описывается структурной формулой $B \cdot C' + A \cdot C$, т.е. сигнал из канала A проходит на выход при адресном входе $C=1$, а из канала B – при $C=0$, что и соответствует логике работы мультиплексора.

Демультимплексоры в функциональном отношении противоположны мультиплексорам. С их помощью сигналы с одного информационного входа распределяются в требуемой последовательности по нескольким выходам. Выбор нужной выходной шины, как и в мультиплексоре, обеспечивается установкой соответствующего кода на адресных входах. При m адресных входах демультимплексор может иметь до 2^m выходов.

Принцип работы демультимплексора поясним с помощью схемы.

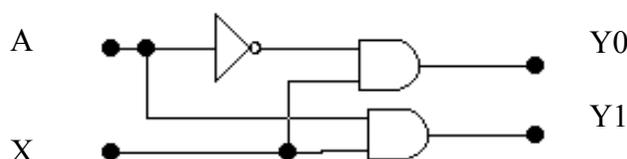


Схема содержит два элемента И и один элемент НЕ. На схеме: X – информационный вход, A – вход адреса, Y0, Y1 – выходы.

Если $A=0$ сигнал информационного входа передается на выход Y0, а при $A=1$ – на выход Y1.

Контрольные вопросы и задания.

1. Что такое мультиплексор, каково его назначение?
2. Что такое демультиплексор, для решения каких задач его можно применить?
3. Придумайте схему трехканального мультиплексора?
4. Придумайте схему трехвыходного демультиплексора?

7.6.7. Арифметические сумматоры

Цель работы: Изучение назначения и принципа работы устройств полусумматора и сумматора. Знакомство с базовыми элементами полусумматора и полного сумматора из библиотеки EWB.

Оборудование: Электронная лаборатория Electronics Workbench.

Краткая теория

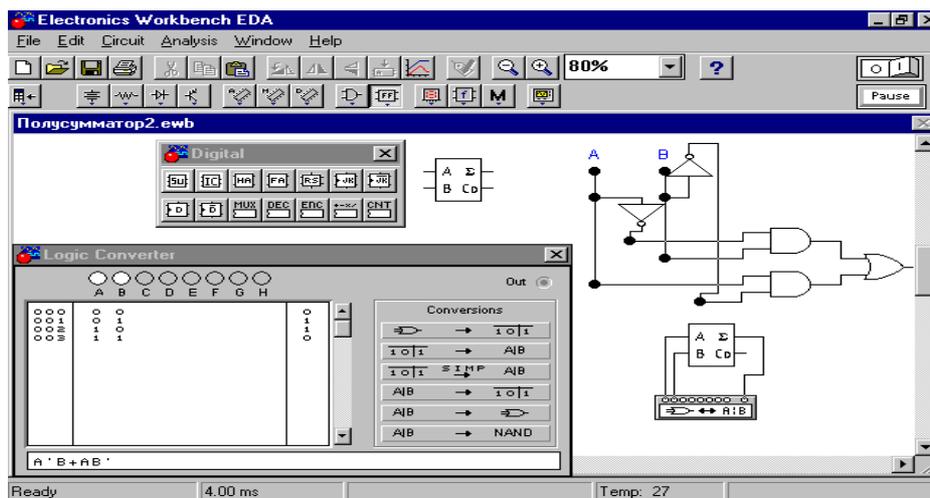


Рис.112

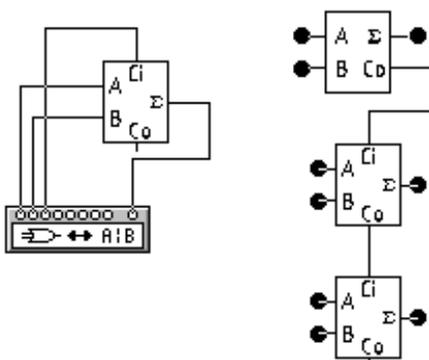


Рис. 113

Арифметические сумматоры являются составной частью так называемых арифметико-логических устройств (АЛУ) микропроцессоров. В программе EWB арифметические сумматоры представлены в библиотеке Digital двумя базовыми устройствами: полусумматорами и полными сумматорами. Они имеют следующие

назначения выводов: A , B – входы слагаемых, Σ – результат суммирования, C_0 – выход переноса, C_i – вход переноса. N -разрядный сумматор создается на базе одного полусумматора и $n-1$ – на базе полных сумматоров. На рис.113 приведено исследование полусумматора.

На рис.113 приведена схема подключения полного сумматора к логическому конвертору и трехразрядный сумматор.

Контрольные вопросы и задания.

1. Чем отличается полусумматор от полного сумматора?
2. Выясните внутреннюю структуру полного сумматора, пользуясь схемой подключения к логическому конвертору, аналогично приведенному анализу полусумматора.
3. Исследуйте выходы переноса полусумматоров и полного сумматора.
4. Исследуйте приведенный трехразрядный сумматор, последовательно подключая выходы к логическому конвертору.

7.6.8. Триггеры

Цель работы: Изучение назначения и принцип работы устройств триггера. Знакомство с базовыми устройствами триггер из библиотеки EWB.

Оборудование: Электронная лаборатория Electronics Workbench.

Краткая теория

Триггеры – устройства, имеющие два устойчивых состояния. Под действием управляющих сигналов они переходят из одного состояния в другое, и после снятия сигналов хранят это состояние до тех пор, пока не отключено напряжение питания. Таким образом, триггер является ячейкой памяти для одного двоичного разряда, т. е. бита информации.

В библиотеке EWB триггеры представлены тремя типами: RS, JK и D, показанных на рис. 114.

Назначение выводов триггеров следующее. Для всех триггеров выходы Q – прямой, Q' – инверсный (обратный). Для RS-триггера: R – установка триггера в 0, при сигнале 1 на этом входе

$Q=0$, $Q'=1$; S – установка в 1, при сигнале 1 на этом входе $Q=1$, $Q'=0$; комбинация $R=1$, $S=1$ не изменяет состояние выходов и относится к запрещенным. Для JK триггера: J, K – информационные входы, \triangleright – тактовый вход; вывод сверху – асинхронная предустановка триггера в единичное состояние ($Q=1$) вне зависимости от состояния сигналов на входах (функционально аналогичен входу S RS триггера); вывод внизу – асинхронная предустановка триггера в нулевое состояние (так называемая очистка триггера, после которой $Q'=1$); наличие кружочков на изображениях выводов обозначает, что активными являются сигналы низкого уровня, а для тактового входа – что переключение триггера производится не по переднему фронту тактового импульса, а по его срезу (так чаще всего называют задний фронт импульса). Для D-триггера вход D – информационный, состояние этого входа после подачи тактового импульса запоминается триггером, т. е. при $D=1$ имеем $Q=1$, при $D=0$ – $Q=0$.

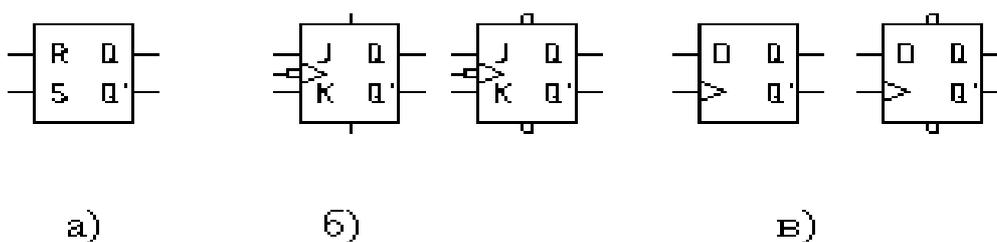


Рис. 114

Схема асинхронного RS-триггера на логических схемах приведена на рис. 115.

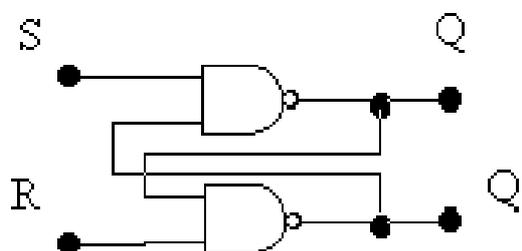


Рис. 115

Для понимания процессов, происходящих в триггерах, приведем схему (рис.116) синхронного одноклапного RS-триггера на логических элементах И-НЕ.

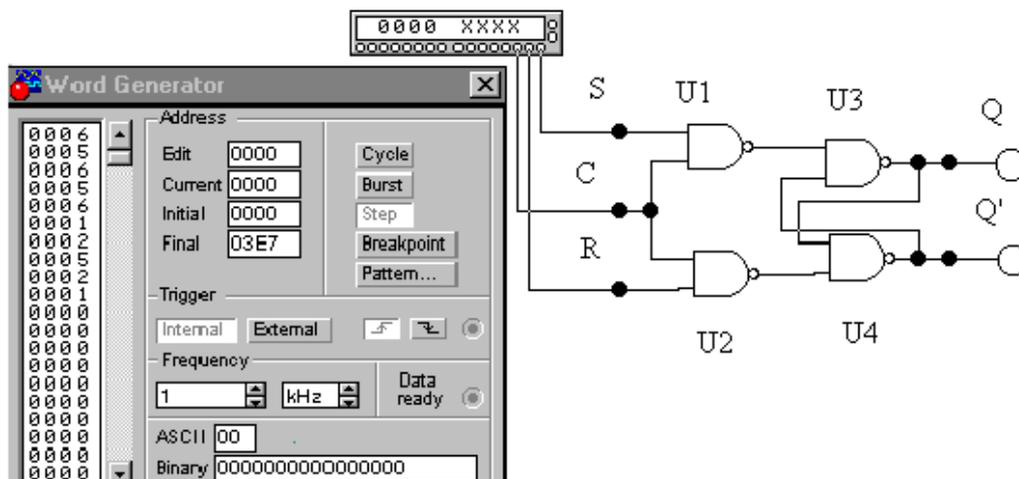


Рис. 116

Триггер имеет входы установки в 0 (R-вход, сигнал на инверсном выходе $Q'=1$) и 1 (S-вход, сигнал на прямом выходе $Q=1$). Установка триггера в 0 или 1 производится только при наличии сигнала синхронизации $C=1$. Возможные комбинации входных сигналов, имитирующие работу триггера в различных режимах, показаны на лицевой панели генератора слова.

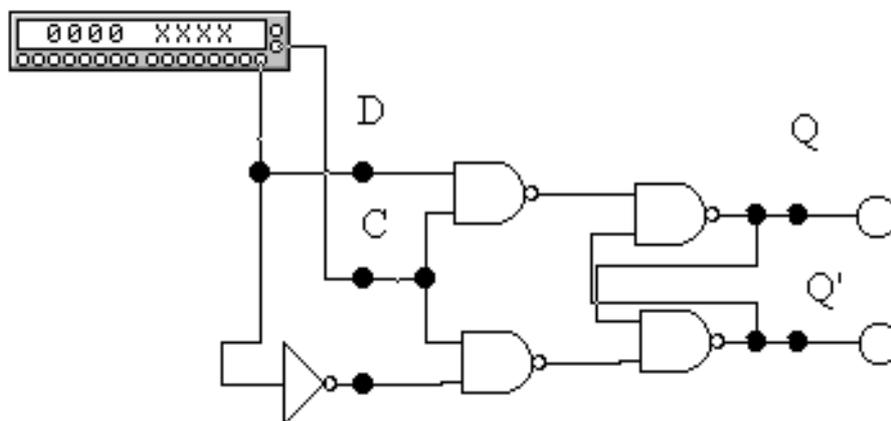


Рис. 117

Если схему триггера дополнить инвертором, то получим схему D-триггера (рис. 117), в котором состояние выхода определяется сигналом на D-входе: при $D=1$ – $Q=1$, при $D=0$ – $Q'=1$. В качестве тактового сигнала используется выход синхросигнала генератора слова.

Если в D-триггере D-вход соединить с инверсным выходом Q' , то получится T-триггер с одним тактовым C-входом.

Контрольные вопросы и задания.

1. Что такое триггер, какого типа они бывают ?
2. Проведите исследования указанных выше схем триггеров.

7.6.9. Счетчик

Цель работы: Изучение назначения и функции устройства счетчик. Знакомство с принципом работы устройства счетчик.

Оборудование: Электронная лаборатория Electronics Workbench.

Краткая теория

Счетчиком называют устройство, сигналы на выходе которого отображают число импульсов, поступивших на счетный выход. Триггер может служить примером простейшего счетчика. Такой счетчик считает до двух. Счетчик, образованный цепочкой из m триггеров, может подсчитать в двоичном коде 2^m импульсов. Каждый из триггеров такой цепочки называют разрядом счетчика. Число m определяет количество разрядов двоичного числа, которое может быть записано в счетчик. Число $K_{сч}=2^m$ называют коэффициентом (модулем) счета.

Информация снимается с прямых и (или) инверсных выходов всех триггеров. В паузах между входными импульсами триггеры сохраняют свои состояния, т. е. счетчик запоминает число входных импульсов.

Нулевое состояние всех триггеров принимается за нулевое состояние счетчика в целом. Остальные состояния складываются по числу поступивших входных импульсов. Когда число входных импульсов $N_{вх} > K_{сч}$, происходит переполнение, после чего счетчик возвращается в нулевое состояние и цикл повторяется. Коэффициент счета, таким образом, характеризует число входных импульсов, необходимое для одного цикла и возвращения в исходное состояние.

Счетчики различаются числом и типами триггеров, способами связей между ними, кодом, организацией счета и другими показателями. Цифровые счетчики классифицируются по следующим параметрам:

- Коэффициент счета – двоичные; двоично-десятичные или с другим основанием счета; с произвольным постоянным и переменным (программируемым) коэффициентом счета;
- Направление счета – суммирующие, вычитающие и реверсивные;
- Способ организации внутренних связей – с последовательным, параллельным или комбинированным переносом, кольцевые.

Классификационные признаки независимы и могут встречаться в различных сочетаниях: например, суммирующие счетчики бывают как с последовательным, так и с параллельным переносом, могут иметь двоичный, десятичный и иной коэффициент счета.

Схема четырехразрядного двоичного счетчика с последовательным переносом на D-триггерах приведена на рис.118.

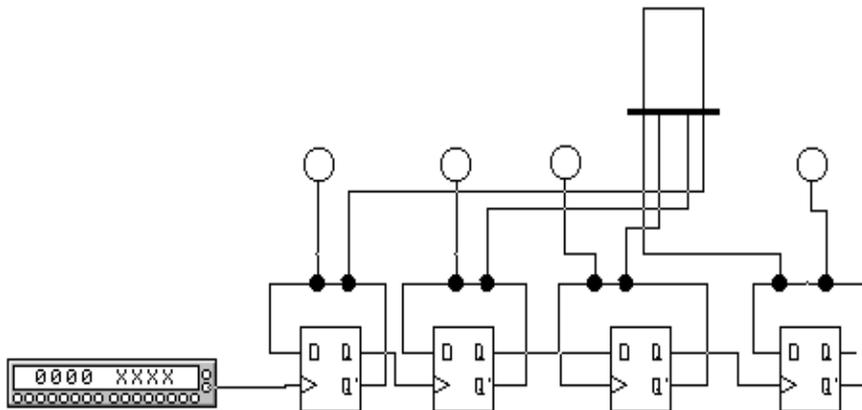


Рис. 118

На вход счетчика подаются импульсы с выхода синхросигналов генератора слова, которые генерируются при каждом нажатии клавиши STEP. Каждый триггер счетчика осуществляет деление на 2, сигнал переноса передается последовательно от одного разряда к другому. Состояние разрядов счетчиков в двоичном коде индицируется логическим пробником (индикатором), а в десятичном – семисегментным индикатором.

Контрольные вопросы и задания.

1. Что такое счетчик, какие функции он может выполнять?
2. Назовите типы счетчиков и их возможные применения.
3. Смоделируйте приведенную выше схему и проанализируйте работу счетчика.

7.6.10. Регистр

Цель работы: Изучение назначения и функций регистра. Знакомство с принципом работы регистров.

Оборудование: Электронная лаборатория Electronics Workbench.

Краткая теория

Регистры – устройства для временного хранения и преобразования информации в виде многоразрядных двоичных чисел. Регистры наряду со счетчиками и запоминающими устройствами являются наиболее распространенными устройствами цифровой техники. При сравнительной простоте регистры обладают большими функциональными возможностями. Они используются в качестве управляющих и запоминающих устройств, генераторов и преобразователей кодов, счетчиков, делителей частоты, узлов временной задержки. Элементами структуры регистров являются триггеры D- или JK-типа с динамическим или статическим управлением. Одиночный триггер может запоминать (регистировать) один разряд (бит) двоичной информации. Такой триггер можно считать одноразрядным регистром. Занесение информации в регистр называют операцией ввода или записи. Выдача информации к внешним устройствам характеризует операцию вывода или считывания. Запись информации в регистр не требует его предварительного обнуления.

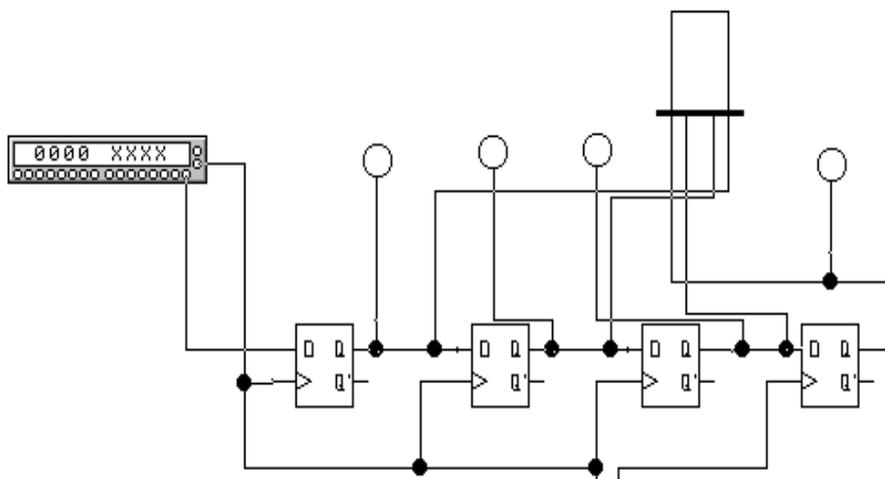


Рис. 119

Все регистры в зависимости от функциональных свойств подразделяются на две категории – накопительные (регистры памяти, хранения) и сдвигающие. В свою очередь, сдвигающие регистры делятся по способу ввода и вывода информации на параллельные и последовательно-параллельные и комбинированные, по направлению передачи (сдвига) информации – на однонаправленные и реверсивные.

На рис. 119 показана схема простейшего четырехразрядного регистра на D-триггерах, в котором информация заносится последовательно, начиная с младшего разряда.

Контрольные вопросы и задания.

1. Что такое регистр, какие функции он может выполнять?
2. Назовите типы регистров и их возможные применения.
3. Смодулируйте приведенную выше схему и проанализируйте работу регистра.

7.6.11. Оперативное запоминающее устройство

Цель: Изучение назначения и функций оперативного запоминающего устройства. Знакомство с принципом работы оперативного запоминающего устройства.

Оборудование: Электронная лаборатория Electronics Workbench.

Краткая теория

Оперативные запоминающие устройства (ОЗУ) являются неотъемлемой частью микропроцессорных систем различного назначения. ОЗУ делятся на два класса: статические и динамические. В статических ОЗУ запоминание информации производится на триггерах, а в динамических – на конденсаторах емкостью 0,5 пФ. Длительность хранения информации в статических ОЗУ не ограничена, тогда как в динамических ОЗУ она ограничена временем саморазряда конденсатора, что требует специальных средств регенерации и дополнительных затрат времени на этот процесс.

На рис.120 показана ячейка статического ОЗУ на D-триггере и вспомогательных логических элементах. Информационный вход

ячейки подключен к шине данных D1 одного из разрядов, ее выход – к соответствующей шине D0 через элемент с тремя состояниями U6. Ячейка выбирается сигналами Y=1, X=1, поступающими с дешифратора адреса. При записи в ячейку памяти на D1 устанавливается 1 или 0, на входе WR/RD' – сигнал 1, в результате чего срабатывают элементы 2И, U1, U2. Положительный перепад сигнала с элемента U2 поступает на тактовый вход D-триггера U4, и в нем записывается 1 или 0 в зависимости от уровня сигнала на его D-входе. При чтении на входе WR/RD' устанавливается 0, при этом срабатывают элементы U1, U3, U5 и на вход РАЗРЕШЕНИЕ ВЫХОДА буферного элемента U6 поступает разрешающий сигнал, в результате чего сигнал с Q-выхода D-триггера передается на разрядную шину D0, состояние которой индицируется логическим пробником IND.

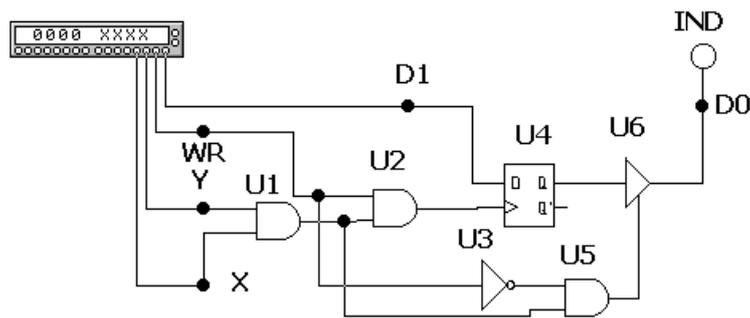


Рис. 120

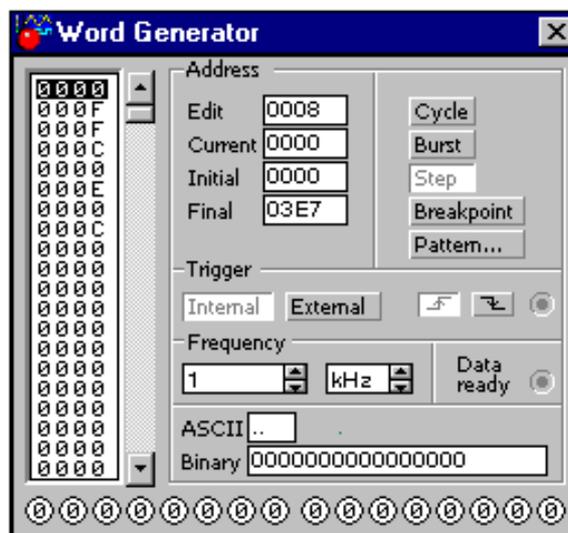


Рис. 121

Заметим, что запоминающие устройства статического типа отличаются высоким быстродействием и в компьютерах используются в качестве так называемой кэш-памяти.

Контрольные вопросы и задания.

1. Какие типы памяти существуют?
2. Чем отличается динамическая память от статической?
3. Смоделируйте и проанализируйте работу ОЗУ, схема которого приведена выше.

ГЛАВА 8. ПРИМЕНЕНИЕ ИНФОРМАЦИОННЫХ СИСТЕМ В ОБУЧЕНИИ

8.1. ОБРАБОТКА ИНФОРМАЦИИ В СИСТЕМЕ MICROCAL ORIGIN 6.0

Microcal Origin является приложением системы Windows. Пользовательский интерфейс данного программного продукта не отличается от других приложений. Этот графический пакет очень удобен для обработки результатов при проведении как реального эксперимента, так и вычислительного эксперимента. Он обладает многими возможностями:

- построения двухмерных графиков в декартовой и полярной системе координат;
- построения множества графиков в одной координатной плоскости;
- построения множества графиков в нескольких координатных плоскостях;
- построения различных видов диаграмм;
- построения трехмерных графиков;
- вращения трехмерных графиков относительно любой из осей;
- регрессии и интерполяции графиков;
- статистической обработки результатов эксперимента и графиков;
- экспорта и импорта данных и графиков в другие приложения и графические пакеты;
- и другими возможностями;

Рассмотрим некоторые возможности и технологию работы в этом приложении. Для работы в этом пакете мы должны иметь экспериментальные данные в виде текстового файла с расширением *.dat. Подготовка такого файла, например, на языке Паскаль осуществляется следующим образом:

```

FIL:TEXT;           {описание текстового файла}
  ASSIGN(FIL,'C:\FILTR.DAT'); {открытие и указание пути
                               сохранения файла}
  REWRITE(FIL);      {обнуление файла}

```

Расчет

```

FOR I:=1 TO N DO BEGIN
WRITELN(FIL,I*N, ' ', U[I]);END; {формирование файла }
CLOSE(FIL);                       {закрытие файла}

```

Построение графика осуществляется следующим образом. В меню File находится подменю Import, в котором имеется подменю ASCII. Нажатие указателя мыши по нему открывает файлы папки Origin. Здесь осуществляется поиск необходимого нам файла ASCII. Далее в окне Origin появляются значения в виде таблицы (рис. 122). Построение графиков проводится с помощью меню Plot с различными возможностями выбора вида графиков.

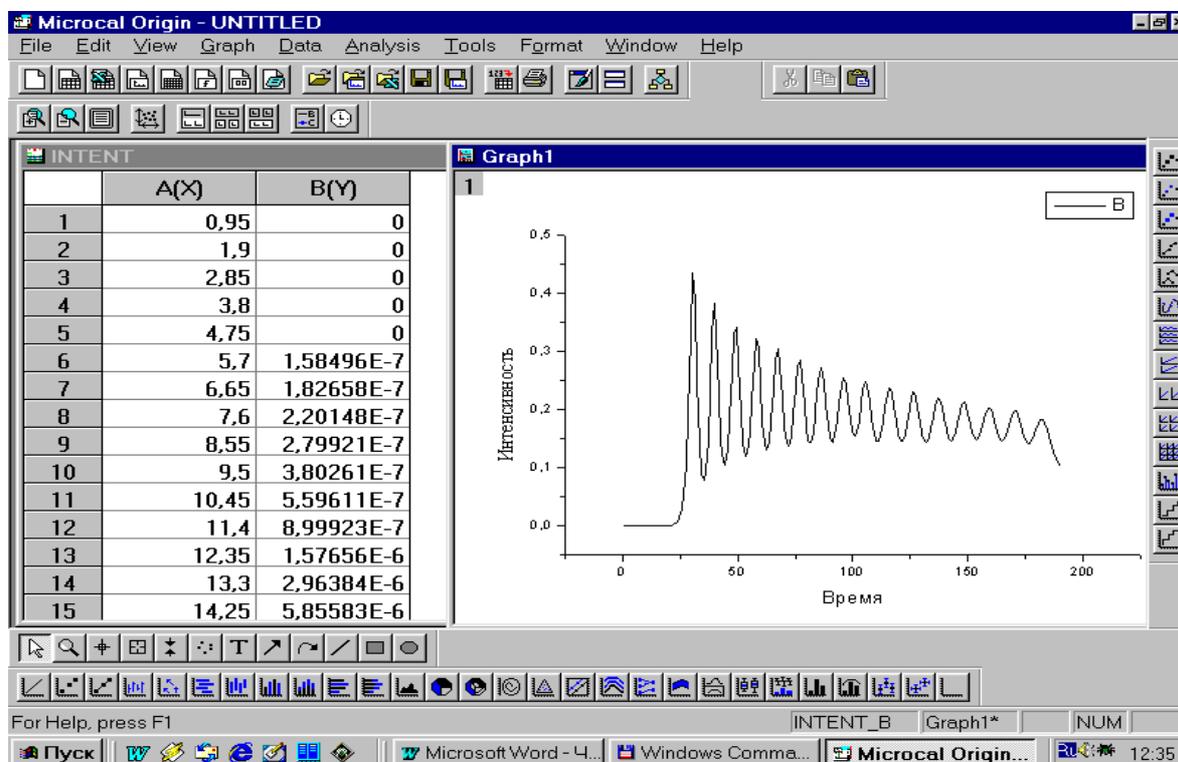


Рис. 122. Построение двумерных графиков

Для построения трехмерных графиков необходимо сформировать ASCII-файл в табличной форме, затем в меню Edit конвертировать его в матрицу с помощью подменю Direct. Матрица обычно закрашена в желтый цвет. Используя меню Plot 3D, можно построить трехмерный график с выбором вида графика.

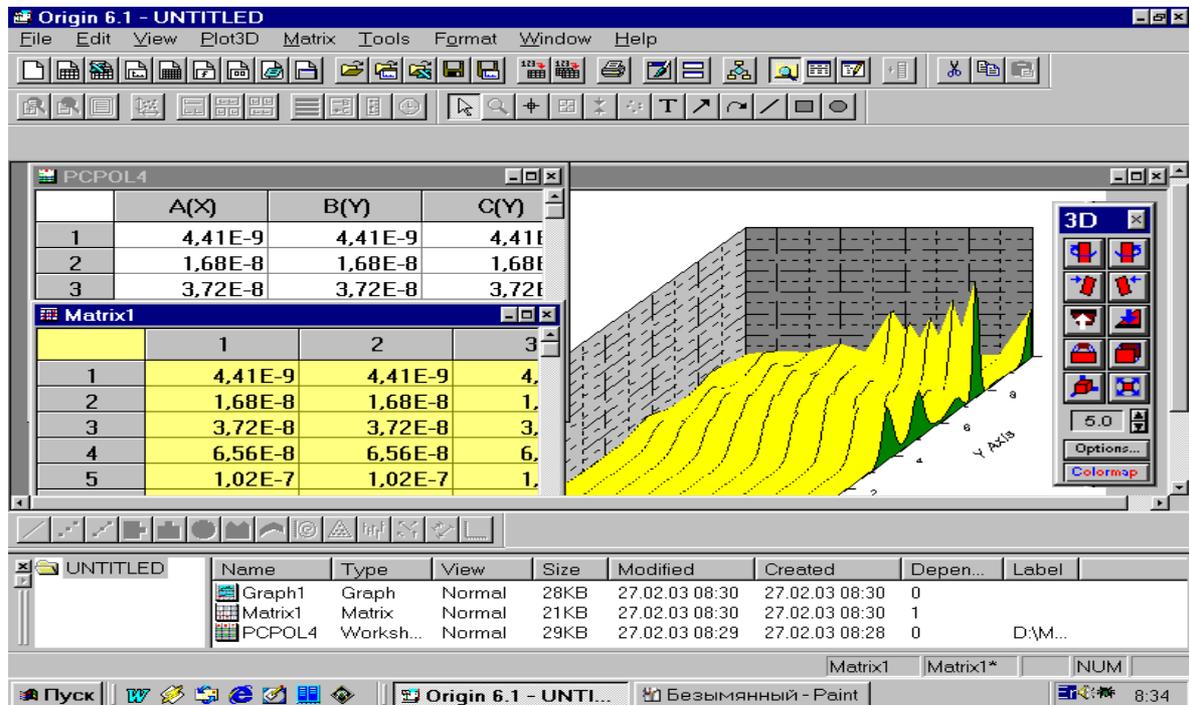


Рис. 123. Построение трехмерных графиков

Меню Matrix позволяет инвертировать, транспонировать и интерполировать полученную матрицу. С помощью меню 3D (рис. 123) трехмерный график можно вращать относительно осей координат, наклонять. Режим Options в меню 3D позволяет изменять параметры и расцветку трехмерного графика.

Изменение значений и параметров координатных осей, цвета графиков производится в подменю, которое открывается быстрым двойным щелчком мыши по графику или его параметрам.

8.2. СИСТЕМА ТЕХНИЧЕСКОГО КОНСТРУИРОВАНИЯ «КОМПАС»

Система КОМПАС-3D LT предназначена для выполнения учебных проектно-конструкторских работ в различных отраслях деятельности. Она может успешно использоваться студентами

машиностроительных, приборостроительных, архитектурных, строительных вузов и техникумов при выполнении домашних заданий, курсовых и дипломных работ (рис.124).

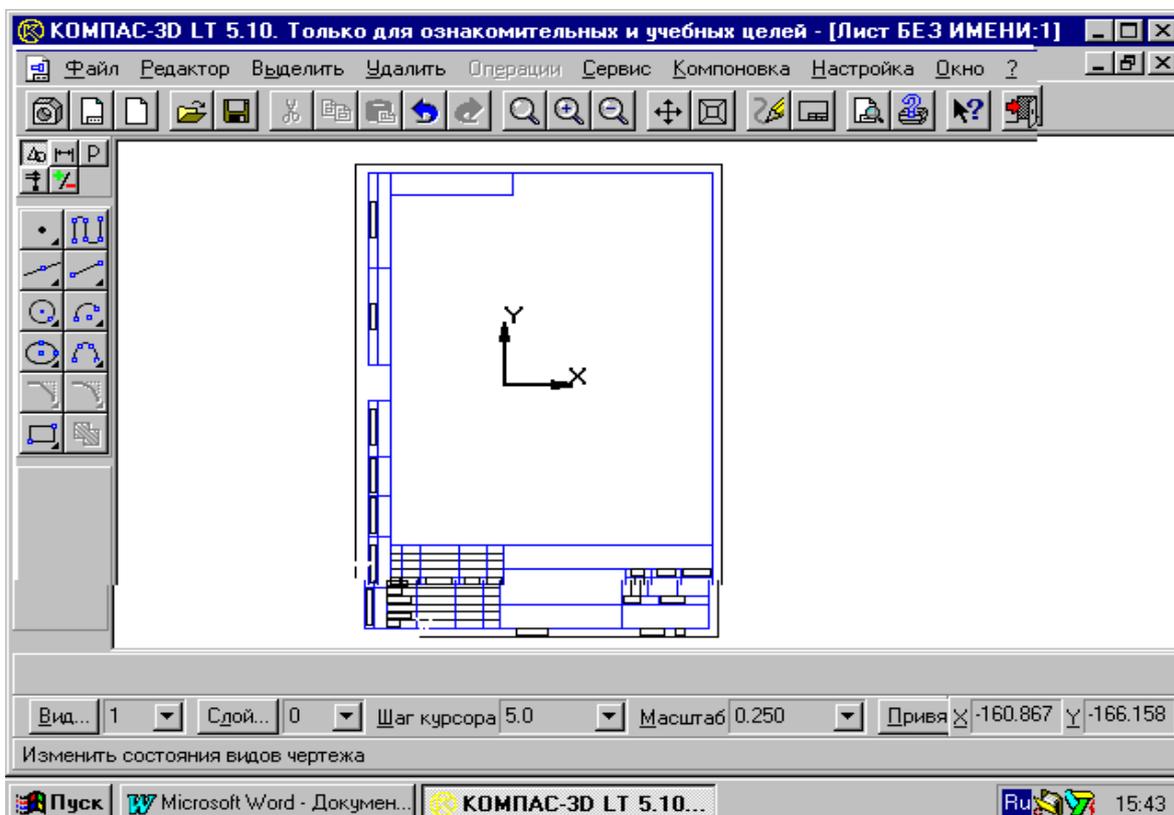


Рис. 124

Система КОМПАС-3D LT разработана специально для операционной среды MS Windows и в полной мере использует все ее возможности и преимущества для предоставления пользователю максимального комфорта и удобства в работе.

При возникновении затруднительных ситуаций во время работы с КОМПАС-3D LT вы можете быстро получить необходимую справочную информацию. Для этого разработана справочная система, которая содержит сведения о командах меню и панелях кнопок, клавиатурных комбинациях, типовых последовательностях выполнения различных операций и т.д.

Получить справочную информацию можно одним из следующих способов:

- Вызовите подходящую команду из меню *Справка*.
- Нажмите клавишу <F1> для получения подсказки по текущему действию.

- Нажмите кнопку *Объектная справка* на *Панели управления* для получения подсказки по объектам рабочего экрана.

Кнопка *Объектная справка*.

Курсор мыши изменит свой внешний вид (превратится в вопросительный знак со стрелкой). Выберите затем нужный объект экрана (например, команду меню или кнопку) и зафиксируйте на ней курсор (щелкните левой кнопкой мыши). Вы можете также быстро получить краткую информацию о какой-либо кнопке с помощью технологии всплывающих подсказок.

Интерфейс системы. По сравнению с традиционными Windows-приложениями в КОМПАС-3D LT наложены ограничения на одновременную работу с несколькими документами. Таким образом, в главном окне системы может быть открыт только один документ: чертеж, фрагмент или деталь. Команды вызываются из страниц *Главного меню*, контекстного меню или при помощи соответствующих кнопок.

Параметры при выполнении команд могут вводиться в полях *Строки параметров* или в специальных диалогах. Во многих командах создания модели детали требуется не только ввести параметры в диалоге, но и указать в окне работы с деталью какие-либо объекты. В этом случае указывать объекты в окне нужно, не закрывая диалог. Для того чтобы активизировать диалог после работы в окне детали, щелкните на нем мышью.

В *Строке сообщений* (если ее показ не отключен при настройке системы) отображаются подсказки по текущему действию или описание выбранной команды.

Типы объектов КОМПАС. В КОМПАС-3D LT поддерживаются следующие графические объекты (рис.125).

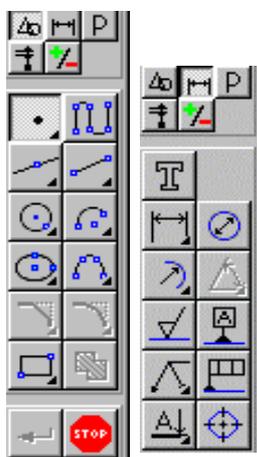


Рис. 125

Геометрические объекты: точка, прямая, отрезок прямой, окружность, дуга окружности, эллипс, многоугольник, ломаная линия, кривая Безье, NURBS-кривая, штриховка, эквидистантная кривая, макроэлемент.

Размеры: линейный, угловой, радиальный, размер диаметральный, размер высоты.

Специальные и технологические обозначения: многострочная текстовая надпись, обозначение базы, допуск формы и расположения, символ шероховатости, линия-выноска, обозначение марки-

ровки, обозначение клеймения, стрелка направления взгляда, линия разреза или сечения, обозначение центра.

Объекты оформления чертежа: технические требования, основная надпись (штамп), обозначение шероховатости неуказанных поверхностей.

Типы документов КОМПАС. Начиная с версии 5.10 в системе КОМПАС-3D LT возможно создание трехмерных моделей деталей (расширение файла *.m3d). Основным графическим документом в системе КОМПАС-3D LT является лист чертежа.

Чертеж хранится в отдельном файле специального двоичного формата (расширение файла *.cdw). Если Ваша конструкторская документация (например, сборочный чертеж) состоит из нескольких листов, то они создаются и обрабатываются отдельно (в различных файлах).

Каждый чертеж состоит из видов, технических требований, основной надписи (штампа чертежа) и обозначения шероховатости неуказанных поверхностей детали (знака неуказанной шероховатости).

Другим типом графического документа КОМПАС-3D LT является фрагмент (расширение файла *.frw). Он отличается от чертежа отсутствием объектов оформления. Во фрагменте нет рамки, основной надписи, знака неуказанной шероховатости и технических требований. Фрагмент, как и вид чертежа, может содержать до 255 слоев. Фрагмент идеально подходит для хранения изображений, которые не нужно оформлять как лист чертежа (эскизные прорисовки, разработки и т.д.). Кроме того, во фрагментах удобно сохранять созданные типовые решения и конструкции для последующего использования в других документах.

Создание, открытие и сохранение документов. Используя КОМПАС-3D LT, вы можете работать с тремя типами документов – чертежами, фрагментами и моделями деталей.

Каждый документ хранится в отдельном файле на диске и при необходимости загружается в систему для редактирования, вывода на бумагу, использования в качестве прототипа и т.д.

Работа с файлами в КОМПАС-3D LT практически ничем не отличается от подобной работы в других приложениях Windows. Поэтому вы можете с успехом использовать все приемы открытия и сохранения файлов, уже знакомые по другим системам.

Создание нового документа. Каждый вновь созданный документ отображается на экране в новом окне, а открытый ранее документ при создании нового документа закрывается. При этом пользователь может сохранить сделанные в этом документе изменения или отказаться от сохранения. Для создания новой детали выберите из меню *Файл* команду *Создать деталь* или нажмите кнопку *Новая деталь* на *Панели управления*.

Кнопка *Новая деталь*.

Для создания нового листа чертежа выберите из меню *Файл* команду *Создать лист* или нажмите кнопку *Новый лист* на *Панели управления*.

Кнопка *Новый лист*.

Для создания нового фрагмента выберите из меню *Файл* команду *Создать фрагмент* или нажмите кнопку *Новый фрагмент* на *Панели управления*.

Кнопка *Новый фрагмент*.

Если перед вызовом команды создания документа в системе уже был открыт какой-либо документ, в него вносились изменения и эти изменения не были сохранены, на экране появится запрос на сохранение изменений.

При создании новых документов используются установленные параметры по умолчанию (например, для чертежа – это формат листа, стиль оформления, стили текстовых надписей в различных объектах). Если впоследствии Вам будет нужно изменить параметры документа, используйте команду *Параметры текущего документа...* из меню *Настройка*.

Типы файлов. В КОМПАС-3D LT используются по умолчанию следующие расширения файлов. Файлы документов: *.cdw – файлы чертежей; *.frw – файлы фрагментов; *.m3d – файлы деталей.

Служебные и вспомогательные файлы: *.tol – файлы предельных отклонений (допусков); *.bss – файлы библиотек специальных знаков; *.cfg – файлы конфигурации, содержащие сведения о настройках рабочей среды системы; *.prj – файлы проектов, содержащие сведения о настройках новых документов; *.dsk – файлы конфигурации, содержащие сведения о настройках рабочей области; *.acs – файлы соответствия обозначений спецзнаков в КОМПАС 5 и AutoCAD.



Единицы измерения. В КОМПАС-3D LT используется метрическая система мер (рис. 126). Расстояния между точками на чертежах и фрагментах вычисляются и отображаются в миллиметрах. При этом пользователь всегда работает с реальными размерами (в масштабе 1:1), а последующее размещение изображения на формате листа чертежа выполняется с помощью выбора подходящего масштаба вида.

Рис. 126 Таким образом, в отличие от вычерчивания на кульмане, при работе с КОМПАС-3D LT нет необходимости заботиться о пересчете реальных координат в зависимости от размеров изделия и формата листа.

При измерении площадей, расстояний, массо-центровочных характеристик фигур и моделей пользователь может управлять представлением результатов, назначая нужные единицы измерений (миллиметры, сантиметры, дециметры или метры).

Системы координат. При работе в КОМПАС-3D LT используются декартовы правые системы координат. Начало абсолютной системы координат чертежа всегда находится в левой нижней точке габаритной рамки формата.

При создании нового вида пользователь может задать положение системы координат этого вида. Начало системы координат фрагмента и начало системы координат детали не имеет такой четкой привязки, как в случае чертежа. Поэтому, когда создается новый фрагмент или новая деталь, точка начала системы координат автоматически отображается в центре окна.

Для удобства работы пользователь может создавать в документе произвольное количество локальных систем координат (ЛСК) и оперативно переключаться между ними. Подробно об этом рассказано в теме *Использование локальных систем координат*.

Управление курсором. Курсор – это главный инструмент при работе с КОМПАС-3D LT. С помощью курсора пользователь вызывает команды, вычерчивает и редактирует различные объекты, указывает точки и выполняет множество других действий.

Основной способ управления курсором – это его перемещение мышью. Вы можете также передвигать курсор, используя клавиши со стрелками на основной или расширенной клавиатуре. В этом случае перемещение будет не произвольным, как в случае

использования мыши, а дискретным в соответствии с текущим шагом курсора. Для изменения шага курсора используйте специальное поле в *Строке текущего состояния*.

В зависимости от того, какое действие выполняется в системе, изменяется внешний вид курсора (стрелка, перекрестие, вопросительный знак со стрелкой и т.д.). Форма и размер курсора могут быть настроены пользователем в соответствующем диалоге (*Настройка системы – Графический редактор – Курсор*).

Стили чертежных объектов. Для того чтобы сделать информацию, содержащуюся в чертежно-графических документах, удобной для чтения и понимания, применяются различные толщины и цвета линий, типы штриховок, символов и так далее.

Набор свойств объекта, влияющих на его отображение, называется стилем этого объекта. Например, стиль точки включает в себя внешний вид символа, которым рисуется точка, а также цвет.

Разновидности стилей

В КОМПАС-3D LT предусмотрено назначение стилей для трех групп чертежных объектов: линий (кривых), штриховок, точек. Кроме того, можно использовать стили текстов при вводе отдельных надписей на чертежах.

В составе КОМПАС-3D LT поставляются готовые системные стили точек, линий, штриховок и текстов. Они хранятся непосредственно в коде программы.

Назначения стиля при вводе объекта

После того, как вызвана команда создания чертежного объекта (например, команда вычерчивания отрезка), на экране отображается *Строка параметров объектов*. В ней находятся не только поля геометрических параметров, но и поле управления стилем.

Поле стиля линии

По умолчанию система назначает новому объекту текущий стиль, который был установлен при выполнении предыдущих команд. Внешний вид текущего стиля отображается в поле стиля в *Строке параметров*.

Если нужно изменить стиль создаваемого объекта, щелкните левой кнопкой мыши на поле стиля (не обязательно точно на самой кнопке *Стиль*). На экране появится диалог, в котором следует

выбрать нужный стиль из списка системных стилей, либо из внешней библиотеки, либо из сформированного пользователем набора.

В процессе ввода объекта вы можете неоднократно изменять его стиль, используя поле управления стилем.

Изменение стиля объекта

Если вам понадобилось изменить стиль объекта (например, толщину и цвет отрезка), дважды щелкните мышью на этом объекте для запуска его редактирования.

Дальнейшая последовательность действий точно такая же, как и при создании объекта. Для изменения стиля используется поле управления стилем. Оно находится в Строке параметров, которая появляется на экране при переходе к редактированию объекта.

Привязка. Механизм, позволяющий точно задать положение курсора, выбрав условие его позиционирования (например, в узлах сетки, или в ближайшей характерной точке, или на пересечении объектов и т.д.).

Меню локальных привязок

Это меню выводится на экран при нажатии правой кнопки мыши во время выполнения различных команд (создание объектов, редактирование, выделение и т.д.). С помощью команд меню можно выполнить привязку курсора к объектам различными способами: *Ближайшая точка, Пересечение, Середина, Центр, По сетке, Угловая привязка, Выравнивание, Точка на кривой, По Y на кривую, Против Y на кривую, По X на кривую, Против X на кривую.*

Команды меню привязок аналогичны командам меню геометрического калькулятора, отображаемым при вводе значений координат точек.

Для вызова нужного способа привязки выберите его название из меню.

Глобальные привязки

Когда в процессе создания или редактирования объектов вы используете меню привязок или клавиатурные комбинации для того, чтобы точно установить курсор в нужную точку, вы применяете так называемую локальную привязку (рис. 127). Однако после того как был выбран вариант локальной привязки из меню,

система не запоминает, какой именно это был вариант. Когда вам потребуется сделать точно такую же привязку к другой точке, придется вызвать меню и снова выбрать нужную команду. Это неудобно в том случае, если требуется выполнить несколько однотипных привязок подряд.

В отличие от локальной, глобальная привязка (если она установлена) всегда действует по умолчанию при выполнении операций ввода и редактирования. Например, если выбран вариант глобальной привязки к пересечениям, то при вводе точки система автоматически будет выполнять поиск ближайшего пересечения в пределах ловушки курсора. В том случае, если пересечение будет найдено, точка будет зафиксирована именно в этом месте.

Можно включать несколько различных глобальных привязок к объектам, и все они будут работать одновременно. При этом расчет точки выполняется «на лету», на экране отображается фантом, соответствующий этой точке, и текст с именем действующей в данный момент привязки. Цвет отображения фантома и текста соответствует цвету, установленному для увеличенного курсора.

Кнопка для вызова диалога настройки глобальных привязок расположена в *Строке текущего состояния*. Для того, чтобы установить нужную комбинацию глобальных привязок, включите или выключите нужные пункты в диалоге.

Вы можете также отключить действие всех глобальных привязок, а затем включить их вновь в прежнем составе, воспользовавшись кнопкой *Запретить/разрешить* действие глобальных привязок в *Строке текущего состояния*.

Кнопка *Запретить/разрешить* действие глобальных привязок.

Эта кнопка служит индикатором действия глобальных привязок: нажатая кнопка означает, что глобальные привязки отключены, отжатая – выключены.

Для того чтобы разрешить действие отключенных ранее глобальных привязок, отожмите кнопку *Запретить/разрешить* действие глобальных привязок.

Для переключения кнопки при помощи клавиатуры воспользуйтесь комбинацией клавиш <Ctrl>+<d>.

Для включения/выключения глобальных привязок служит также опция *Запретить привязки* диалога настройки глобальных привязок.

Локальная привязка является более приоритетной, чем глобальная, то есть при вызове какой-либо команды локальной привязки она подавляет установленную глобальную на время своего действия (до ввода точки или отказа).

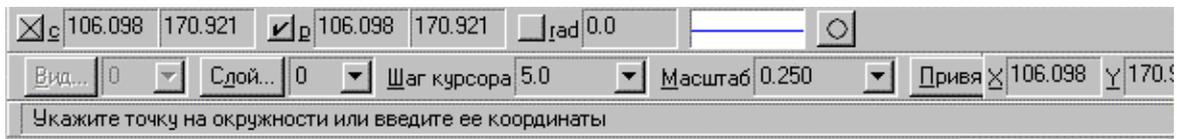


Рис. 127

Геометрический калькулятор. Когда вы создаете или редактируете какой-либо объект, в Строке параметров объектов отображаются поля для ввода значений параметров этого объекта.

Вы можете ввести данные об элементе вручную, явно набрав их в соответствующих полях. Однако КОМПАС-3D LT предоставляет и другой способ ввода – непосредственное снятие значений параметров с чертежа. Для подобного снятия параметров используется так называемый геометрический калькулятор.

Геометрический калькулятор – это механизм получения количественной информации о параметрах и взаимном расположении объектов с целью использования ее при построении других объектов. Например, при помощи геометрического калькулятора можно построить окружность с радиусом, равным длине какого-либо объекта, отрезок с углом наклона, равным углу между другими отрезками, и т.д.

Рассмотрим ситуацию, когда вычерчивается отрезок. В Строке параметров открыты поля ввода значений координат точек отрезка, его угла наклона и длины. Если щелкнуть правой кнопкой мыши на любом из полей, на экране появляется меню команд геометрического калькулятора, причем набор команд зависит от типа параметра. Например, если геометрический калькулятор был вызван на поле ввода длины отрезка, то будут предложены именно команды снятия длин (расстояние между точками, длина элемента и т.п.). Для поля ввода угла будет соответственно выдано меню снятия угловых величин, а для полей координат – меню снятия значений координат (оно практически совпадает с меню привязок).

Поясним применение геометрического калькулятора на не-сложном примере вычерчивания отрезка, параллельного другому отрезку. Щелкните правой кнопкой мыши на поле угла наклона, выберите в появившемся меню команду Наклон прямой/отрезка и затем укажите курсором тот отрезок, параллельно которому нужно выполнить построение. Значение угла наклона относительно оси X текущей системы координат будет снято, занесено в поле Строки параметров и зафиксировано.

Таким образом, мощные средства геометрического калькулятора позволяют использовать параметры уже существующих объектов чертежа при построении или редактировании других объектов.

Подробную информацию о каждой команде геометрического калькулятора вы сможете найти в разделе *Меню* геометрического калькулятора.

8.3. СИСТЕМА АВТОМАТИЗИРОВАННОГО ПЕРЕВОДА PROMT 98

К средствам автоматизации перевода можно отнести два вида программ: электронные словари и программы перевода. Электронные словари представляют собой средства для перевода отдельных слов, отображаемых на дисплее или в имеющемся документе. Удобство их использования состоит в возможности немедленно получить перевод неизвестного слова без поиска его в отдельном толстом томе. Программы перевода получают на входе текст, выполненный на одном языке, и выдают текст на другом языке, то есть автоматизируют перевод текста. Программы автоматического перевода имеет смысл использовать для перевода технических текстов в следующих случаях:

- при абсолютном незнании иностранного языка;
- при необходимости получить перевод быстро, даже ценой снижения его качества;
- для перевода на иностранный язык;
- для быстрого создания первоначального черновика, используемого в ходе подготовки полноценного перевода.

Программа PROMT 98 позволяет автоматизировать технический перевод. Переводит документы с английского, немецкого, французского и обратно.

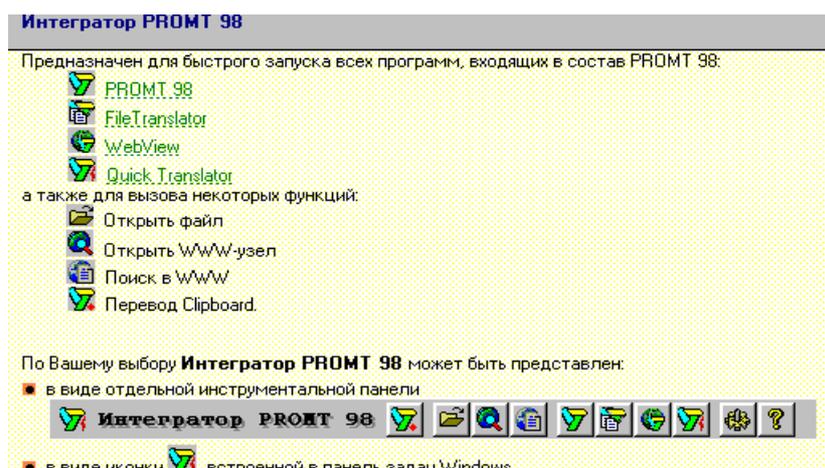


Рис. 128

Программа PROMT 98 запускается – Главное меню (Пуск – Программы – Главное меню – PROMT 98 – PROMT 98).

Для запуска можно воспользоваться Интегратором PROMT 98. (см. рис. 128). PROMT 98 – запускается основная программа автоматизированного перевода.

Внешний вид File Translator приведен на рис. 129.

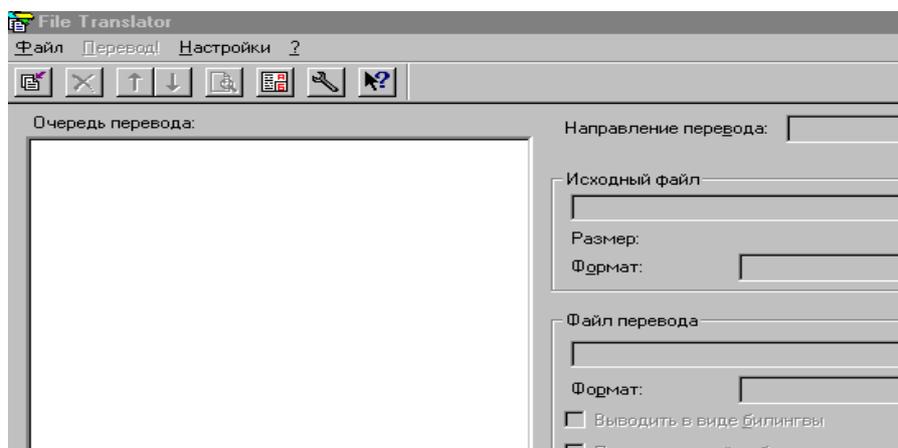


Рис. 129

File Translator – средство пакетного перевода файлов, предназначен для автоматического перевода файлов в фоновом режиме. В левой части окна располагается список файлов, ожидающих очереди перевода. В правой части окна располагаются элементы управления, позволяющие задать все настройки правил перевода, используемые в основной программе PROMT 98. Теряется лишь диалоговый характер работы. Внешний вид QTrans представлен на рис.130.

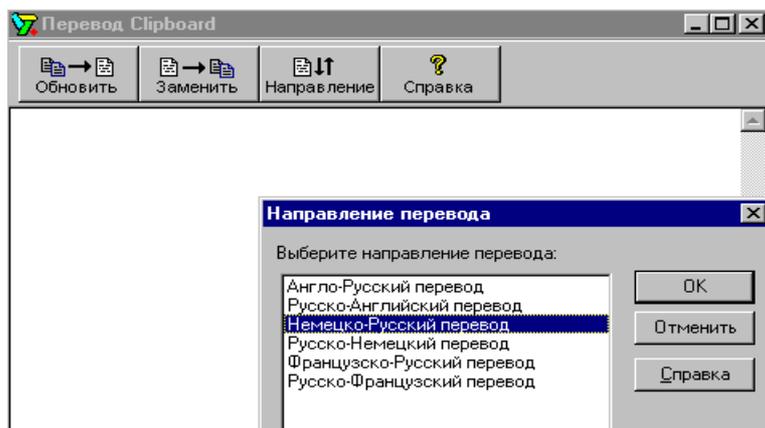


Рис. 130

QTrans предназначен для быстрого перевода неформатированного текста. Оно не содержит никаких средств открытия или сохранения документов, так как предполагается, что переводимый текст вводится на верхнюю панель окна программы вручную или переносит туда через буфер обмена. Чтобы сохранить перевод, его следует поместить в буфер обмена при помощи кнопки *Копировать перевод*, после чего произвести вставку в той программе, в которой этот текст будет использован.

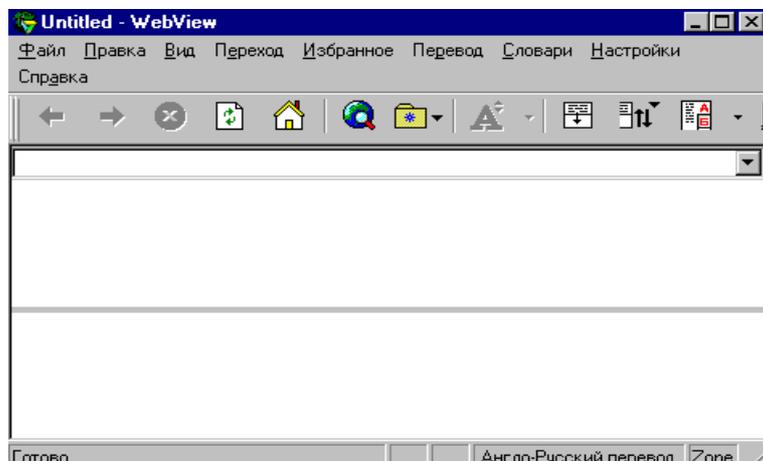


Рис. 131

WebView представляет собой полноценный браузер, эквивалентный по своим возможностям программе Internet Explorer (рис.131). Отличие от обычных браузеров состоит в том, что окно программы разбито на две части. В верхней части страница отображается в том виде, в каком она получена из Интернета. Одновременно с началом загрузки страницы в нижней части начинается формироваться ее перевод. При этом переводу подвергается

только текст. Адреса, гиперссылки, вставные объекты остаются без изменений. Переходы по гиперссылкам можно осуществлять как с верхней, так и с нижней части страницы.

Внешний вид основной программы PROMT 98 представлен на рис.132–133.

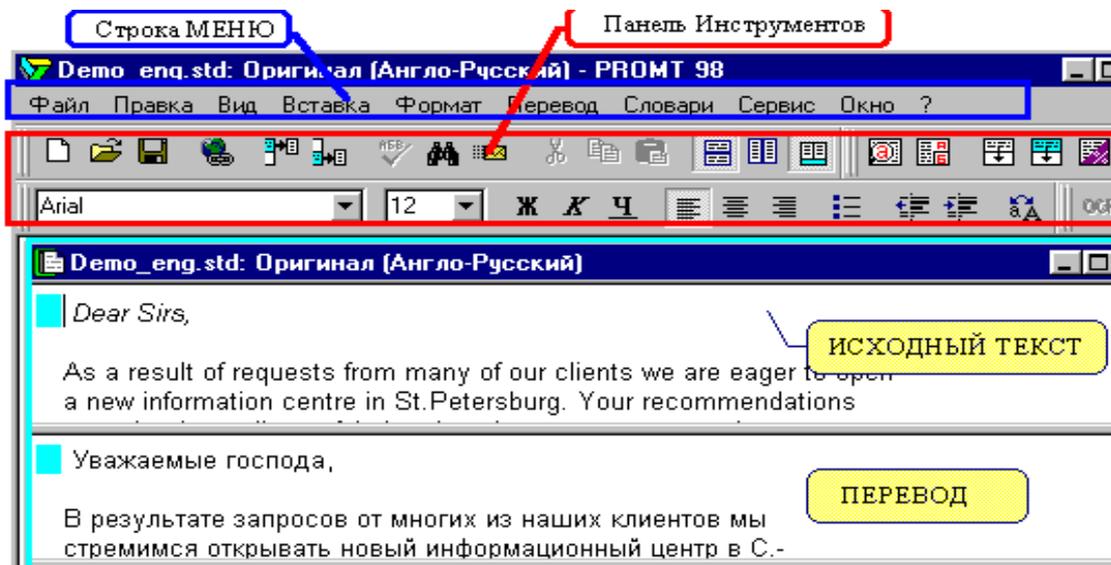


Рис. 132

Изображение панелей инструментов

Нажатие кнопки на инструментальной панели позволяет быстро выполнить некоторые команды меню. В PROMT существуют панели: «Основная», «Форматирование», «Перевод» и «Сервис».

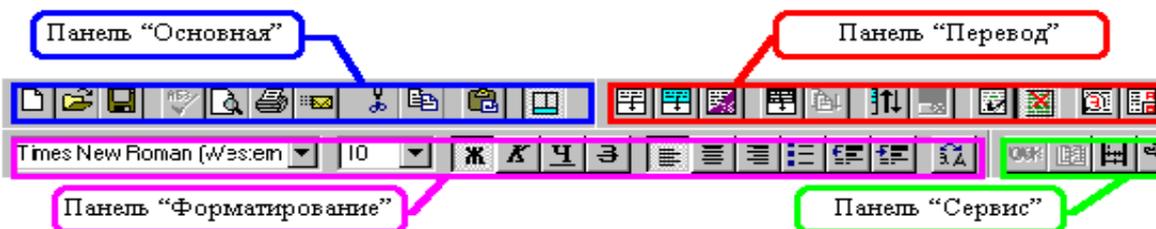


Рис. 133

Элементы работы показаны на рис. 134. Создание нового документа. Если вы хотите набрать текст непосредственно в системе PROMT с целью его дальнейшего перевода, выполните одно из следующих действий: нажмите кнопку создать документ, либо выберите команду *Создать* в меню *Файл*.

В окне «Направление перевода» укажите, с какого языка и на какой вы собираетесь переводить текст, и нажмите ОК. Вы можете потом в любой момент изменить направление перевода, выбрав команду *Изменить направление* в меню *Перевод*. Появится пустое

окно документа с мигающим курсором на первой позиции. Начните набирать текст. После того, как весь текст или его часть набрана, вы можете приступить к выполнению перевода.

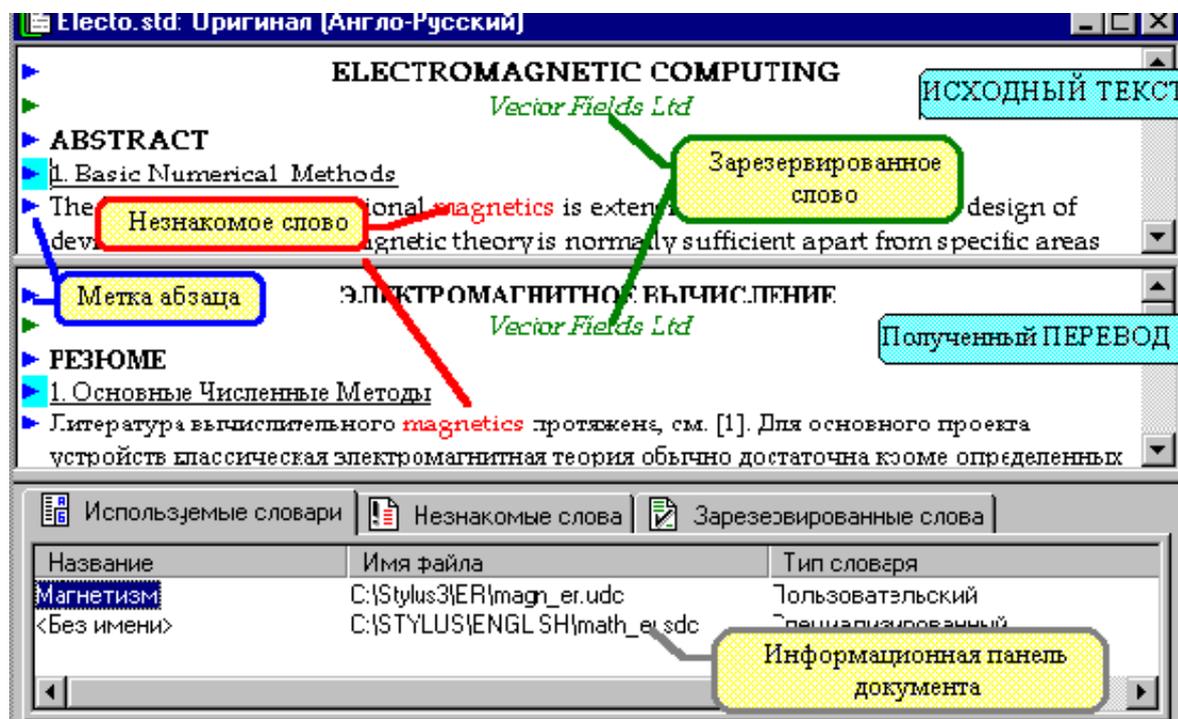


Рис. 134

Открытие существующего PROMT-документа. Если вы ранее переводили текст и сохранили результаты перевода в файле формата PROMT Документ (*.std), то вы можете открыть и продолжить далее работу над исходным текстом и его переводом.

Для открытия существующего документа:

1. Выполните одно из следующих действий:
 - нажмите кнопку,
 - либо выберите команду *Открыть* в меню *Файл*.
2. Появится диалоговое окно. Выберите путь к каталогу, где хранится документ, и имя файла.
3. Нажмите кнопку *Открыть*.

Открытие файла с исходным текстом. В системе PROMT Вы можете как переводить тексты, которые находятся в файлах, так и непосредственно набирать их и тут же переводить. Ниже описаны действия по выполнению перевода текста, набранного в каком-либо текстовом редакторе, а затем сохраненного в файле.

1. Откройте меню *Файл* и выберите команду *Открыть*.

Файл с текстом для перевода может быть подготовлен в следующих форматах: Только текст(*.txt), MS-DOS текст(*.txt), Word for Windows 6.0, 7.0(*.doc), Word for MS-DOS 3.x-5.x(*.doc), Форматированный текст(*.rtf), Windows Write 3.x(*.wri), Файлы гипертекста HTML.

2. В появившемся диалоговом окне *Открыть* в текстовом поле *Файл* наберите путь к файлу и его имя, например:

C:\DIRCT\rus.txt ,

затем нажмите кнопку *Открыть*.

3. Появится диалоговое окно *Конвертировать файл*, в котором программа выделит формат открываемого файла и направление перевода: с какого языка будет выполняться перевод.

Затем нажмите кнопку ОК.

Исходный текст появится в окне документа.

4. Документ готов к переводу.

Выберите команду *Весь текст* из меню *Перевод*.

Появится окно с индикацией процента выполненного перевода и времени, которое понадобится, чтобы перевести документ.

После выполнения перевода окно документа разделится на два подокна: в верхнем будет исходный текст, в нижнем – его перевод. Непереведенные слова будут выделены красным цветом.

Примечание. Причины нечитаемости исходного текста в окне документа могут быть следующие:

– при открытии документа неверно был указан формат открываемого файла в окне *Конвертировать файл*. Закройте документ и откройте файл заново, выбрав правильный формат.

– в вашей системе Windows нет шрифтов, которыми был набран исходный текст. В этом случае замените шрифты.

5. Для сохранения полученного текста перевода выберите команду *Сохранить – Перевод* из меню *Файл*.

6. В списке *Тип* выберите желаемый формат для сохранения результатов перевода: текстовый или RTF. Наберите путь и имя файла в поле *Файл*.

7. Нажмите кнопку *Сохранить*.

В результате вы получили файл с текстом перевода в нужном вам формате и можете использовать привычный редактор для его просмотра и редактирования.

Если вы хотите набрать текст непосредственно в системе PROMT и перевести его, выберите команду *Создать* из меню *Файл*. В окне Направление перевода выберите, с какого языка вы будете переводить, и нажмите ОК. Появится пустое окно документа с мигающим курсором на первой позиции. Начните набирать исходный текст.

Ввод текста в компьютер при помощи сканера. Если на вашем компьютере установлена система оптического распознавания текстов (OCR) FineReader, Cuniiform или AutoR, то Вы можете запустить ее непосредственно из системы PROMT. Система OCR прочтет напечатанный на бумаге текст при помощи сканера и передаст его в систему PROMT для перевода и редактирования. Чтобы ввести бумажный документ в компьютер при помощи сканера:

1. Выберите в меню *Сервис* команду *Подключение внешних приложений*. На вкладке «*Программы OCR*» укажите, какую систему OCR Вы будете использовать. Это достаточно выполнить один раз.

2. Вставьте документ в сканер.

3. Выберите команду *Запуск OCR* в меню *Сервис*. Будет запущена указанная вами программа.

4. Прочитайте текст при помощи сканера.

5. Выполните команду пересылки текста в систему PROMT, где будет открыт новый документ с этим текстом (можно выбрать команду пересылки, которая начнет перевод документа одновременно с его открытием).

Примечание: Приложение WINDOWS, предназначенное для оптического считывания информации, не входит в комплект поставки системы PROMT и должно быть приобретено отдельно.

Сохранение всего документа в формате PROMT. Если вы предполагаете в дальнейшем продолжить работу с документом в системе PROMT (например, вы выполнили перевод текста не до конца и хотите закончить его в другой раз), то необходимо сохранить документ в файле формата PROMT *Документ* (по умолчанию файлу присваивается расширение .std). Тогда будут сохранены все результаты вашей работы: исходный текст, полученный перевод, список незнакомых и зарезервированных слов.

Чтобы сохранить документ в формате PROMT:

- 1) Выберите команду *Сохранить Документ* в меню *Файл*.

2) Задайте путь и имя файла.

Сохранение документа в виде билингвы. Если вы хотите сохранить результаты перевода в следующем виде: абзац исходного текста, затем его перевод и т.д., то:

1) Выберите в меню *Файл* команду *Сохранить* – Билингву.

2) Выберите формат файла: *MS DOS Текст, Только текст, Форматированный текст(RTF), Word* для *Windows*.

Сохранение только текста перевода в файле. Если вы хотите сохранить в файле только текст перевода, выберите в меню *Файл* команду *Сохранить* – *Перевод*. Формат файла может быть *MS DOS Текст, Только текст, Форматированный текст(RTF), Word* для *Windows*.

Сохранение списка незнакомых слов.

Весь список слов, не найденных системой при переводе в подключенных словарях, вы можете увидеть в окне документа, активизировав вкладку «*Незнакомые слова*».

При желании вы можете сохранить список в текстовом файле.

Зачем это нужно? Например, вы можете открыть файл со списком в любом редакторе, распечатать его, затем найти переводы этих слов в «бумажных» словарях и ввести их в PROMT-словарь.

Чтобы сохранить список незнакомых слов:

1. Активизируйте вкладку «*Незнакомые слова*», щелкнув по названию левой кнопкой мыши.

2. Щелкните по вкладке правой кнопкой мыши, появится меню.

3. Выберите команду *Сохранить список*.

4. Укажите формат (Текстовый, RTF), путь и имя файла.

Горячие клавиши. Чтобы сделать вашу работу в PROMT более удобной, используйте горячие клавиши, их неполный список приведен ниже.

Обратите внимание, что большинство горячих клавиш указаны на соответствующих пунктах меню, но значения некоторых клавиш вы нигде больше не найдете.

Работа с документами

Чтобы создать новый документ, нажмите **Ctrl+N**.

Чтобы сохранить документ, нажмите **Ctrl+S**.

Чтобы открыть файл, нажмите **Ctrl+O**.

Чтобы распечатать документ на принтере или послать его по факсу, нажмите **Ctrl+P**.

Максимизация окна документа производится по нажатию клавиши F5.

Форматирование текста

Чтобы вызвать диалог «Шрифт» (команда *Шрифт* меню *Формат*), нажмите Ctrl+D.

Чтобы включить жирное начертание символов (Bold), нажмите Ctrl+B.

Чтобы включить курсивное начертание символов (Italic), нажмите Ctrl+I.

Чтобы включить режим подчеркивания символов (Underline), нажмите Ctrl+U.

Чтобы выровнять текст в абзаце по центру, нажмите Ctrl+E.

Чтобы выровнять текст в абзаце по левому краю, нажмите Ctrl+L.

Чтобы выровнять текст в абзаце по правому краю, нажмите Ctrl+R.

Команды редактора

Чтобы выделить весь документ, нажмите Ctrl+A.

Чтобы сменить регистр выделенного текста, нажмите Ctrl+Shift+C.

Чтобы вставить в документ текущие дату или время, нажмите Ctrl+Shift+T.

Чтобы вставить в документ любой (в том числе и нестандартный) символ, нажмите Ctrl+Shift+S.

Вы можете переходить из окна исходного текста в окно текста перевода и обратно, нажимая клавишу F6.

Чтобы перейти к редактированию исходного текста, нажмите Ctrl+F1.

Чтобы перейти к редактированию текста перевода, нажмите Ctrl+F2.

Чтобы перейти к работе с информационной панелью, нажмите Ctrl+F3.

Чтобы включить (или выключить) информационную панель, нажмите Ctrl+Shift+P.

Чтобы найти текст в документе, нажмите Ctrl+F.

Чтобы заменить в документе один текст на другой, нажмите Ctrl+H.

Повторный поиск текста в документе осуществляется по нажатию F3.

Чтобы отменить последнее действие редактора, нажмите Ctrl+Z или Alt+Back.

Работа с Clipboard

Чтобы скопировать текст в Clipboard, нажмите Ctrl+C или Ctrl+Ins.

Чтобы вставить в документ текст из Clipboard, нажмите Ctrl+V или Shift+Ins.

Чтобы вырезать текст из документа и поместить его в Clipboard, нажмите Ctrl+X или Shift+Del.

Перевод

Чтобы перевести следующий еще не переведенный параграф, нажмите F9.

Чтобы перевести текущий параграф, нажмите Alt+F9.

Чтобы перевести весь текст, нажмите Ctrl+F9.

Чтобы перевести выделенные параграфы, нажмите Shift+F9.

Чтобы оставить без перевода текущий параграф или несколько выделенных, нажмите Shift+F8.

Работа со словарями и зарезервированными словами

Чтобы ввести в словарь слово под курсором (или выделенное выражение), нажмите F8. Этого же можно добиться, дважды щелкнув мышкой по нужному слову при нажатой клавише Alt.

Чтобы отредактировать список используемых словарей, нажмите Ctrl+Shift+D.

Чтобы ввести в список зарезервированных слов слово под курсором (или выделенное выражение), нажмите Ctrl+F8. Этого же можно добиться, дважды щелкнув мышкой по нужному слову при нажатой клавише Ctrl.

Работа с другими программами

Чтобы вызвать внешний электронный справочный словарь, нажмите Ctrl+Shift+G. Чтобы проверить орфографию текста, нажмите F7.

Разное. Чтобы завершить работу с программой, нажмите Alt+X.

Контроль качества перевода

Качество перевода определяется полнотой используемых словарей и учетом грамматических правил. При переводе можно как

применять стандартные ресурсы, так и добавлять собственные. Программа использует при переводе три типа словарей: генеральный – содержит общеупотребительную лексику и бытовое значение слов; специализированные – содержат термины из различных областей знаний, причем значение термина выбирается в соответствии со специализацией словаря; пользовательский словарь формируется пользователем вручную. Применяются пользовательские словари (обычно в первую очередь), а также специализированные и генеральные, которые загружаются с самой системой.

8.4. ПОДГОТОВКА ПРЕЗЕНТАЦИЙ В СИСТЕМЕ POWERPOINT

Система Microsoft PowerPoint входит в пакет деловых приложений Microsoft Office. Презентация Microsoft PowerPoint – это набор слайдов, собранных в слайд-фильм. В создаваемых слайдах можно использовать текстовую, графическую, звуковую и видеоинформацию, готовые варианты дизайна.

Общий алгоритм работы в PowerPoint

1. Запуск программы.
2. Создание презентации.
3. Создание слайда.
4. Подготовка слайда. Пункты 3 и 4 повторяем до нужного количества слайдов. В случае необходимости слайды можно менять местами.
5. Просмотр презентации.
6. Сохранение презентации.

После запуска появляется рабочее окно (рис.135).

Создание презентации. Существуют следующие способы создания новой презентации. Во-первых, с помощью мастера автосодержания, предлагающего выбрать в качестве исходного материала презентацию с определенным типовым содержанием и оформлением. Мастер автосодержания предоставляет несколько образцов презентаций на различные темы; например, проведение совещания в организации или определение стратегического направления работы. Кроме того, имеются презентации, используемые в сети Интернет.

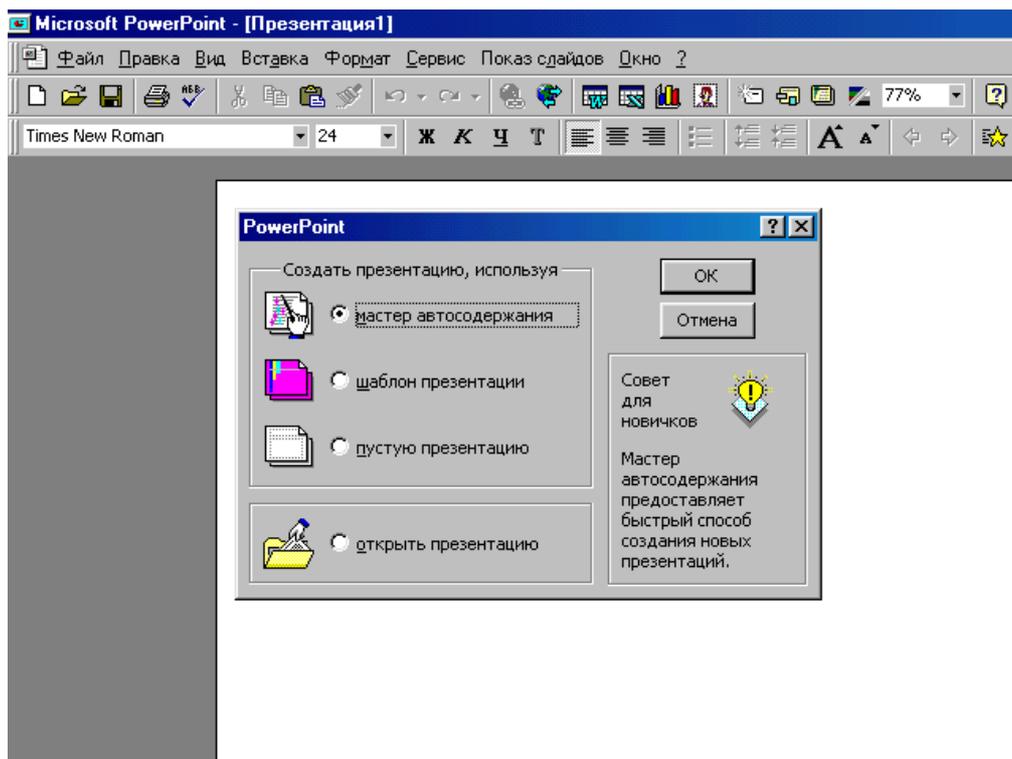


Рис. 135

Другой способ создания презентации состоит в выборе шаблона оформления, определяющего ее композицию, но не включающего содержание. Можно также начать со структуры, импортированной из другого приложения, такого, как Word, а также с пустой презентации, в которой не заданы ни оформление, ни содержание.

Создание презентации с помощью шаблона. Закрыв меню **Создать презентацию, используя шаблон**, Открываем **Файл – Создать – Новая презентация – Ок**, получаем стандартную разметку шаблона слайда (рис. 136).

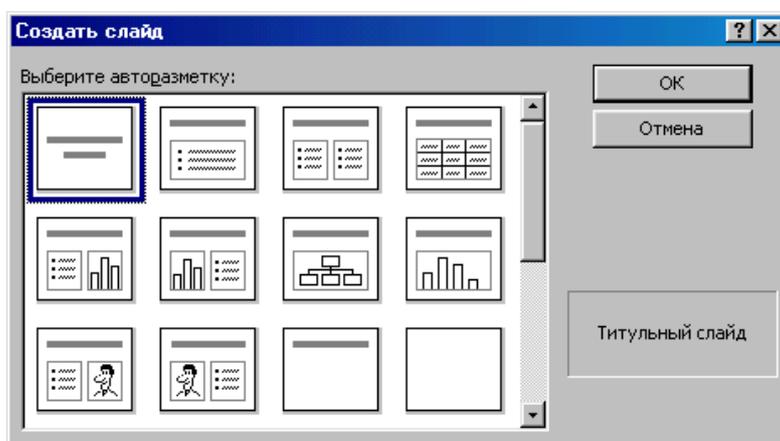


Рис. 136

Выбираем нужную форму слайда и активизируем щелчком мыши, и можно вводить необходимую информацию (рис. 137).

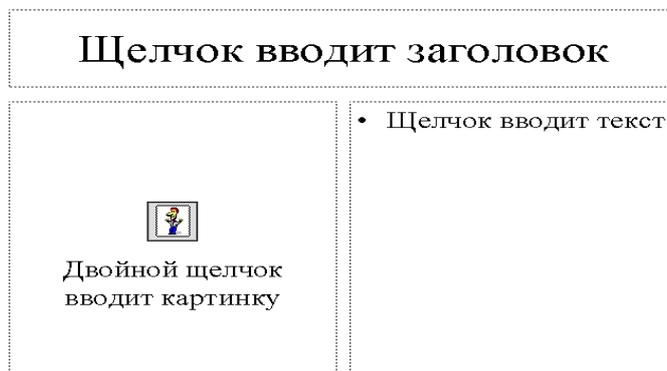


Рис. 137

Вставка текста. Как правило, самый простой способ добавления текста к слайду — ввести его непосредственно в местозаполнитель на слайде. Чтобы вставить текст вне местозаполнителя или фигуры (например, снабдить рисунки надписями или выносками), можно воспользоваться инструментом Надпись, расположенным на панели инструментов Рисование. Чтобы вставить текст без перехода на следующую строку (например, надпись), щелкните указанный инструмент, затем щелкните место, где разместится текст, и наберите текст. Чтобы вставить текст с переходом на другую строку, щелкните инструмент, перенесите его в точку начала текста и наберите текст.

Чтобы добавить текст в автофигуру, щелкните в ней и наберите текст. Этот текст закрепляется за фигурой, перемещается и вращается вместе с ней. Текст можно вставить в любую автофигуру, кроме линии произвольной формы и соединительной линии.

Способы вставки рисунков в презентацию. В комплект PowerPoint входит стандартный набор рисунков в виде коллекции. Эта коллекция включает множество картинок, выполненных на профессиональном уровне и позволяющих придать презентации более красочный вид. Выбор рисунков самый широкий — карты, изображения людей, зданий, пейзажей и т.д.

Для выбора рисунка нажмите кнопку **Вставить картинку** и перейдите на вкладку **Графика** или **Картинки**. В коллекции предусмотрено удобное средство поиска, помогающее найти нужные изображения для презентации. Кроме того, здесь имеется своя система справки, где можно узнать, как вставлять в коллекцию

свои рисунки, как обновлять ее и как настроить ее для своих целей. Чтобы воспользоваться средством поиска или системой справки, нажмите на вкладке **Графика** или **Картинки** кнопку **Поиск** или **Справка** соответственно.

В меню **Сервис** есть команда **Автографика**, просматривающая презентацию, определяющая круг используемых понятий и затем предлагающая изображения из коллекции, которые наиболее адекватны содержанию презентации.

Можно вставлять рисунки и сканированные фотографии из других приложений или из других мест (это называется импортированием графики). Чтобы вставить рисунок из другого приложения, укажите в меню **Вставка** на команду **Рисунок** и затем щелкните **Из файла**. При этом появится панель инструментов **Настройка изображения**, с помощью которой можно кадрировать рисунок, перекрасить его, обвести рамкой, отрегулировать яркость и контрастность. Чтобы вставить сканированную фотографию, укажите в меню **Вставка** на команду **Рисунок**, затем щелкните **Со сканера**. Изображение откроется в приложении Microsoft Photo Editor, где его можно изменить.

Существует два вида рисунков: растры (их нельзя разгруппировать) и рисунки типа метафайлов, которые можно разгруппировывать, преобразовывать в объекты PowerPoint и затем редактировать с помощью инструментов рисования PowerPoint. Большинство картинок принадлежат формату метафайла. Чтобы разгруппировать картинку и преобразовать ее в объект PowerPoint, выделите ее и выберите в меню **Действия** команду **Разгруппировать**. После этого картинку можно изменять, как любой другой нарисованный объект. Например, можно вставить изображение человека, разгруппировать его, изменить цвет одежды и затем включить измененное изображение в другую картинку.

Можно создавать собственные рисунки, используя инструменты рисования PowerPoint. PowerPoint распознает множество форматов рисунков. Поскольку все приложения Office используют одни и те же графические фильтры, то в PowerPoint можно использовать фильтр, установленный, например, вместе с приложением Word.

Открытие презентации.

Диалоговое окно **Открыть** (меню **Файл**) позволяет открывать презентации в различных местах, в том числе на жестком диске

вашего компьютера или на сетевом диске, к которому вы подключены. Если сеть поддерживает адреса UNC, вы сможете открыть презентацию в сети даже при отсутствии соединения с сетевым сервером.

При наличии доступа к сети Интернет можно также открыть презентацию, расположенную на сервере HTTP службы Web или на сервере FTP. Если в вашей организации имеется корпоративная сеть с выходом в Интернет, в ней вы также сможете открывать презентации. О том, как получить доступ в Интернет, см. Microsoft Office 97 Resource Kit. Для сведений о том, как достать пакет Office Resource Kit, нажмите кнопку.

Если вы обладаете правом записи какой-либо презентации, ее можно открыть (на жестком диске или сетевом диске) в виде копии и работать с ней вместо оригинала. Презентацию, независимо от ее местонахождения, можно открыть только для чтения, чтобы сохранить оригинал нетронутым.

На рис. 138 приведены примеры создания документов.

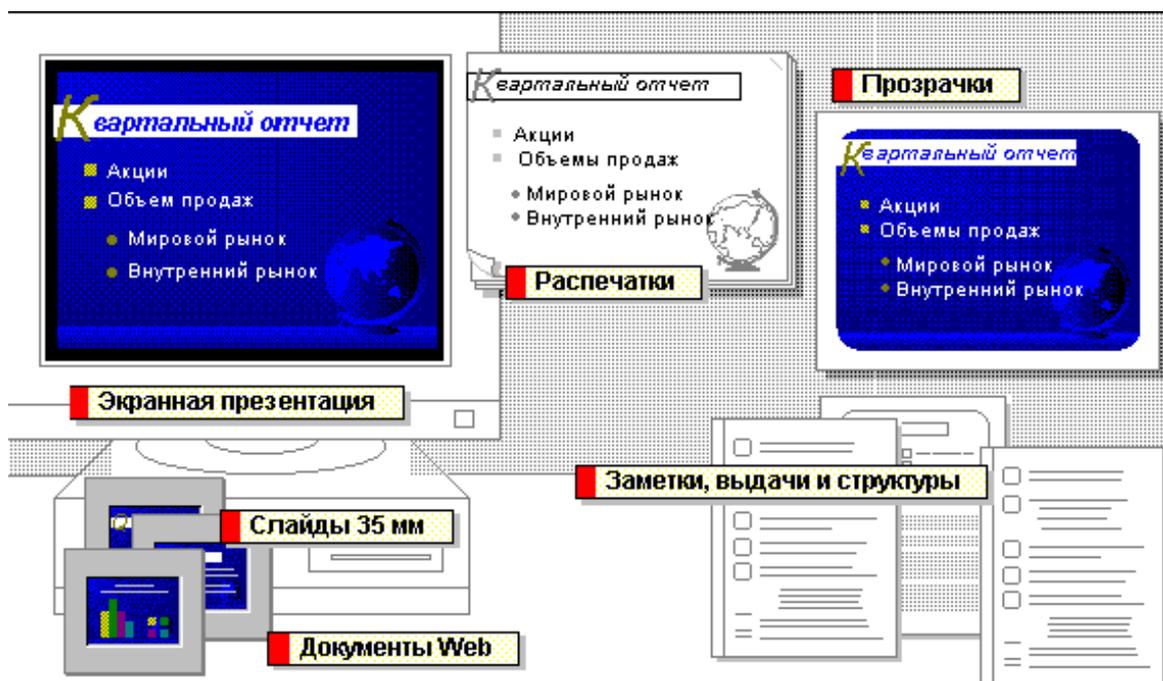


Рис. 138

Возможности PowerPoint позволяют придать всем создаваемым презентациям единый вид. Существует три метода управления внешним видом слайдов: с помощью образцов, цветовых схем и шаблонов оформления.

Образец слайдов определяет формат и размещение заголовков и текста, вводимых в слайды, а образец заголовков контролирует формат и размещение титульного слайда и всех остальных слайдов, описанных вами как титульные (например, начального слайда раздела). Кроме того, в образцах находятся элементы фона; например, графика, включаемая в слайды. Любое изменение, внесенное в образец слайдов, отражается в каждом слайде. Чтобы какой-либо слайд отличался от образца, измените слайд.

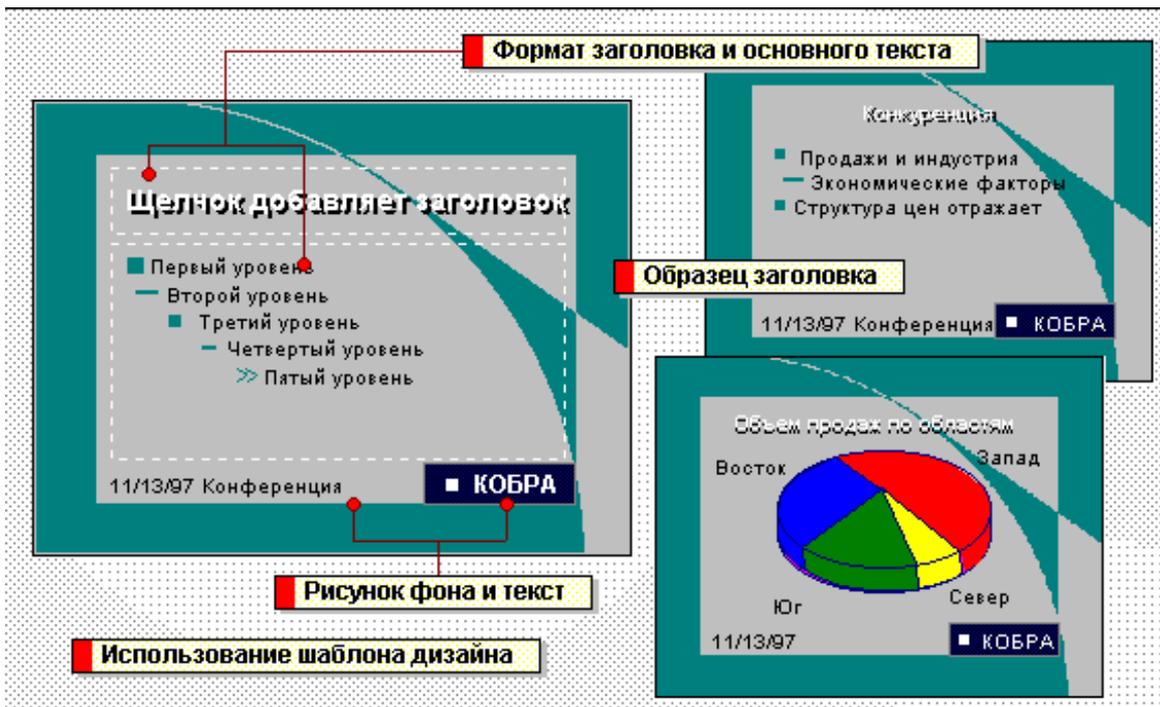


Рис.139

На рис.139 приведены образцы слайдов. Шаблоны оформления содержат цветовые схемы, образцы слайдов и заголовков с нестандартным форматированием, а также стилизованные шрифты, предназначенные для конкретных видов оформления. Если к презентации применяется шаблон оформления, его образец слайдов и цветовая схема заменяют образец слайдов и цветовую схему исходной презентации. После применения шаблона каждый добавляемый в презентацию слайд, независимо от авторазметки, будет гармонизировать с остальными слайдами.

Режимы просмотра в PowerPoint. На рис.140 приведены различные режимы просмотра презентаций, подготовленных в PowerPoint.

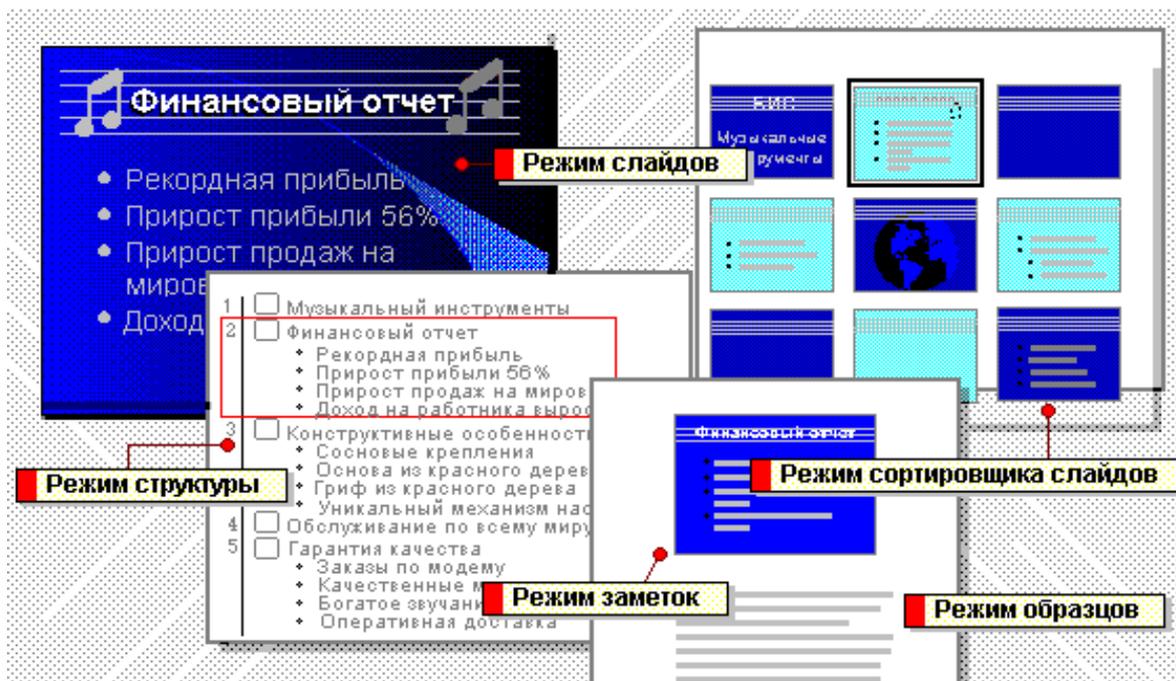


Рис. 140

Выбор нескольких файлов.

- Чтобы выбрать в списке диалогового окна **Открыть** несколько файлов вразброс, щелкните один из них, затем нажмите клавишу CTRL и, удерживая ее, щелкните поочередно остальные файлы.

- Чтобы выбрать в списке диалогового окна **Открыть** несколько подряд идущих файлов, щелкните первый из этих файлов, затем нажмите клавишу SHIFT и, удерживая ее, щелкните последний файл.

Совет . Если случайно выбран ненужный файл, нажмите клавишу CTRL и, удерживая ее, щелкните этот файл.

Анимация текста и объектов. Анимация текста, графики, звука, кино и других объектов на слайдах позволяет подчеркивать различные аспекты содержания, управлять потоком информации, а также делает презентацию более привлекательной. Вы можете обеспечить появление каждого маркированного пункта независимо от остальных или постепенное появление объектов одного за другим. Для каждого пункта или объекта можно установить порядок его появления на слайде, например, «влетание» с левой или с правой стороны; а также порядок изменения пунктов или объектов при добавлении нового элемента, например, потускнение или изменение в цвете. В составе PowerPoint 97 появилась новая возможность: анимация элементов диаграммы.

Порядок и время показа анимационных элементов можно изменять, а показ можно автоматизировать, чтобы не пользоваться мышью. Для подготовки и предварительного просмотра анимации текста и объектов выберите в меню **Показ слайдов** команду **Настройка анимации**.

Вставка гиперссылки в презентации.

1. Сохраните презентацию, в которую вставляется гиперссылка. При вставке гиперссылки в несохраненную презентацию нельзя создать относительную ссылку.

2. Выделите текст или объект, представляющий гиперссылку.

3. В меню **Показ слайдов** выберите команду **Настройка действия**.

4. Чтобы задать переход по щелчку в выделенном объекте, щелкните вкладку **По щелчку мыши**.

Чтобы задать переход по указанию мышью на объект, щелкните вкладку **По указанию мышью**.

Чтобы назначить объекту более одного действия, например, переход по гиперссылке и звук, установите гиперссылку на вкладке **По щелчку мыши**, а звук – на вкладке **По указанию мышью**.

5. Щелкните **Перейти по ссылке**, затем выберите место назначения гиперссылки.

6. Установите другие необходимые параметры, затем нажмите кнопку **ОК**.

7. Для предварительного просмотра гиперссылки нажмите кнопку **Показ слайдов**, находящуюся в левом нижнем углу окна PowerPoint.

Советы.

- Чтобы вставить гиперссылку в готовую фигуру, такую, как кнопка или стрелка, укажите в меню **Показ слайдов** на команду **Управляющие кнопки**, затем выберите нужную кнопку. После этого автоматически откроется диалоговое окно **Настройка действия**, в котором можно установить гиперссылку.

- Устанавливая гиперссылку для перехода в какой-либо слайд, вставьте также в этот слайд гиперссылку для возврата в исходный слайд.

- Перед перемещением файлов убедитесь в доступности места назначения гиперссылки, установив флажок **Относительный путь к ссылке**. Для сведений о том, как установить базовый адрес гиперссылки для презентации, нажмите кнопку .

Задание базового адреса гиперссылки для презентации.

При создании гиперссылки можно задать путь к ее месту назначения в виде абсолютной или относительной ссылки.

Абсолютная ссылка – это фиксированное описание местоположения файла, содержащее полный адрес места назначения, например, с:\Мои документы\Отчет.doc. Для перемещения или копирования файла, содержащего гиперссылку или ее файл назначения, используйте относительную ссылку. Чтобы изменить путь, описывающий относительную ссылку, установите для презентации базовый адрес гиперссылки.

1. Откройте презентацию, для которой следует установить базовый адрес гиперссылки.

2. Выберите в меню **Файл** команду **Свойства**, затем перейдите на вкладку База гиперссылки.

3. В поле **База гиперссылки** введите путь относительной ссылки для всех гиперссылок данной презентации.

8.5. РАБОТА В СИСТЕМЕ MICROSOFT FRONTPAGE

Microsoft Frontpage призван облегчить вам задачу достойно представить себя в WWW или создать Web-сайт для сети *Intranet* вашей организации. Frontpage, органично вписывающийся в пакет приложений *Microsoft Office*, стал первым продуктом широкого использования для *Internet*, сочетающим в себе клиентскую и серверную части и обеспечивающим возможность разработки сайта в целом и установки его на большинство популярных серверов. Если вы хотите создать свой Web-сайт, но с программированием знакомы только понаслышке, не беспокойтесь — Frontpage способен взять на себя всю необходимую работу по программированию. Microsoft Frontpage позволяет создавать и обслуживать Web-сайт с помощью Проводника Frontpage (Frontpage Explorer), разрабатывать высококачественные страницы с помощью Редактора Frontpage (Frontpage Editor) и запускать Web-сайт под управлением одного из Персональных Web-серверов для Windows или любого другого сервера по вашему желанию. Кроме того, встраивать в сайт материал из файлов Microsoft Office и создавать в этих файлах внутренние, внешние и перекрестные ссылки; графические декорации сайтов, редактирование фреймов в режиме

WYSIWYG, поддерживать динамический HTML, усовершенствованные формы представления сайтов в Проводнике, поддерживать таблицы описаний стилей, редактировать изображения прямо в Редакторе Frontpage. Frontpage позволяет вам создавать страницы профессионального качества, при этом не требуя от вас знания тонкостей HTML. Одна из главных прелестей Редактора заключается в его умении представлять страницы в режиме WYSIWYG (what you see is what you get — что видите, то и получаете). Это означает, что в браузере они будут выглядеть так же, как и в Редакторе. Работа в Редакторе Frontpage весьма похожа на работу с текстовым процессором, подобным Microsoft Word. Те, кому приходилось иметь дело с Microsoft Word, наверняка узнают стандартные кнопки, такие, как полужирный шрифт, курсив и подчеркивание, маркированные и нумерованные списки, повтор и отмена операции и т.п. В Редакторе очень удобно — буквально в несколько движений мыши — строить таблицы и фреймы (frames). Как Word, Редактор Frontpage позволяет одновременно открывать несколько файлов, а т. к. Редактор оперирует страницами, то смело можно сказать, что он позволяет открывать одновременно несколько страниц. Эта возможность очень полезна, например, в тех случаях, когда, чтобы соблюсти согласованность и точность информации, требуется быстро переходить от одной страницы к другой. Редактор также позволяет скопировать страницу (вместе со всем кодом HTML) из Internet и отредактировать ее. Это может быть удобно, когда вы хотите быстро получить информацию с других своих сайтов. Разумеется, при копировании информации с других сайтов будьте крайне внимательны, чтобы не нарушить ничьи авторские права; помните, что плагиат в результате очень дорого обходится.

Запуск Редактора

Как попасть в Редактор? Это можно сделать несколькими способами:

– Дважды щелкнув на странице в структуре папок (**Folders View**), списке файлов (**All Files View**), карте навигации (**Navigation View**) или схеме гиперссылок (**Hyperlinks View**) Проводника. При этом будет вызван Редактор и соответствующая страница открыта в его окне.

– Щелкнув правой кнопкой мыши в структуре папок, списке файлов, карте навигации или схеме гиперссылок Проводника и выбрав в появившемся контекстном меню команду **Open**.

– Щелкнув правой кнопкой мыши в списке заданий Проводника (**Tasks View**) на задаче, связанной со страницей, и выбрав в контекстном меню опцию **Do Task**; откроется Редактор с указанной в окне страницей.

– Запустив непосредственно Редактор (либо прямо из Microsoft Windows, либо из Проводника Frontpage при нажатии на панели инструментов кнопки **Show Frontpage Editor** или командой **Show Frontpage Editor** из меню **Tools**). В этом случае Редактор откроется с пустым экраном; чтобы открыть нужную страницу, используйте команду **Open** меню **File** и приступайте к редактированию.

При запуске Редактор откроет свое собственное окно, в верхней части которого, как в большинстве программ подобного класса, расположены меню и панели инструментов. Когда все панели инструментов видимы, Редактор выглядит так, как показано на рис. 141.

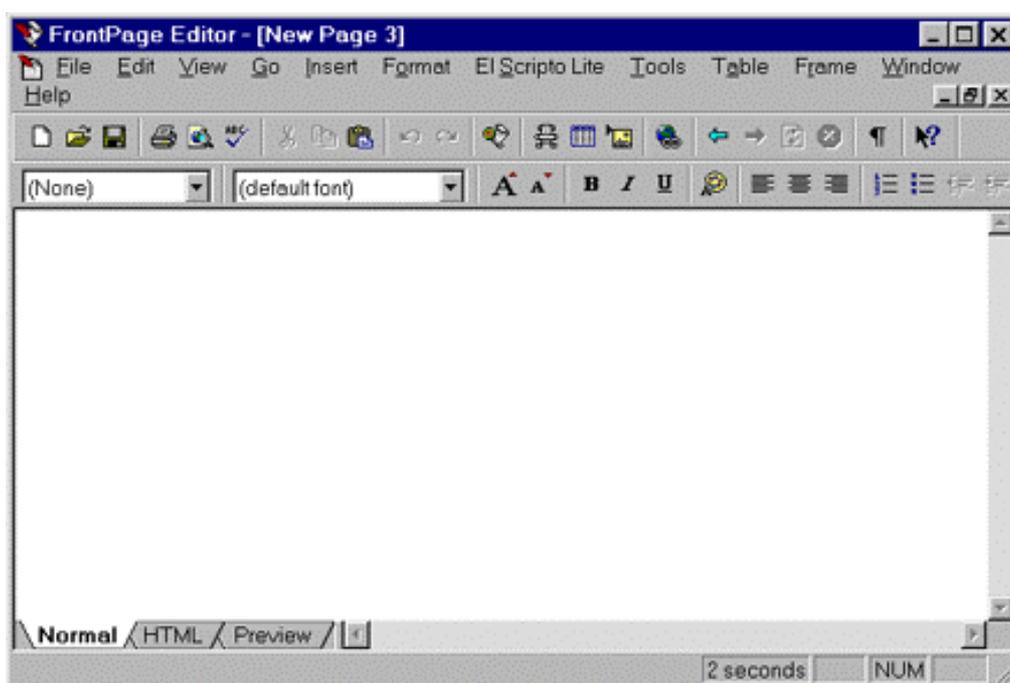


Рис. 141. Общий вид окна Редактора Frontpage

Внизу окна редактирования – **Normal** (Обычный вид), **HTML** (Код HTML) и **Preview** (Предварительный просмотр) похожи на вкладки рабочих листов. Эти вкладки позволяют увидеть редак-

тируемую страницу с различных точек зрения. Для начала мы будем работать в обычном режиме (вкладка **Normal**). Однако прежде чем начать работу с Редактором, изучим, как наилучшим образом настроить его и максимально приспособить к своим потребностям.

Изменение размеров окна

Обычно бывает удобно раздвинуть окно редактора на весь экран, чтобы видеть как можно большую площадь редактируемой страницы. Обратите внимание на три кнопки, расположенные в правом верхнем углу окна Редактора. Чтобы максимально распахнуть окно, нажмите кнопку максимизации — она средняя в этой группе (разумеется, если окно еще не развернуто на весь экран). Чтобы восстановить предыдущий размер и местоположение окна Редактора, нажмите кнопку восстановления, которая займет место кнопки максимизации. Этот набор из трех кнопок абсолютно такой же, как и в любых других приложениях Microsoft Windows NT 4.0 и Windows 95, Windows 98.

Работа с панелями инструментов

Редактор располагает рядом панелей инструментов, которыми можно управлять через меню **View** (Вид): **Standard** (Стандартная), **Format** (Форматирование), **Image** (Рисование), **Forms** (Формы), **Advanced** (Дополнительно) и **Table** (Таблица). Когда они все видны, занимают существенную часть окна Редактора. Поэтому те, что вам не нужны, лучше спрятать. Чтобы скрыть панель инструментов, откройте меню **View** и уберите галочку напротив имени этой панели. Чтобы показать спрятанную панель, найдите ее имя в меню **View** и поставьте напротив него галочку, тогда панель появится на том же самом месте, которое она раньше занимала. Приглядевшись, вы увидите в меню **View** пункты **Status Bar** (Строка состояния) и **Format Marks** (Символы форматирования). Панели инструментов Редактора можно располагать в любом месте экрана. Чтобы передвинуть панель инструментов, щелкните на любом месте панели, не занятом кнопками, и перетащите ее на новое место. Если вы оставите ее где угодно в окне редактирования, то панель будет плавающей. Если такой вариант вас не устраивает, подтащите панель к любому краю окна и оставьте ее там.

Строка состояния и символы форматирования

Строку состояния, расположенную в самом низу окна Редактора, можно прятать и показывать; для этого служит команда **Status Bar** меню **View**.

Редактор в некоторых случаях неявно использует символы форматирования, такие, как знаки абзаца. Чтобы показать или спрятать эти метки на странице, выберите в меню **View** опцию **Format Marks**.

Приступим к строительству

В вашем распоряжении будут все обычные элементы страниц, такие, как тексты, гиперссылки, заголовки и ряд менее ординарных: *таблицы (tables)*, *фреймы (frames)*, *бегающие строки (marquees)*, *фоновые звуки (background sounds)* и *видео (video)*. Еще один важный элемент Web-страниц — изображения, также Редактор позволяет вставлять в страницы *Формы (forms)* и *Компоненты (Components)* самые передовые Web-технологии, такие, как компоненты ActiveX и апплеты Java, которые легко встраиваются в HTML-страницы.

Размещая на странице очередной элемент, старайтесь придерживаться простого принципа: считайте, что вы работаете в обычном текстовом процессоре, например, в Microsoft Word. Многие приемы манипулирования элементами страниц в Редакторе полностью аналогичны тем, которые знакомы вам по работе в Word. Многие меню и панели инструментов также повторяют аналогичные элементы в Word. В Редакторе Frontpage реализованы 25 наиболее популярных клавиатурных комбинаций Microsoft Word, так что пользователи Office будут чувствовать себя в Редакторе как дома. Frontpage припас целых два подарка: это *шаблоны (templates)* и *мастера (wizards)*. Шаблон, как несложно догадаться, представляет собой каркас, который вы можете использовать как основу при создании нового документа. Мастер же представляет собой программный модуль, состоящий из одного или нескольких экранов, который, задавая ряд вопросов, выясняет ваши потребности и в результате генерирует удовлетворяющий им документ.

Шаблоны — это образцы сайтов или страниц, которые Frontpage предлагает заполнять по мере необходимости. Как и мастера, они создают структуру, которая может служить хорошей стартовой площадкой для разработки сайта или страницы. Однако

мастера предлагают вам процесс заказного создания сайта или страницы, тогда как выбрав шаблон, вы получите точную копию этого шаблона, с образцом текста, который вы сможете заменить своим собственным.

Вы можете получить шаблон сайта в Проводнике Frontpage, выбрав пункт **New** (Создать) меню **File** (Файл) и затем выбрав **Frontpage Web** (Сайт Frontpage) из появившегося подменю.

Использование шаблонов страниц

Страницы с помощью шаблонов создаются в Редакторе. Это довольно простой процесс:

1. Откройте сайт, в котором будет находиться новая страница, в Проводнике. (При желании можно пропустить этот шаг, т. к. после того, как вы сохраните страницу, ее можно импортировать в любой из существующих сайтов.)

2. В Редакторе выберите пункт **New** из меню **File**.

3. В диалоговом окне **New** выберите нужный шаблон из предлагаемого списка (рис.142) и нажмите кнопку **OK**.

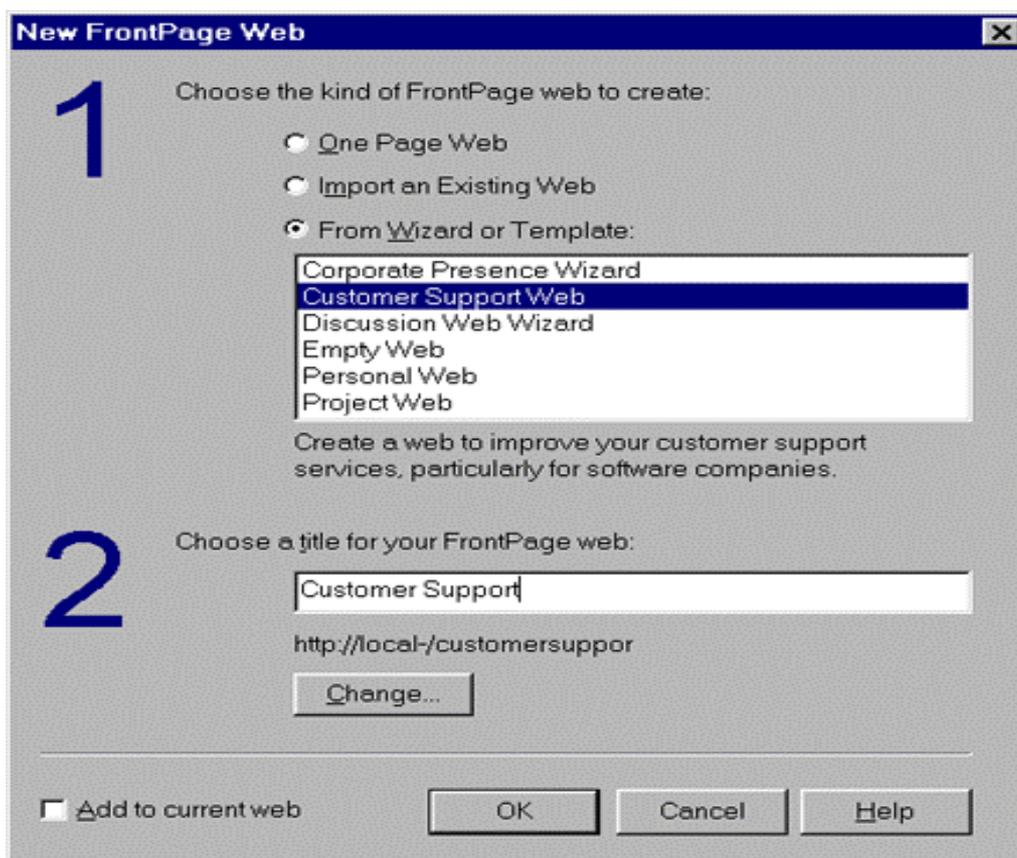


Рис.142. Окно диалога New Frontpage Web

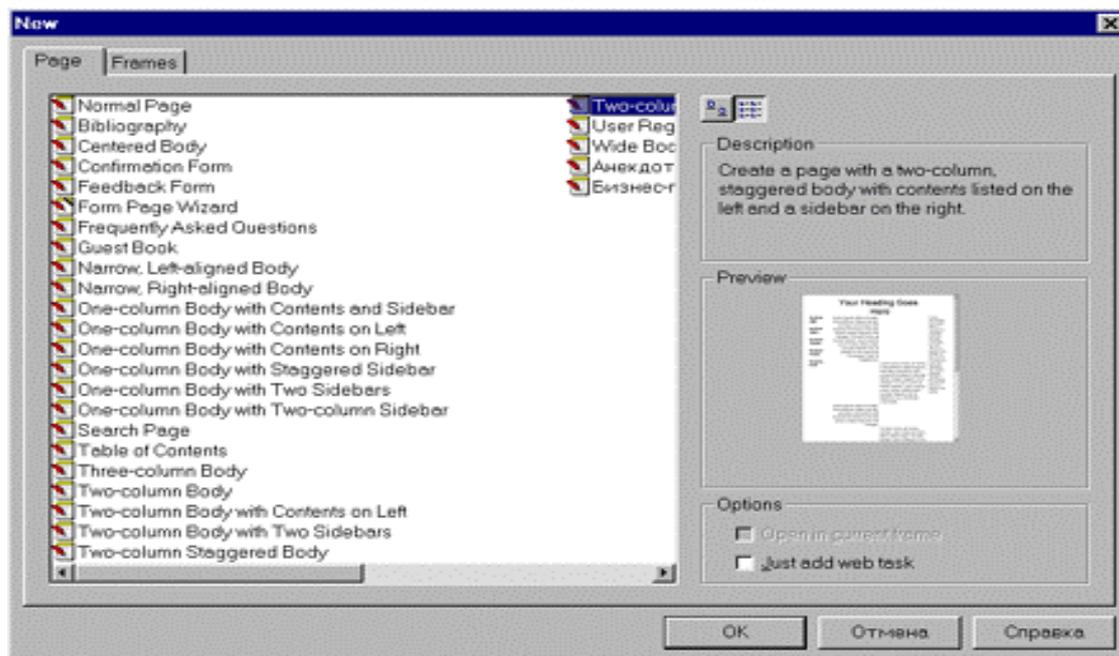


Рис. 143. Диалоговое окно New, вкладка Page

Вам предоставляется богатый выбор различных шаблонов страниц, которые вы сможете использовать в своем сайте. Шаблоны перечислены в списке в левой части диалогового окна **New**, а в правой части этого окна отображается описание текущего шаблона и его примерный вид. Если в данный момент в Редакторе открыта страница фреймов и вы хотите, чтобы новая страница открылась в текущем фрейме, то в этом случае вам следует установить флажок **Open in current frame** (Открыть в текущем фрейме). В том случае, если ваш сайт открыт в Проводнике и вы хотите внести создаваемую страницу в перечень заданий с тем, чтобы доработать ее впоследствии, поставьте флажок **Just add web task** (Только добавить в перечень заданий).

В каждой новой странице, созданной с помощью шаблона, присутствует «рыба» (некий греческий текст, т. н. *lorem ipsum*) и образец текста, например, *Place Main Title Here* (Поместите здесь основной заголовок), которые вы впоследствии замените своим собственным текстом. На рис. 144 приведен пример, созданный при помощи шаблона четырехстолбцовой страницы с шахматным расположением (**Four Column Staggered page**), где показаны заголовки и образец текста.

Фреймы (frames) позволяют разбивать страницы на прямоугольные области, в каждой из которых отображается своя собст-

венная страница. Вы можете разместить один или несколько фреймов на странице (такая страница называется страницей фреймов или *фреймсет (frameset)*). Это означает, что вы можете создать страницу, различные области которой отображают различное содержание. Изменение содержания в одном из фреймов необязательно влечет за собой изменение содержания в другом, но если вы захотите, то сможете сделать так, чтобы страница, получаемая по ссылке, находящейся в одном фрейме, отображалась в области другого фрейма.

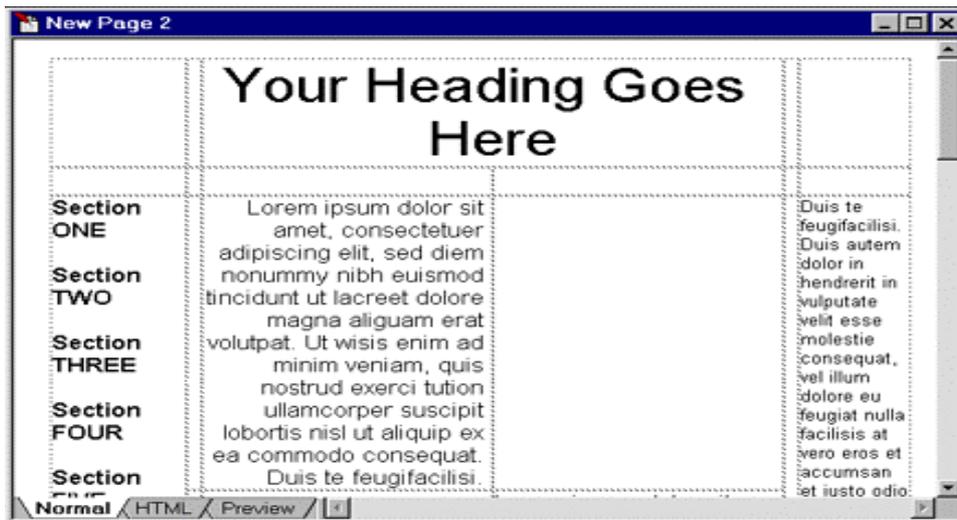


Рис. 144. Пример новой страницы

Создание фреймов

На рис.145 приведен классический пример использования фреймов: представьте себе страницу, поделенную на две вертикальные области, каждая из которых представляет собой отдельный фрейм. Левый фрейм занимает оглавление, содержащее в себе полный список ссылок на все страницы сайта. Содержимое правой части страницы меняется в зависимости от того, по какой ссылке в оглавлении вы щелкнете на левой стороне страницы. Если вы щелкнете на ссылке, ведущей на страницу под названием «Записки», то страница «Записки» возникнет на правой стороне экрана; если вы щелкнете на ссылке "Результаты", то отобразится страница "Результаты" и т.д.

Страница разделена на два фрейма. Щелкнув мышью на любой из ссылок в оглавлении, вы увидите, что в правой части будет отображена соответствующая страница.

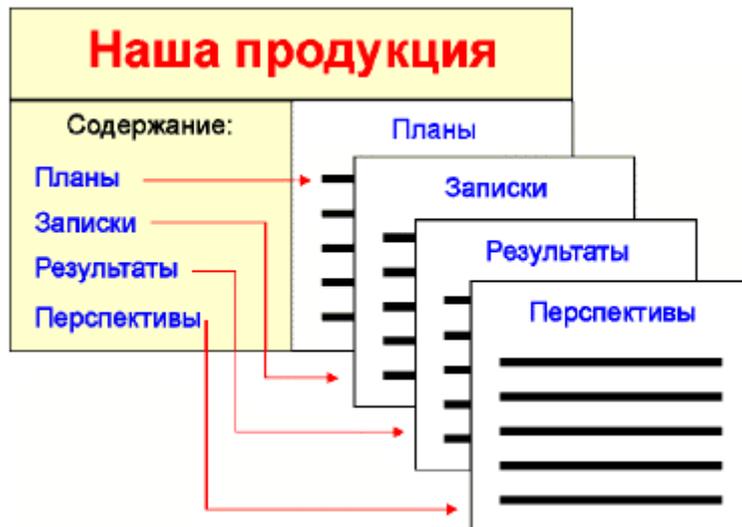


Рис. 145

Использование шаблонов фреймов

Страницы фреймов вы будете создавать в Редакторе. Этот процесс происходит примерно так:

1. Откройте в Проводнике сайт, в котором вы хотите создать свой новый фреймсет.
2. В Редакторе выберите пункт **New** из меню **File**.
3. В появившемся диалоговом окне **New** выберите шаблон из вкладки **Frames** (рис.146), а затем нажмите кнопку **ОК**.

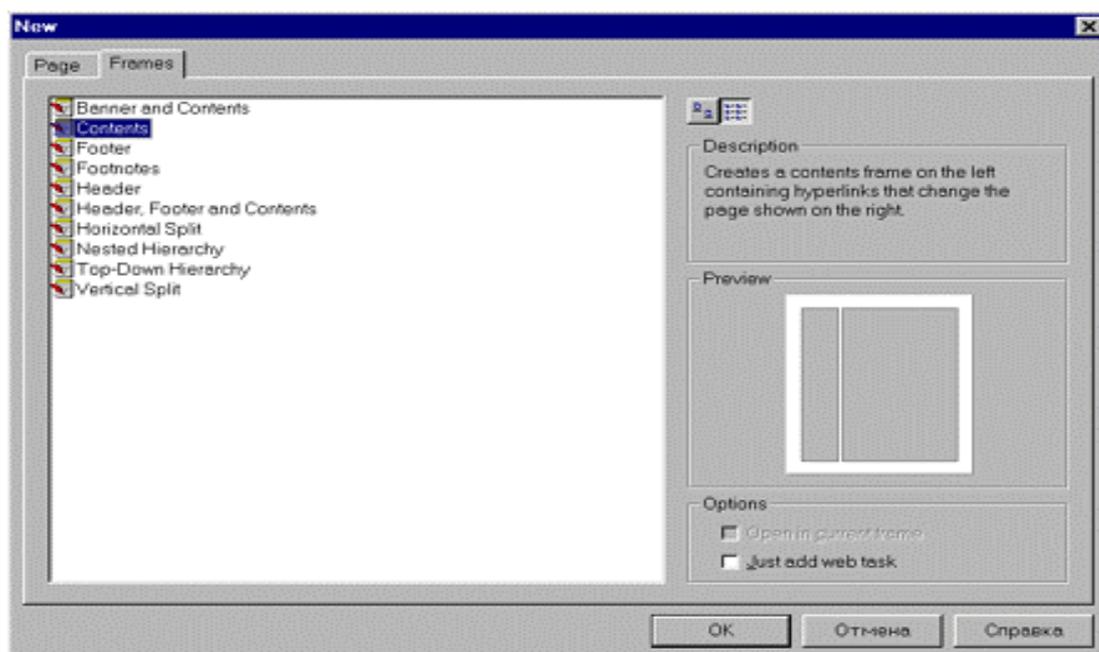


Рис. 146. Окно New, вкладка Frames

Вы сможете предварительно оценить выбранный вами фреймсет, т.к. его макет будет показан в правой части диалогового окна **New**. Данное диалоговое окно работает точно так же, как ранее описанное. Как только вы сделаете свой выбор, Frontpage сгенерирует страницы и поместит их в Редактор.

Закладки

Закладка (*bookmark* или *anchor*) — это участок текста (или просто один или несколько символов), являющийся точкой перехода по гиперссылке. Использование закладок как целевых точек при простановке ссылок позволит посетителям вашего сайта «перепрыгивать» именно на то место в документе, где стоит закладка, а не в начало страницы, как происходит по умолчанию. Закладки видны в Редакторе как текст, подчеркнутый пунктиром. Для того чтобы создать закладку, сделайте следующее:

1. В той точке текста, куда вы хотите «перескакивать» по гиперссылке, выделите один или несколько символов.
2. Выберите **Bookmark** (Закладка) из меню **Edit** (Редактирование). Появится диалоговое окно **Bookmark** (рис.147).

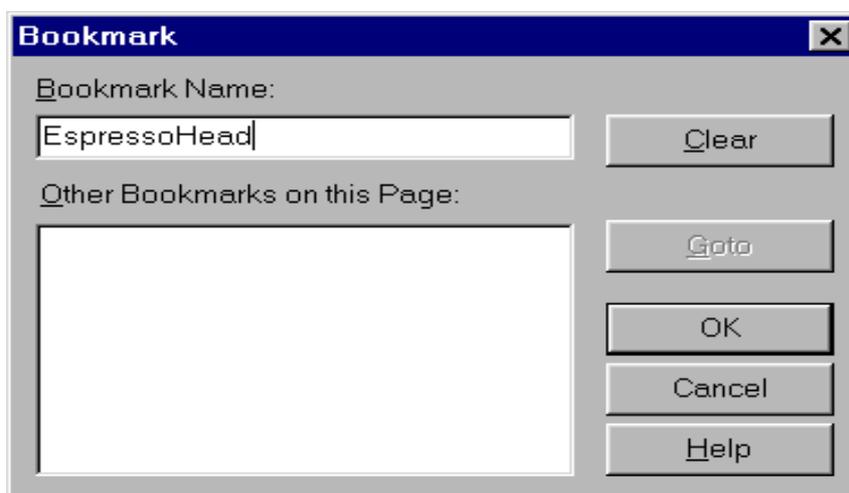


Рис. 147. Диалоговое окно **Bookmark**

3. Введите название закладки в строке **Bookmark Name**. Старайтесь называть ваши закладки по возможности просто и понятно, потому что позже при простановке ссылок вам понадобится ввести имя закладки или выбрать его из списка. Если документ уже содержит какие-либо закладки, то они будут отображены в данном диалоговом окне.

4. После того как вы введете имя закладки, щелкните мышью по кнопке ОК. В Редакторе вы увидите, что отмеченный текст теперь подчеркнут пунктиром — это означает, что здесь установлена закладка.

Поиск нужной закладки

Предположим, на текущей странице много закладок и вы, будучи в диалоговом окне **Bookmark**, пытаетесь отыскать местоположение в тексте одной из них. Для того чтобы быстро найти участок текста, соответствующий закладке, выделите закладку в списке и нажмите кнопку **Goto** (Переход). Нужный участок текста с выделенной закладкой автоматически отобразится на экране. Это простая альтернатива ручной прокрутке страницы в поисках нужной закладки.

Удаление закладки

Чтобы удалить закладку, выделите ее и выберите пункт **Bookmark** из меню **Edit** или просто щелкните на закладке правой кнопкой мыши и в контекстном меню укажите пункт **Bookmark Properties**. В появившемся диалоговом окне **Bookmark** нажмите кнопку **Clear** (Очистить). Диалоговое окно после этого закроется, а закладка будет удалена. Данная процедура никак не влияет на текст, она просто удаляет закладку.

Ссылки

Ссылка (она же *гиперссылка*) — это соединение между двумя точками. Посетители сайта могут, щелкнув по ссылке, «перепрыгнуть» туда, куда она ведет; место назначения ссылки представляется в виде URL (Uniform Resource Locator, Унифицированный указатель ресурса). Ссылки могут указывать на текст, изображения и другие файлы (например, на документы Microsoft Office), а также на закладки внутри документов

Если у вас есть сайт intranet с включенными в него документами Microsoft Office, то эти документы также могут быть соединены между собой при помощи ссылок. Например, вы можете проставить ссылку с документа Microsoft Word на файл Excel. Когда вы щелкаете по такой ссылке, автоматически запускается программа Excel и открывается файл, на который указывает ссылка. Используя Frontpage, вы сможете очень просто связывать между

собой HTML-страницы и файлы Office, создавая мощный и динамичный сайт intranet. В данной книге описывается только способ создания ссылок из Frontpage на файлы Office.

Создание ссылки на страницу или закладку

Для того чтобы установить ссылку, выделите участок текста или изображение, с которого будет осуществляться переход, и нажмите комбинацию клавиш <Ctrl>+<K> или выберите пункт **Hyperlink** (Гиперссылка) из меню **Edit**. Вы увидите диалоговое окно **Create Hyperlink** (Создание гиперссылки), показанное на рис.148.

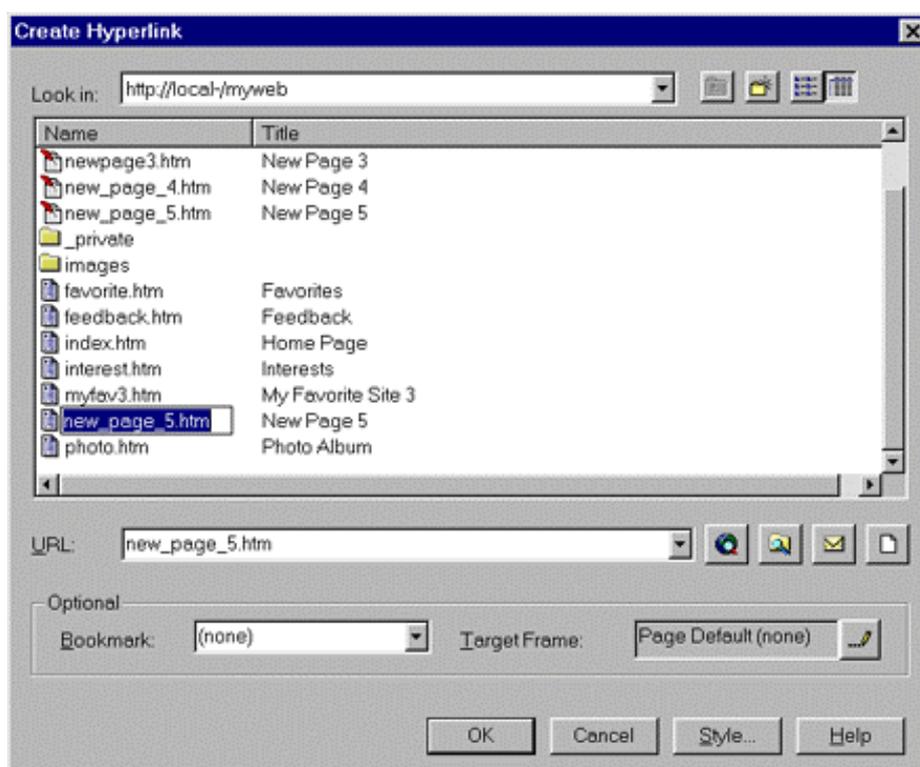


Рис. 148. Диалоговое окно Create Hyperlink

Вы можете устанавливать ссылки на объекты нескольких различных типов: на страницу в Проводнике Frontpage, на любой URL, используя при этом ваш Web-браузер, на любой файл, находящийся на вашем компьютере, на любой адрес e-mail. Вы можете также создать новую страницу и установить ссылку на нее. Помимо этого, вы можете указать, что целью перехода является определенная закладка или фрейм. Ниже описаны различные способы создания гиперссылок:

– *Ссылки на страницы сайта, открытого в данный момент в Проводнике.* Если какой-либо сайт в данный момент открыт в проводнике Frontpage, то в диалоговом окне вы увидите перечень всех страниц этого сайта, и на любую из них можно установить гиперссылку.

– *Ссылки на URL.* Вы можете ввести точный адрес (URL) страницы в текстовом поле или же нажать на кнопку **Use your Web Browser to select a page** (Указать страницу с помощью браузера). По нажатию этой кнопки запускается Web-браузер, установленный на вашем компьютере. Когда вы найдете нужную страницу в Internet, адрес этой страницы будет автоматически помещен в текстовое поле **URL** в диалоговом окне **Create Hyperlink**.

– *Ссылки на локальные файлы.* Если вы хотите установить гиперссылку на файл, находящийся на вашем компьютере, вы можете сделать это, используя кнопку **Make a hyperlink to a file on your computer** (Создать ссылку на файл на вашем компьютере). В появившемся диалоговом окне **Select File** (Выбор файла) вы можете указать нужный файл.

– *Ссылки на адрес e-mail.* Если вы хотите создать ссылку, которая позволяла бы посетителям вашего сайта отправить письмо по какому-либо конкретному адресу, то в этом случае можно использовать кнопку **Make a hyperlink that sends E-mail** (Создать ссылку для отправки письма по электронной почте). Вы увидите появившееся диалоговое окно **Create E-mail Hyperlink** (Создание почтовой ссылки), изображенное на рис.149. Для того чтобы такая ссылка работала, вам достаточно просто ввести адрес получателя в текстовом поле.

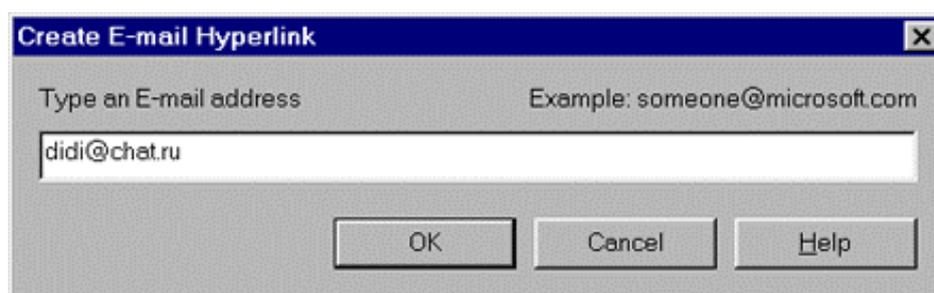


Рис. 149. Диалоговое окно Create E-mail Hyperlink

– *Ссылка на новую страницу.* Данная возможность позволяет создать ссылку, указывающую на еще не созданную страницу. Нажатие на кнопку **Create a page and link to the new page** (Соз-

дать страницу и проставить ссылку) вызывает появление диалогового окна **New** (Новая страница).

Из списка предлагаемых шаблонов выберите тип страницы, который вы хотите создать. В зависимости от вашего выбора Frontpage либо немедленно создаст новую страницу, либо просто добавит задачу создания этой страницы в список задач. Вы также можете использовать еще две дополнительные возможности, предоставляемые диалогом **Create Hyperlink**:

– *Закладка*. Если вы хотите, чтобы создаваемая гиперссылка указывала на закладку, установленную в целевом документе, то вам необходимо ввести название закладки в предназначенном для этого поле **Bookmark**.

– *Целевой фрейм*. Чтобы указать, в каком фрейме должна отображаться страница, загружаемая в результате перехода по гиперссылке, введите название целевого фрейма в поле **Target Frame**.

После того, как сделаны все необходимые настройки для данной ссылки, нажмите кнопку **OK**. Вы увидите, что выделенный участок текста окрасился цветом, установленным для ссылок текущей страницы. Чтобы проверить новую ссылку, нажмите клавишу <Ctrl> и щелкните по ссылке кнопкой мыши.

Как выделить гиперссылку

Если вы хотите изменить однажды созданную гиперссылку, то сначала вы должны ее выделить. Все, что вам для этого понадобится – это щелкнуть кнопкой мыши в любом месте внутри участка текста, занимаемого ссылкой. Вы можете, конечно, выделить текст ссылки обычным способом, но делать это вовсе не обязательно. Когда курсор становится на участок текста, содержащий гиперссылку, активизируются кнопки и пункты меню, относящиеся к редактированию ссылок.

Удаление ссылки. Чтобы удалить ссылку, выделите ее и выберите в меню **Edit** пункт **Unlink** (Отсоединить).

Общие поля страниц. Предположим, что вы хотите разместить логотип вашей компании внизу каждой из страниц создаваемого вами сайта. Вы, наверное, думаете, что придется вручную помещать изображение на каждую страницу или прибегать к использованию фреймов, но это не так. Для этого имеется другой, более легкий способ. Вы можете указать, что внизу каждой страницы вашего сайта должен располагаться участок, который на са-

мом деле принадлежит какой-либо иной странице HTML. Таким образом, вы можете, поместив логотип на такой странице, получить его изображение на всех документах вашего сайта.

Чтобы применить такой совместно используемый фрагмент ко всему сайту, вам необходимо в *Проводнике* выбрать в меню **Tools** пункт **Shared Borders** (Общие поля). Если вы хотите установить общие поля на отдельно взятой странице, то вам следует совершить те же самые действия в *Редакторе*.

Если вы решите установить общие поля на всех страницах сайта, то вы увидите диалоговое окно **Shared Borders**, показанное на рис. 150.

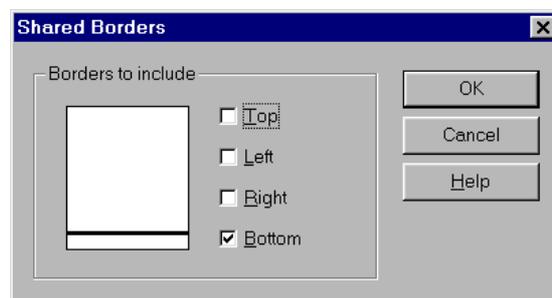


Рис. 150. Диалоговое окно Shared Borders

Если вы захотите поместить общие поля на отдельной странице, то получите диалоговое окно **Page Borders** (Поля страницы), изображенное на рис. 151.

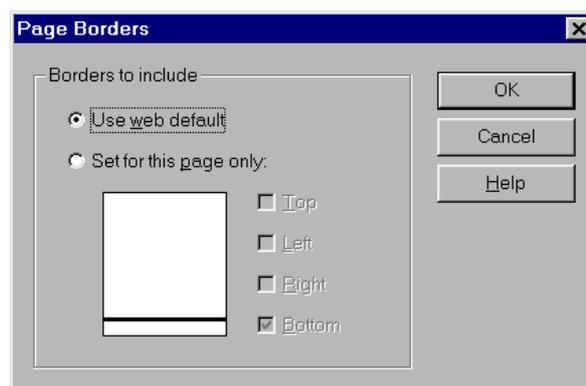


Рис. 151. Диалоговое окно Page Borders

Оба диалоговых окна работают примерно одинаково, за исключением того, что в Редакторе у вас есть возможность применить все настройки, уже установленные в Проводнике для всего сайта, или назначить их отдельно для текущей страницы.

8.6. ОБРАЗОВАТЕЛЬНЫЕ САЙТЫ В ИНТЕРНЕТ

Сегодня Internet стал не мечтой, а реальностью. Пользователь практически любой специальности может в нем найти необходимую, интересующую его информацию. Данный параграф посвящен краткому обзору образовательных сайтов. Думаем, материал будет полезен учителям, студентам и учащимся.

http://wepsib.ru/noos/informatika/z_inf/pos.Htm

Сайт для преподавателей (рис.152). На главной странице в разделе Информатика расположены следующие ссылки: Кабинет; Стандарты и программы; Курсы; Учебная литература; Медиатека; Методика, опыт; Программное обеспечение; Проекты, конкурсы; Олимпиады; Наука и профессия; Конференция; Занимательная информатика; Железо; Компьютерная безопасность; Компьютер и здоровье; Архив.

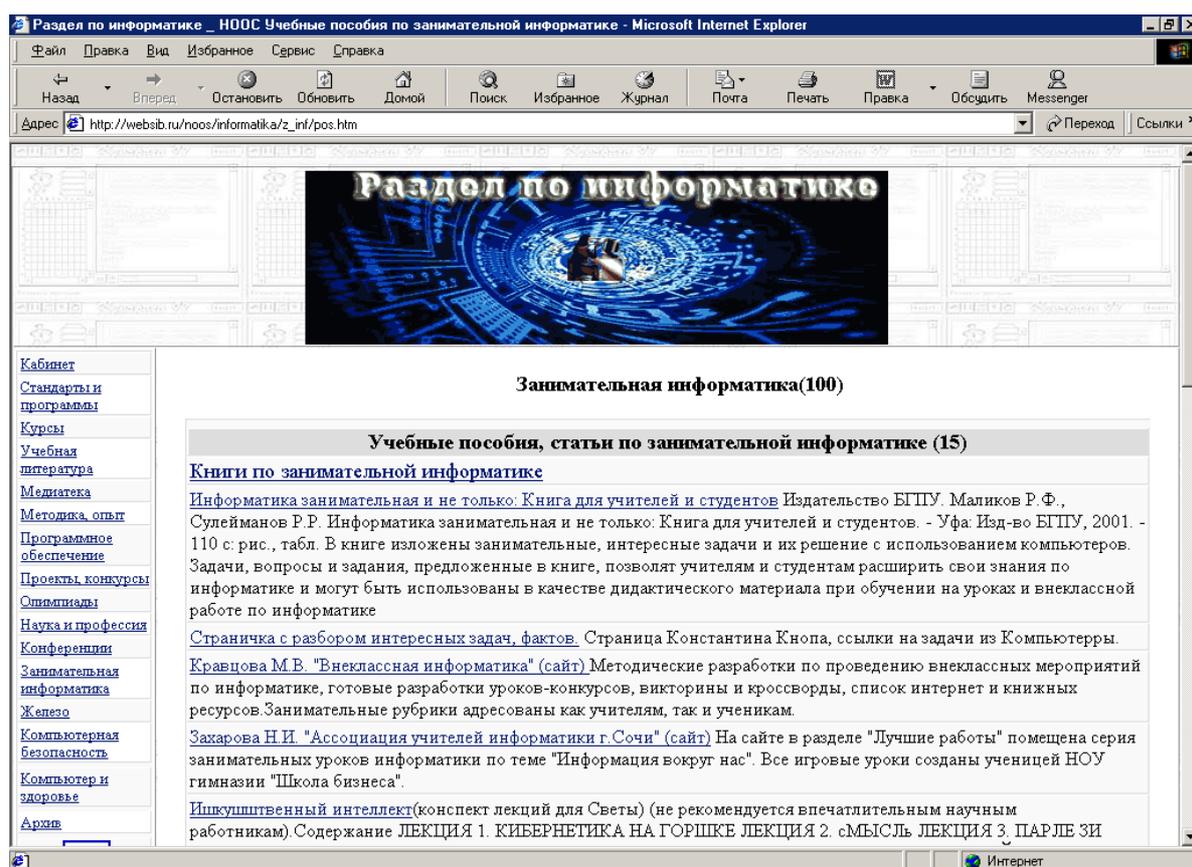


Рис. 152

При вызове ссылки Занимательная информатика авторы обнаружили, к удивлению, аннотацию своей книги «Информатика за-

нимательная и не только...» (Уфа: Изд-во БГПУ, 2001. 110 с.). Там же расположена ссылка на книгу М.В. Кравцова «Внеклассная информатика», где находятся методические разработки по проведению внеклассных мероприятий по информатике, готовые разработки уроков, викторины и кроссворды, список интернет и книжных ресурсов. Занимательные рубрики и многое другое адресованы как учителям, так и ученикам.

<http://infoschool.narod.ru/>

Название сайта «Информатика в школе» (рис.153). Главная страница сайта содержит следующие ссылки: Информационные технологии; Интернет-технологии; Makromedia Flash; Учебник по Photoshop; Учебник по HTML; Тематическое планирование; Материалы к урокам; Программное обеспечение.

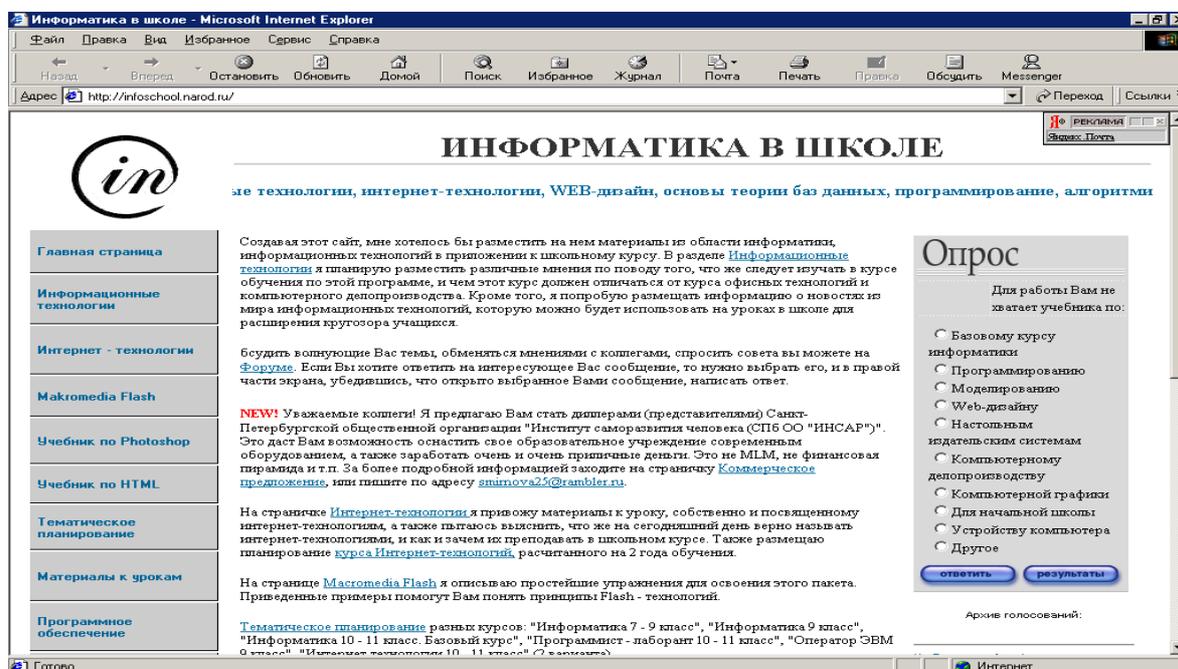


Рис. 153

На главной странице также размещен опросник о нехватке учебников по базовому курсу информатики, программированию, моделированию, Web-дизайну, настольным издательским системам, компьютерному делопроизводству, для начальной школы, устройству компьютера.

<http://www.bitpro.ru/>

Сайт «Бит про» (рис. 154) посвящен учебным компьютерным программам. Приведены каталоги программных продуктов по предметам: информатика, физика, химия, естествознание, математика, русский язык, иностранные языки, энциклопедии и словари, отечественная история, экономика, система контроля знаний, в помощь администрации. Можно посмотреть демонстрационные версии.

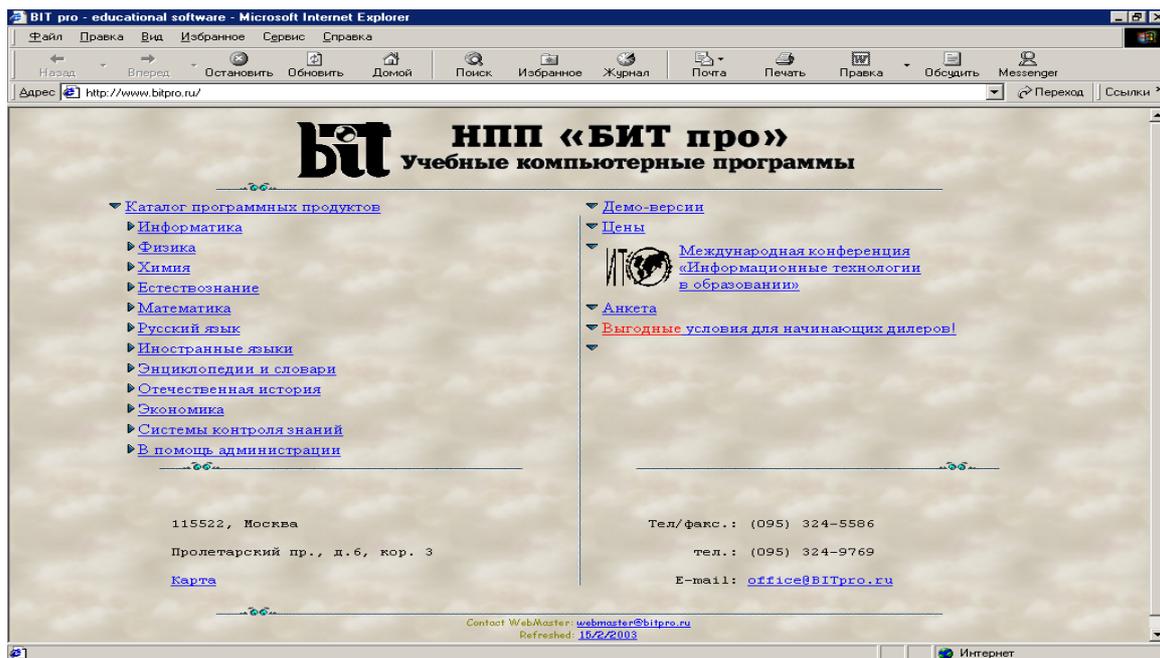


Рис. 154

<http://virlib.eunnet.net/mif/>

Электронный журнал «МИФ» (аббревиатура от слов: математика, информатика, физика). Журнал по математике, информатике и физике для школьников. Школьники в этом сайте найдут много полезной информации, в том числе по олимпиадам, о турнирах, экзаменах и т.д.

<http://www.fizika.ru/>

Название сайта Физика.ru (рис. 155). Сайт предназначен для учащихся и преподавателей физики. На главной странице расписаны ссылки: Теория; Дидактические задания; Исторический материал; Политехнический; Факультатив; Тестирование; Планирование уроков; Дистанционный урок; Полезные ссылки. Есть страницы, обращенные к преподавателям, учащимся, родителям.

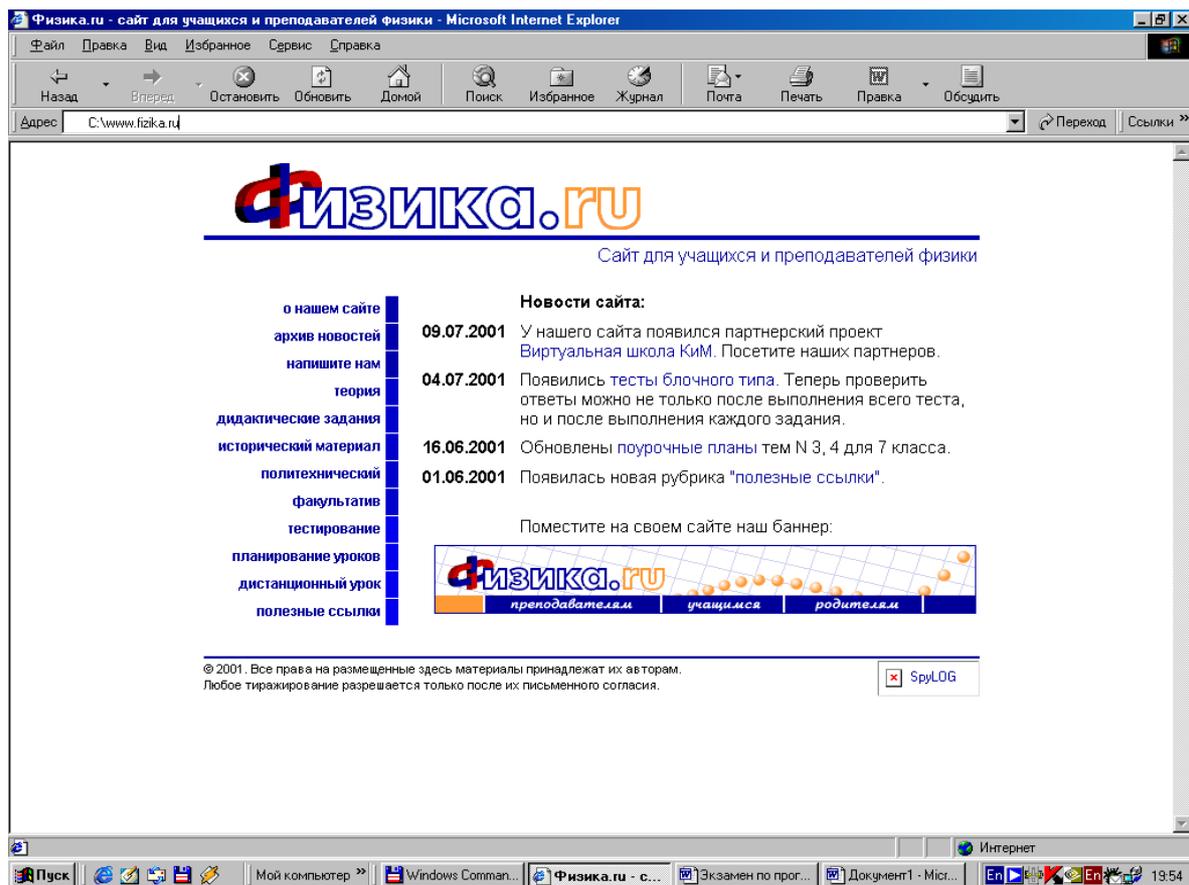


Рис. 155

К примеру: вызов ссылки Политехнический вызывает страницы об измерительных приборах, о промышленном оборудовании, бытовой технике.

<http://www.spin.nw.ru/>

Сайт Санкт-Петербургского государственного университета (рис. 156). Название сайта «Физика для школ через интернет» На главной странице ссылки: Что нового; Как стать студентом; Олимпиады; Ваши успехи; На ваш вопрос – наш ответ; Письма из далека; Физики всё ещё шутят; Полезные адреса; Вернисаж; и др.

<http://www.exponenta.ru/>

Образовательный математический сайт (рис.157). Название «Exponenta.ru». Ссылки на странице internet-класс, примеры, методики, банк задач, консультации & форум, конкурсы. Ссылки на пакеты программ Matlab, Mathcad, Maple, Mathematica, Statistica. И многое другое.

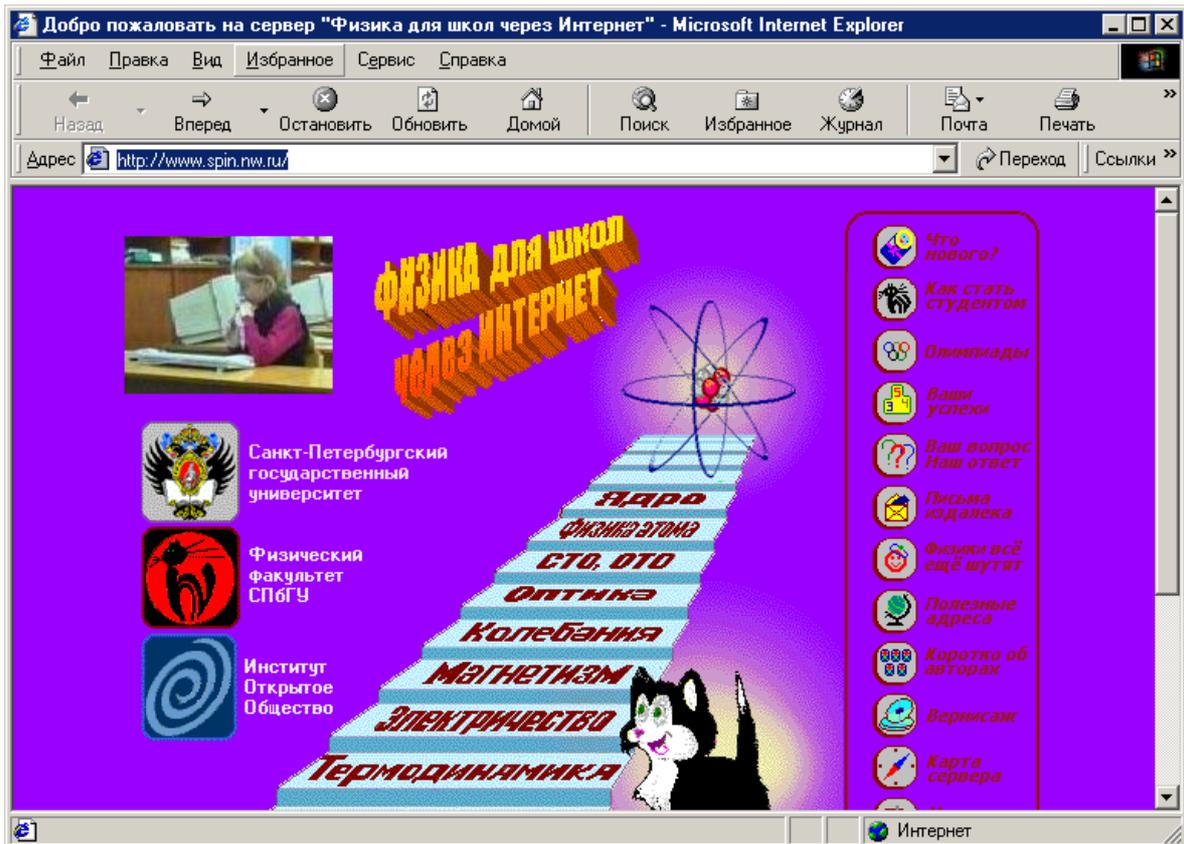


Рис. 156

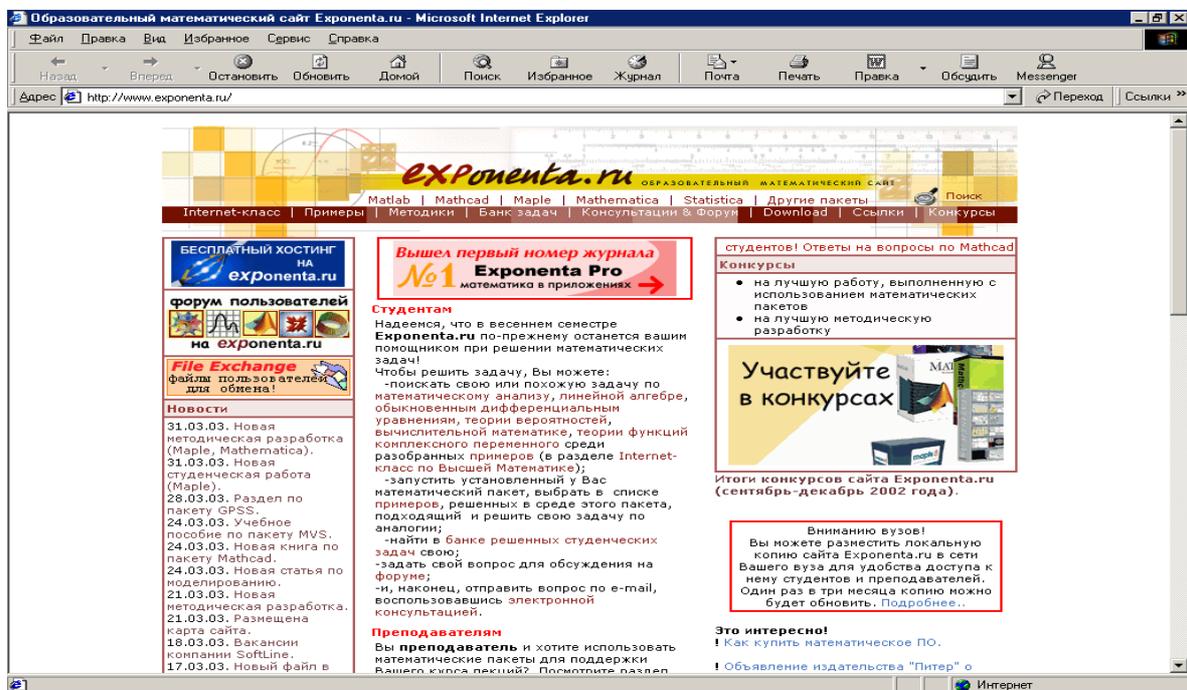


Рис. 157

<http://www.physicon.ru/>

Образовательный сайт систем дистанционного обучения и программного обеспечения. Название «Физикон». На главной странице ссылки по предметам: математика, физика, химия, астрономия, экономика, статистика, биология. При вызове ссылки математика (рис. 158) выпадает страница, в которой можно выбрать темы: Операции с целыми числами; Дроби; Операции с дробями; Пропорции; Проценты; Рациональные числа; Сокращение выражений; и др.

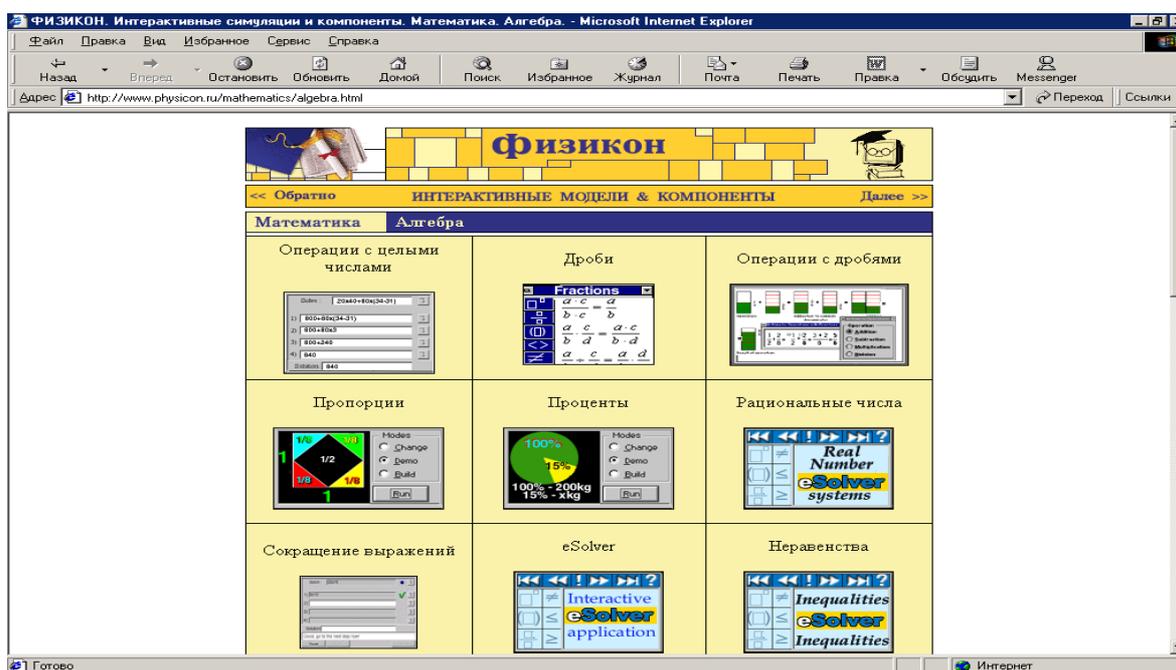


Рис. 158

Сайт <http://www.informika.ru>.

Название – chemy. Сайт содержит текстовые и графические материалы, размещенные во 2-м издании CDROM «Химия для ВСЕХ», выпущенным РНПО «РОСУЧПРИБОР» совместно с ведущими разработчиками обучающего и программного обеспечения, а также материалы, не вошедшие по тем или иным основаниям в данное издание. Включает в себя учебники по общей, органической и неорганической химии. Внешний вид сайта показан на рис. 159.

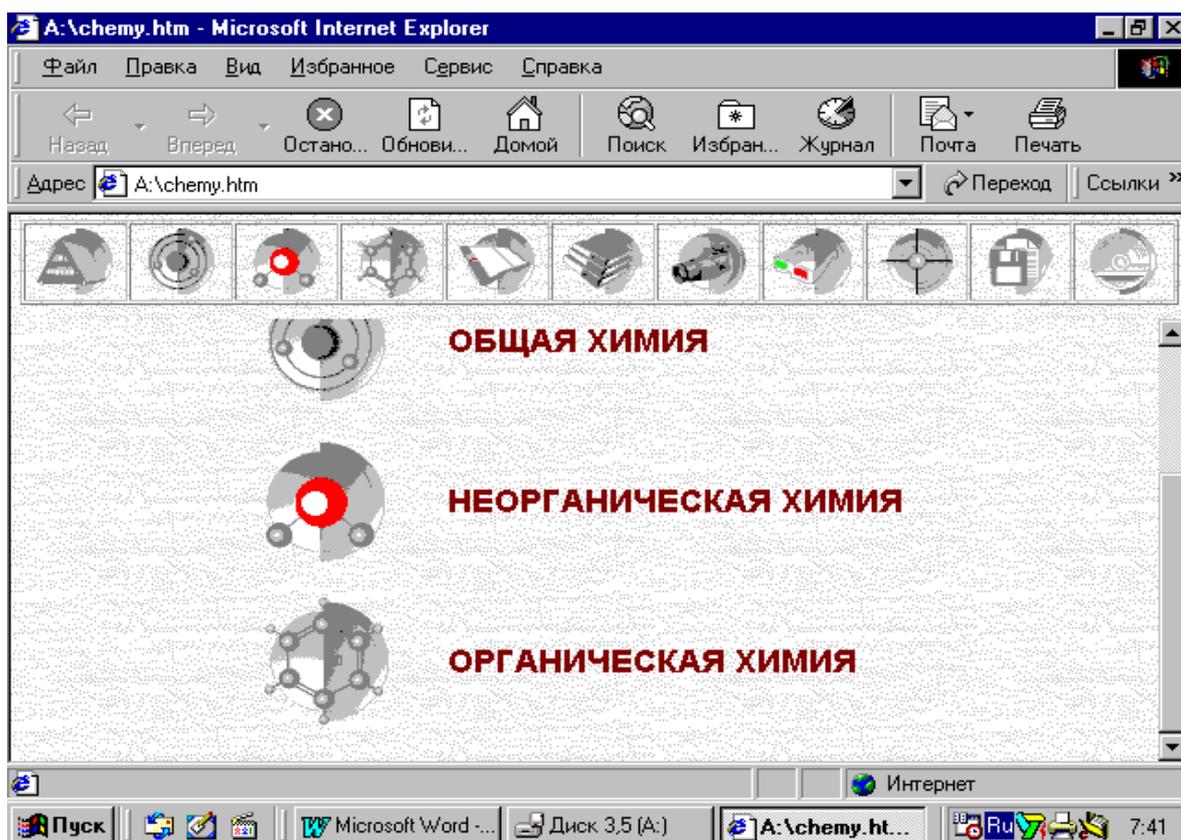


Рис. 159

Сайт <http://www.schoolchemisty.ru>.

Название – Школьная химия – справочник по химии и активная помощь ученику или студенту. Имеются в сайте следующие ссылки: Таблица Менделеева, Вещества, Повседневная химия, Ученые-химики, Щит, Химкалькалькулятор, Программы. Расшифруем содержание некоторых из них. Таблица Менделеева – содержит обширную информацию по химическим элементам. Химкалькулятор – помощник в решении химических задач. Программы – полезны тем, кто увлекается химией. Щит – шпаргалки, рефераты.

Адрес: <mailto:usmanova@ngs.ru>.

Название – Химический раздел. Содержит следующие ссылки: *Программы* – государственные и авторские программы по химии; *Органическая химия* – ресурсы по органической химии; *ХимSoft* – программы, которые можно скачать и установить у себя на компьютере; *Учительская* – преподавание химии, уроки, методики; *Из истории* – развитие химии и ученые-химики; *Это интересно* – интересные и забавные истории; *Советы химикам* – советы, правила техники безопасности, интересные опыты; *Химия и жизнь* – применение химии в повседневной жизни;

жизнь – применение химии в повседневной жизни; Абитуриенту; Книги, журналы, статьи; Химические ресурсы.

Сайт <http://alhimik.ru>.

Название – ALHIMIK. Содержит следующие ссылки: Абитура; Учительская – рассказы, где учат на химика, как сдавать экзамен, знакомства с виртуальным репетитором, новинки книжного рынка, учебники; Студиозус – кунсткамера – диковинки химического мира, читальный зал; Веселая химия; Детская; Химия на каждый день; Химические новости; Химическая Всячина; Кунсткамера; Справочник; Карта сайта.

Сайт <http://www.chemistry.ssu.samara.ru>.

Название – Химический каталог – химия – справочники – пособия – рефераты – статьи – справочная информация – книги по химии. Сайт содержит каталоги фирм, справочники, учебные занятия, разное, доски объявлений, работа, химический софт, продажа/покупка. Имеется достаточное количество ссылок. Например: Школьная химия: справочник по химии для школьников и студентов – создан в помощь ученикам и студентам, содержит статьи, шпаргалки, рефераты, справочную литературу и др. Органическая химия – Электронный учебник по органической химии для средней школы. Общий объем web-версии учебника 10,9 МВ, количество графических иллюстраций – около 500, анимации – 60, виртуальных моделей и анимации (в формате VRML) – более 30, интерактивных flash-иллюстраций – около 20, контрольных вопросов и задач – 150. Содержание книги соответствует образовательному стандарту средней школы, а по глубине и обоснованию основных положений органической химии выходит за его рамки.

Литература

1. *Алехин В.* Делим с любой точностью // Информатика и образование. 1992. № 1.
2. *Болтянский В.Г.* Программы перебора // Квант. 1988. № 1.
3. *Будно А.Л., Каплан Л.И.* Московские олимпиады по программированию. М: Наука, 1990. 208 с.
4. *Бурсиан Э.В.* 100 задач по физике для компьютера. М.: Просвещение, 1997. 245 с.
5. *Бурсиан Э.В.* Задачи по физике для компьютера. М.: Просвещение, 1991. 256с.
6. *Извозчиков В.А., Слуцкий А.М.* Решение задач по физике на компьютере. М.: Просвещение, 1999. 255 с.
7. *Говорухин В.Н., Цибулин В.Г.* Введение в Maple. Математический пакет для всех. М.: Мир, 1997. 208 с.
8. *Годунов С.К., Рябенский В.С.* Введение в теорию разностных схем. М.: Физматгиз, 1962. 340 с.
9. *Гульд Х., Тобочник Я.* Компьютерное моделирование в физике. Ч. I. М.: Мир, 1990. 350с.
10. *Доценко А.В.* Применение дифференциальных уравнений для математического моделирования реальных процессов. Учебное пособие к спецкурсу. Ленинград: Изд-во ЛГПИ им. А.И.Герцена, 1986.
11. *Дьяконов В.* Maple 6: учебный курс. СПб.: Питер, 2001. 608 с.
12. *Дьяконов В.П., Абраменкова И.Р.* Mathcad 7 в математике, физике и Интернете. М.: Нолидж, 1999. 346 с.
13. *Дьяченко В.Ф.* Основные понятия вычислительной математики. М.: Наука, 1977. 128 с.
14. *Ерохина Р., Степанова Е.* ПМК на уроках астрономии // Информатика и образование. 1989. №1.

15. *Залогова Л., Плаксин М., Русаков С. и др.* Информатика. Задачник – практикум. Т.2 / Под ред. И Семакина, Е.Ханнера. М.: Лаборатория Базовых Знаний, 2001. 278 с.
16. *Изучение основ информатики и вычислительной техники: Методическое пособие для учителей и преподавателей сред. учебных заведений. Ч.2 / А.П. Ершов, В.М. Монахов, В.М. Витиньш и др.; Под ред. А.П. Ершова, В.М. Монахова.* М.: Просвещение, 1986. 207 с.
17. *Информатика: Учебное пособие для студ. пед. вузов / А.В. Могилев, Н.И. Пак, Е.К. Хеннер; Под ред. Е.К Хеннера.* М.:, 1999. 816 с.
18. *Каймин В.А., Щеголев А.Г., Ерохина Е.А., Федюшин Д.П.* Основы информатики и вычислительной техники: пробный учебник для 10-11 классов средних школ. М.: Просвещение, 1989.
19. *Карлащук В.И.* Электронная лаборатория на IBM PC. Программа Electronics Workbench и ее применение. М.: Солон-Р, 2000. 506 с.
20. *Карлащук В.И.* Обучающие программы. М.: Солон-Р, 2001. 528 с.
21. *Касаткин В.Н.* Сколько цифр в числе 100!?! // Квант. 1988. №7.
22. *Климов Д., Петров М., Урнов В.* «Первые шаги» и «Большие проекты» // Информатика и образование. 1988. №4.
23. *Компьютеры и нелинейные явления.* М.: Наука, 1988. 192 с.
24. *Компьютеры, модели, вычислительный эксперимент.* М.: Наука, 1988. 430 с.
25. *Кордемский Б.А.* Математическая смекалка. М.: Физмат. лит-ра, 1958. С.255–260.
26. *Любарский Г.Я.* Математика в эксперименте. М.: Знание, 1983. 62 с.
27. *Лютикас В.С.* Школьнику о теории вероятности. М.: Просвещение, 1976.
28. *Мак-Кракен Д., Дорн У.* Численные методы и программирование на Фортране. М.: Мир, 1977. 583 с.
29. *Маликов Р.Ф., Сулейманов Р.Р.* Информатика занимательная и не только... Уфа: Изд-во БашГПУ, 2001. 112 с.
30. *Маликов Р.Ф., Саитов Р.К.* Практикум по компьютерному моделированию физических явлений и объектов. Уфа: Изд-во БашГПУ, 2002. 60 с.

31. *Маликов Р.Ф., Мустафин Р.Х.* Компьютерное моделирование физических явлений и объектов. Уфа: Изд-во БашГПУ, 2003. 80 с.
32. *Матвеев Н.М., Доценко А.В.* Математическое моделирование реальных процессов. Л.: Знание, 1985. 32 с.
33. *Мячев А.А.* Персональные ЭВМ: Краткий энциклопедический справочник. М.: Финансы и статистика, 1992. 384 с.
34. *Новожиллов Б.В.* Метод Монте-Карло. М.: Знание, 1966.
35. *Носков Н.Н., Столбоушкин С.К.* Спутник на дисплее // Квант. 1989. №10.
36. *Основы информатики и вычислительной техники: Пробное учебное пособие для средних учебных заведений. Ч.2 / А.П. Ершов, В.М. Монахов, А.А. Кузнецов и др.; Под ред. А.П. Ершова, В.М. Монахова.* М.: Просвещение, 1986. 143 с.
37. *Основы информатики и вычислительной техники: Пробный учебник для средних учебных заведений / А.Г.Кушниренко, Г.В.Лебедев, Р.Я.Сворень.* М.: Просвещение, 1990.
38. *Педагогико-эргономические условия безопасного и эффективного использования средств вычислительной техники, информатизации и коммуникации в сфере общего среднего образования // Информатика и образование. 2000. № 4, 5, 6.*
39. *Пирумов У.Г.* Численные методы: Учебное пособие. М.: Изд-во МАИ, 1998. 188 с.
40. *Плис А.И., Сливина Н.А.* Mathcad: математический практикум. М.: Финансы и статистика, 1999. 656 с.
41. *Покосовская О.В.* Решение задач из теории чисел // Информатика и образование. 1996. №6; 1997. №1,2.
42. *Практикум по математическому моделированию физических явлений и процессов / Сост. Р.Ф. Маликов.* Уфа: Изд-во БГПИ, 2000. 56 с.
43. *Практикум по изучению основ электроники и вычислительной техники в Electronics Workbench / Сост. Р.Р. Сулейманов, Р.Ф.Маликов.* Уфа: Изд-во БГПУ, 2003. 30 с.
44. *Ракитин В.И., Первушин В.Е.* Практическое руководство по методам вычислений с приложением программ для персональных компьютеров. М.: Высшая школа, 1998. 383 с.
45. *Ремеев И.С.* Математическое моделирование физических процессов: Учебное пособие. Уфа: Изд-во Баш. ун-та. 1966. 72 с.

46. Рыжиков Ю.И. Решение научно-технических задач на персональном компьютере. СПб.: КОРОНА принт, 2000. 272 с.
47. Саитов Р.И. Новые информационные технологии в обучении физике в средней школе. Уфа: Изд-во БГПИ, 1995. 68 с.
48. Самарский А.А., Михайлов А.П. Математическое моделирование. М.: Наука. 1997. 316 с.
49. Сахабутдинов Ж.М. Анализ дискретных моделей движения точки / ИММ РАН. Казань, 1995. 196 с.
50. Сорокин В.А., Илларионов В.В. Основы программирования: Учебное пособие по курсу «Основы информатики». Уфа: Изд-во БГПИ. 1996. 188 с.
51. Сулейманов Р.Р. Занятный алгоритм // Учитель Башкортостана. 2000. №10.
52. Сулейманов Р.Р. Литерные величины // Информатика. 1996. № 32 .
53. Сулейманов Р.Р. Ребусы // Информатика. 1997. № 23.
54. Сулейманов Р.Р. Удивительные числа // Информатика и образование. 2000. № 9.
55. Сулейманов Р.Р. Числовая загадка цифровых клавиш // Учитель Башкортостана. 2001. №3.
56. Сулейманов Р.Р. Вариации на тему «Простые числа» // Информатика. 1997. №3.
57. Сулейманов Р.Р. Задачи о кросс-суммах // Информатика и образование. 1998. № 6.
58. Сулейманов Р.Р. Задачи перестановки с проверкой условий или магические фигуры // Информатика. 1997. № 20.
59. Сулейманов Р.Р. Интегрированный урок « Приближенное вычисление интеграла» // Учитель Башкортостана. 1998. № 2.
60. Сулейманов Р.Р. Использование электронных таблиц в графическом представлении числовой информации // Учитель Башкортостана. 1995. №6.
61. Сулейманов Р.Р. Исследование периодических дробей // Информатика и образование. 2001. № 7.
62. Сулейманов Р.Р. КВН в кабинете информатики // Информатика и образование. 1999. №3.
63. Сулейманов Р.Р. Лабораторно-практическая работа по теме «Вычисление значения Π » // Информатика. 1998. № 21.
64. Сулейманов Р.Р. Математические задачи на уроках информатики // Информатика и образование. 1999. № 6.

65. Сулейманов Р.Р. Признаки делимости в двоичной системе // Информатика и образование. 2001. № 9.
66. Сулейманов Р.Р. Расчет на ЭВМ ожидаемой точности изготовления детали // Учитель Башкортостана. 1999. № 7.
67. Сулейманов Р.Р. Решение математических задач в системе MathCAD 8.1. // Учитель Башкортостана. 2002. № 2.
68. Сулейманов Р.Р. Семинар на тему «Уравнение» // Учитель Башкортостана. 1998. № 7.
69. Сулейманов Р.Р. Симметричная сумма // Информатика и образование. 2000. № 4.
70. Сулейманов Р.Р. Составление задач учащимися // Информатика и образование. 2000. № 6.
71. Сулейманов Р.Р. Электронная таблица Excel как инструмент решения физических задач // Учитель Башкортостана. 2002. № 5.
72. Сулейманов Р.Р., Маликов Р.Ф. Ключевые программы для изучения языка Бейсик. Уфа: Изд-во БГПИ, 1998. 60 с.
73. Урмаев А.С. Практикум по моделированию на АВМ. М.: Наука, 1976. 192 с.
74. Ходяков И.А. Mathcad 6.0 и Electronics Workbench 5.12 в средней школе // Информатика и образование. 1999. № 6, 7, 9.
75. Шахмаев Н.М., Шодиев Д.Ш. Физика: Учеб. за 11 класс средней школы. М.: Просвещение, 1991.
76. Щербинин С.А. Введение в численные методы решения дифференциальных уравнений: Учебное пособие. Уфа: Изд-во Башкирского ун-та, 1999. 151 с.

Рамиль Фарукович Маликов
Ринат Рамилевич Сулейманов

ИНФОРМАТИКА КЛАССНАЯ И ВНЕКЛАССНАЯ

Ответственный за выпуск *Ф.С. Тикеев*
Главный редактор *Р.М. Габдуллина*
Технический редактор *Л.М. Сюндюкова*
Редактор *Е.Р. Малая*
Компьютерная верстка *Ю.В. Федоровой*

Лицензия ИД № 05001 от 07 июня 2001 г.
Подписано в печать 26.12.03.
Формат 60x84¹/₁₆. Бумага офсетная.
Гарнитура «Таймс». Печать на ризографе.
Усл. печ. л. 21,39. Уч.-изд. л. 22,14.
Тираж 1 000 экз. (1-й завод – 500 экз.) Заказ №

Издательство «Гилем».
450077, г.Уфа, ул. Кирова, 15
Тел.: 23-05-93, 22-36-82



Отпечатано на оборудовании
ООО «ДизайнПолиграфСервис»
450005, г. Уфа, ул. Кирова, 65, оф. 102,
тел.: (3472) 52-70-88