ИНФОРМАЦИОННАЯ КУЛЬТУРА

МОДУЛЬ

КЛАСС

9

МЕТОДИЧЕСКОЕ ПОСОБИЕ

■

ПОД РЕДАКЦИЕЙ
А.Г.КУШНИРЕНКО, М.Г.ЭПИКТЕТОВА

МОСКВА
ИЗДАТЕЛЬСКИЙ ДОМ
Д Р О Ф А
1995

УДК 373:002
ББК 74.261.6
И74

Материалы книги разработаны и подготовлены в объединении ИнфоМир

Авторы:
А. Г. Кушниренко, А. Г. Леонов, М. Г. Эпиктетов,
В. В. Борисенко, М. А. Кузьменко,
Б. А. Назаров, С. Б. Ханжин

Информационная культура: Модуль: Класс 9:
И74 Методическое пособие / А. Г. Кушниренко, А. Г. Леонов, М. Г. Эпиктетов и др. — М.: Дрофа, 1995.— 112 с.

ISBN 5—7107—0593—4

В книге представлены поурочные методические рекомендации для учителей, которые ведут уроки в девятом классе общеобразовательной школы по новому школьному курсу «Информационная культура».

Курс, имеющий в первую очередь образовательное, общеразвивающее значение, охватывает 11 учебных лет. Предусмотрено несколько «точек входа», в которых можно подключиться к изучению курса «Информационная культура»; 9-й класс — одна из них.

ББК 74.261.6

Введение

Настоящее пособие для учителя содержит примерное поурочное планирование и методические рекомендации по модулю 9 («Кодирование информации») курса «Информационная культура». Не весь материал из книги для ученика должен быть пройден на уроках. Необязательные, с точки зрения авторов, пункты помечены в книге звездочками. В поурочном планировании намечен выбор материалов для каждого урока, однако окончательный выбор остается за учителем.

Основные тенденции нынешнего этапа информатизации современного общества

Эти тенденции могут быть, на наш взгляд, сформулированы так:

1) возникают мировые информационные сети (конгломерат сетей), охватывающие большинство рабочих мест и домашних хозяйств;

2) вся накопленная человечеством к концу XX века информация переводится в компьютерное представление и поступает на бессрочное хранение в мировые информационные сети;

3) вся вновь генерируемая человечеством информация производится с помощью компьютеров и технически может быть сделана доступной на планетарном уровне;

4) процессы 1)—3) развиваются децентрализованно, что приводит к возникновению и конкуренции различных подходов, технологий, фирм, стандартов, протоколов и т.д.;

5) полноценный член общества XXI века должен будет каждодневно и эффективно взаимодейство-

вать с информационными сетями от локального до планетарного уровня.

Тем самым

6) объем доступной гражданину XXI века информации увеличится на много порядков;

7) формы представления и методы доступа к информации неизбежно будут отличаться большим разнообразием;

8) умение работать с информационными инфраструктурами будет во многом определять социальный статус индивидуума;

9) такое умение придется поддерживать и обновлять на протяжении всей жизни.

Мы попытались отразить эти новые тенденции в модулях 9—10 курса «Информационная культура».

От курса информатики к курсу информационной культуры

По нашему мнению, основные цели изучения алгоритмизации в курсе «Основы информатики» состоят в том, чтобы:

- сформировать у учащегося интуитивное понятие алгоритма;
- продемонстрировать основное свойство компьютера — возможность быстрого и безошибочного выполнения алгоритмов;
- ознакомить практически с процедурой составления и отладки простейших алгоритмов в какой-нибудь системе программирования.

Изучение способов кодирования и обработки информации неизбежно должно опираться на понятие алгоритма. Однако по существующей школьной программе курс «Основы информатики» начинается

только в 10 классе. Поэтому в начало нашего курса добавлена глава «Повторим программирование», которая позволит ученикам освоить необходимые понятия и приобрести достаточные навыки.

Вместе с тем, по нашему мнению, в разделах курса информатики, посвященных алгоритмизации, собственно понятие информации оказывается вторичным, вспомогательным: выполняемый на ЭВМ «его величество Алгоритм» обрабатывает Информацию, и главным героем этого процесса, безусловно, выступает **алгоритм**. За бортом остаются вопросы о том, откуда берется обрабатываемая информация, почему она представлена в той или иной форме, какие свойства объектов реального мира она отражает, почему выбраны именно эти свойства и т.д.

Но именно эти вопросы будут выступать на первый план при работе человека XXI века с информационными структурами, при освоении новых методов и инструментов для получения и обработки информации. Поэтому мы постарались сместить акценты с понятия «алгоритм» на понятия собственно информации, кодирования информации, передачи информации и построения информационных моделей окружающих нас процессов.

Роль языка программирования и системы программирования

В модулях 9—10 язык программирования не является самостоятельным объектом изучения, а выступает как нотация, система обозначений, с помощью которой ведется обсуждение проблем и их решений. Одновременно язык программирования выступает как инструмент, обеспечивающий практическую проверку получаемых теоретических решений.

Без языка программирования не обойтись, какой-то язык необходимо выбрать.

Выбор школьного алгоритмического языка в качестве базисного языка программирования не существен. Его можно было бы заменить современным Бейсиком (в котором есть процедуры с параметрами и локальные переменные) или — в хорошо подготовленных классах — Паскалем.

Существенным является то, что школьный алгоритмический язык погружен в гипертекстовую систему программирования ***КуМир-Гипертекст***.

Это позволило снабдить большинство тем курса специально подготовленными гипертекстами. Гипертексты предназначены как для организации активной работы учащихся, т.е. для ответов на контрольные вопросы и выполнения других заданий, так и для подачи нового материала. Некоторые общие сведения о системе КуМир-Гипертекст приведены в приложении. Учителю рекомендуется просмотреть это приложение заранее, чтобы быть готовым к возможным вопросам учеников.

Несколько слов о гипертекстах к курсу

Напомним, что ***гипертекст*** — это обычный текст, который можно читать на экране. Однако, если установить курсор в одну из предусмотренных автором гипертекста позиций и нажать левую кнопку мыши (или клавишу Enter), то выполнится действие, связанное с этой позицией: переход в другое место гипертекста или запуск определенной программы. Такие программы могут выводить на экран картинки или мультфильмы, проигрывать аудио- или видеоклипы, задавать контрольные вопросы и анализировать ответы.

Структура гипертекста к каждой главе

Гипертекст состоит из страниц, каждая из них целиком умещается на экране, и начинается с ***головной страницы***, содержащей перечень тем. Если поставить курсор на название темы и нажать клавишу Enter, то на экране появится страница с перечнем заданий выбранной темы.

Перед названием каждого задания стоит одинарная или двойная стрелка. Чтобы начать работу с заданием, нужно поставить курсор на эту стрелочку и нажать Enter. Одинарными стрелками обычно помечены задания, где степень самостоятельности ученика невелика: просмотр справочного материала, мультфильма, демонстрации. Двойными стрелками помечены задания, где активность и самостоятельность ученика выше: ответ на контрольный вопрос, исправление ошибки в программе или ее данных, дописывание программы, самостоятельное составление программы.

В Приложение 3 вынесено подробное описание гипертекста к главе 1 вместе с методикой его использования. Это описание позволяет делать поурочные методические комментарии существенно более компактными. Поэтому, несмотря на то что рассказ о гипертексте помещен в приложение, учителю очень важно с ним внимательно ознакомиться. Приложение 3 — типовое представление гипертекста: гипертексты по всем остальным главам устроены аналогично.

Запуск гипертекста

Процедура установки системы КуМир-Гипертекст описана в документации по системе, поставляемой на диске. После установки системы для запуска ги-

пертекста к главе 1 нужно набрать команду

```
> НС1
```

и на экране появится головная страница гипертекста. Аналогично, для запуска гипертекста к главе 2 нужно набрать команду

```
> НС2
```

и т.д. (обратите внимание на отсутствие пробела между НС и цифрой в командной строке).

Для выхода из системы надо:

- закончить выполнение задания, нажав (в зависимости от типа задания) Break или Esc End;
- вернуться в головную страницу клавишей Esc;
- еще раз нажать Esc для выхода в операционную систему.

Замечание. Командные последовательности вида Esc End вводятся двумя *последовательными* нажатиями: сначала надо нажать клавишу Esc, отпустить ее, а потом нажать End.

Глава 1. ПОВТОРИМ ПРОГРАММИРОВАНИЕ

Материал рассчитан на 10 уроков. За это время школьники должны:

а) изучить или вспомнить основные понятия из курса информатики: алгоритм, величина, ветвление, повторение, присваивание;

б) научиться выражать эти понятия на школьном алгоритмическом языке;

в) научиться основным приемам работы в системе КуМир-Гипертекст.

На первый взгляд кажется, что выполнение б) и в) потребует запоминания большого количества справочной информации и на это уйдет много времени и сил. К счастью, это не так.

Во-первых, система КуМир включает развитую гипертекстовую справочную систему и меню для выполнения основных команд. Так что запоминать придется только то, что нужно каждую секунду, а все остальное можно будет посмотреть по ходу дела.

Во-вторых, все конструкции алгоритмического языка в КуМире вводятся в программу «в готовом виде» по нажатии всего двух клавиш. Запоминать в деталях формат этих конструкций не обязательно.

В-третьих, если при наборе программы допущена ошибка, КуМир мгновенно отмечает в тексте программы место, где она сделана, а на полях указывает возможную причину ошибки. Если что-то забылось, то можно попробовать несколько вариантов, пока не вспомнится верный.

Наконец, в гипертекстах по курсу много демонстраций, просмотр которых позволяет мгновенно уловить основные идеи тех или иных команд.

Глава 1 учебника методически ориентирована на использование только совместно с гипертекстом.

Урок 1

Тема урока: Программа — план будущих работ компьютера.

Программное обеспечение: Страница *Что такое гипертекст* из гипертекста к главе 1 (НС1).

Цель урока: Ознакомление с общими понятиями *программа* и *гипертекст*.

План урока:

1. Программы в окружающем мире.
2. Языки программирования.
3. Определение и примеры гипертекстов.
4. Практическая работа с гипертекстом.

Домашнее задание.

1. Программы в окружающем мире.

Первый урок следует начать с обсуждения материалов введения (пункты 1.1—1.4). Составление планов будущих работ вовсе не обязательно связано с компьютерами. Разного рода инструкции и расписания и есть программы будущих работ, адресованные человеку. Ученики без труда приведут примеры планов будущих работ. Такими примерами могут быть и расписание уроков ученика данного класса, и инструкция, как себя вести при возникновении пожара. Программа для человека должна быть написана на известном ему языке. Ученики согласятся с тем, что расписание уроков на китайском языке было бы для них совершенно бесполезно. Программа для человека может быть написана не очень формально и не очень подробно. Человек сообразителен и способен сам дополнить недостающие детали.

2. Языки программирования.

Компьютер не обладает такой сообразительностью, потому планирование работ компьютера требует специального формализованного способа записи. Разумеется, этот способ записи должен быть известен компьютеру. Формализованные системы выдачи инструкций и приказов встречаются и в человеческой практике. Попросите учеников назвать такие системы. Могут быть названы: программы вязания на спицах, нотная азбука, уставные обращения в армии, инструкции к бытовой электронике, в которой действия расписаны по шагам, и т.д.

3. Определение и примеры гипертекстов.

Важным инструментальным средством в главе 1 (как, впрочем, и во всем модуле 9 класса) является ***гипертекст***. На уроке следует обсудить основные принципы устройства гипертекстов и особенности реализации гипертекста в системе КуМир (по материалам пункта 1.7 книги для ученика и Приложения 2 к книге для учителя). Несколько минут урока можно посвятить обсуждению самого слова *гипертекст*.

> По одной из версий, этот термин навеян научной фантастикой на космические темы. Фантасты давно уже используют ***гиперпереход*** космического корабля в ***гиперпространстве*** (многомерном пространстве) для того, чтобы мгновенно отправить своих героев в нужное место. Точно так же и в ***гипертексте***, поставив курсор на выделенное поле и нажав Enter, можно мгновенно попасть в нужное место текста.

4. Практическая работа с гипертекстом.

Учитель должен напомнить (или объяснить) назначение клавиш Enter (выбор поля) и Esc (возврат к предыдущей странице) и запустить гипертекст к главе 1. Учащиеся должны поработать со страницей *Что такое гипертекст*, но могут заглянуть и на другие страницы.

Домашнее задание.

Пункт 1.5 рекомендуется задать на дом для самостоятельного изучения и кратко обсудить его в начале следующего урока.

Урок 2

Тема урока: Управление исполнителем по программе.

Программное обеспечение: Страница *Знакомство с исполнителем Робот* гипертекста НС1.

Цель урока: Формирование у школьников представления о схемах управления исполнителями.

План урока:

1. Обсуждение домашнего задания.
2. Представление об исполнителе.
3. Схемы управления исполнителем.

Домашнее задание.

1. Обсуждение домашнего задания.

Ученики должны вынести из заданного на дом материала не фактографические данные об истории языков программирования, а фундаментальную идею формализации задания компьютеру. Вместе с тем важно, чтобы школьники обратили внимание на эволюцию средств общения человека с компьютером от систем кодирования, удобных для компьютера, к языковым системам, удобным для человека.

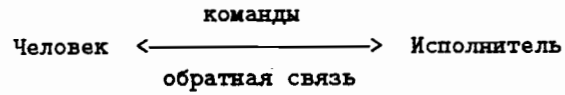
2. Представление об исполнителе.

Поскольку схемы управления исполнителем отрабатываются на примерах исполнителей Робот и Чертежник (которые вводятся в гипертексте к главе 1), перед их обсуждением полезно несколько минут поработать со страницей *Знакомство с исполнителем Робот*.

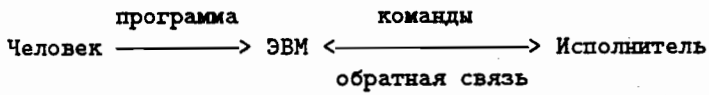
Для школьников, включающихся в курс «Информационная культура» в 9 классе, понятие исполнителя появляется впервые, и ему стоит уделить достаточно внимания, чтобы показать фундаментальность этого понятия и многообразие мира исполнителей.

3. Схемы управления исполнителем.

Далее следует обсудить схему ручного управления исполнителем



и ее отличие от схемы автоматического управления исполнителем по заданной программе

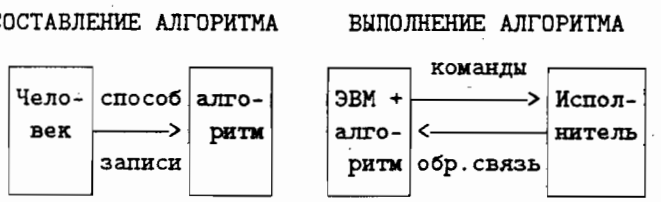


В ситуации ручного управления Человек нажимает на кнопку, отдавая команду, а Робот или другое устройство (*Исполнитель* в терминологии учебника) эту команду выполняет. Разумеется, какая-то информация должна поступать и от Исполнителя к Человеку, это тоже делается по специальным командам управления. Итак, в этой схеме все просто: Человек нажимает на кнопку — Исполнитель выполняет действие. Весь глобальный план действий Человек может не помнить в деталях, а придумывать по ходу дела, после выполнения каждой команды.

Полезно обсудить, почему вторая схема сложнее первой. Во-первых, кроме двух персонажей (Человек и Исполнитель) появляется новый — ЭВМ. Во-вторых, алгоритм придется продумать во всех деталях, никакую из них нельзя отложить на потом — ведь выполнять-то будем уже не мы сами, а ЭВМ.

В-третьих, с ЭВМ нужно будет как-то договориться о правилах записи алгоритма, чтобы он был понят.

Схема «Человек—ЭВМ—Исполнитель» описывает два разных этапа работы:



Домашнее задание.

Прочитать пункт 1.6 из § 1 и введение к § 2.

Урок 3

Тема урока: Знакомство с исполнителем Робот.

Программное обеспечение: Страницы *Исполнитель Робот* и *Составление простейших алгоритмов* гипертекста НС1.

Цель урока: Формирование навыков записи простейших алгоритмов.

План урока:

1. Обсуждение домашнего задания.
2. Просмотр демонстраций в гипертексте.
3. Практическая работа по составлению простейших алгоритмов.
4. Запись алгоритмов в других языках.

1. Обсуждение домашнего задания.

Предметом обсуждения в начале урока является введение к § 2. Точное определение понятия ***программа*** здесь не столь обязательно, но его толкование и примеры школьники должны уметь приводить.

2. Просмотр демонстраций в гипертексте.

Далее идет работа со страницами *Исполнитель Робот* и *Составление простейших алгоритмов*. Основной упор надо делать не на изучение Робота (который слишком прост и может показаться школьникам скучным), а на правила записи алгоритмов и методику работы в КуМире.

Подробно гипертекстовые демонстрации и упражнения описаны в Приложении 3, и здесь мы не будем на них останавливаться.

3. Практическая работа по составлению простейших алгоритмов.

В зависимости от уровня подготовки школьников учитель может дать от одного до трех заданий из числа приведенных в гипертексте.

4. Запись алгоритмов в других языках.

Важно подчеркнуть, что, хотя правила записи алгоритмов различаются в разных языках программирования, различия совершенно непринципиальны. В качестве иллюстрации можно предложить ученикам перевести простую программу управления Роботом с языка КуМир на Паскаль, Лого или Фортран (это можно сделать, ничего не зная об этих языках и пользуясь только приведенными примерами).

Урок 4

Тема урока: Знакомство с исполнителем Чертежник.

Программное обеспечение: Страницы *Исполнитель Чертежник* и *Вспомогательные алгоритмы* гипертекста НС1.

Цель урока: Формирование навыков использования команд с аргументами.

План урока:

1. Повторение декартовых координат на плоскости.

2. Знакомство с исполнителем Чертежник.

3. Начало изучения темы «Вспомогательные алгоритмы».

Домашнее задание.

1. Повторение декартовых координат на плоскости.

Нарисуем на доске две взаимно перпендикулярные оси координат и отложим на каждой из них единичный отрезок. Точка пересечения этих осей называется ***началом координат***. Для любой точки T первой четверти опустим перпендикуляры на координатные оси. Расстояния от оснований этих перпендикуляров до начала координат называются ***координатами*** точки T.

Для точки первой четверти обе координаты положительны (а точнее сказать, неотрицательны). Для точек остальных четвертей одна или обе координаты могут быть отрицательны.

На поле Чертежника начало координат (как правило) размещено в левом нижнем углу, поэтому все поле Чертежника расположено в первой четверти.

2. Знакомство с исполнителем Чертежник.

Далее идет работа со страницей *Исполнитель Чертежник*. Гипертекст подробно демонстрирует работу Чертежника, и единственная возможная трудность — работа команды Чертежника

```
сместиться на вектор (a,b)
```

Объяснить работу этой команды лучше на примерах. Пусть, например, перо Чертежника находится в точке с координатами (1,1), и мы даем команду

```
сместиться на вектор (2,3)
```

Тогда перо Чертежника переместится в точку с координатами $(1+2, 1+3)$, т.е. в точку $(3, 4)$. Если же

перо находится в точке $(4, 5)$ и мы дадим команду

```
сместиться на вектор (-3,2),
```

то перо окажется в точке $(4-3, 5+2)$, т.е. в точке $(1, 7)$. В общем случае, если перо было в точке (x, y), то по команде

```
сместиться на вектор (a,b)
```

оно переместится в точку $(x+\mathrm{a}, y+\mathrm{b})$.

Тем учащимся, у которых это вызовет затруднения, нужно порекомендовать поработать с заданием Р2 страницы *Исполнитель Чертежник*.

3. Начало изучения темы *Вспомогательные алгоритмы.*

В конце урока следует перейти к странице гипертекста *Вспомогательные алгоритмы*. На 4-м уроке надо рассмотреть понятие вспомогательного алгоритма и решить упражнение А4. Завершение изучения этой темы займет весь следующий урок.

Домашнее задание.

Прочитать пункты 2.3 и 2.4.

Урок 5

Тема урока: Вспомогательные алгоритмы.

Программное обеспечение: Страницы *Вспомогательные алгоритмы* гипертекста НС1.

Цель урока: Формирование навыков использования вспомогательных алгоритмов при решении задач.

План урока:

1. Обсуждение домашнего задания.
2. Просмотр демонстраций в гипертексте.
3. Практическая работа по составлению алгоритмов.

Домашнее задание.

1. Обсуждение домашнего задания.

Понятие вспомогательного алгоритма и идея относительной независимости основного и вспомогательного алгоритмов очень важны, поэтому в начале урока следует подробно обсудить прочитанные дома пункты.

2. Просмотр демонстраций в гипертексте.

Демонстрации А5 (пошаговое выполнение вспомогательного алгоритма) и А6 (работа алгоритма квадрат (вещ а)).

3. Практическая работа по составлению алгоритмов.

Центральное место уроков 4 и 5 — упражнения А7 (а)–(г), приведенные в гипертексте. Основная цель этого цикла упражнений — научить согласовывать основной и вспомогательные алгоритмы. В упражнениях А7(а) и А7(б) основной алгоритм дан заранее и требуется составить вспомогательный алгоритм. Следует обратить внимание школьников на то, что короткое описание вспомогательного алгоритма содержится в строках дано–надо. Этим описанием и нужно руководствоваться при составлении алгоритма.

В упражнении А7(в) требуется составить алгоритм обход стены, выбрав один из двух вспомогательных алгоритмов закрашивание блока N1 или закрашивание блока N2. Сами алгоритмы не даны, приведены только дано–надо. Формально задача не имеет однозначного решения, так как при любом выборе из N1 и N2 можно составить алгоритм обход стены, дающий решение задачи. С практической же точки зрения задача решается однозначно, и нужно выбрать вспомогательный алгоритм закрашивание блока N1, так как в этом случае алгоритм обход стены получается гораздо короче.

В упражнении А7(г) по тем же причинам нужно выбрать закрашивание блока N2.

У сильных школьников может возникнуть вопрос, как КуМир проверяет решение по нажатии Ctrl T. Делается это так. По информации о положении пера на поле Чертежника и нарисованной Чертежником картинке можно построить некоторое многоразрядное число, характеризующее нарисованную картинку. Для ожидаемой правильной картинки это число вычисляется заранее и запоминается. По нажатии Ctrl T выполняется программа школьника и подсчитывается число для этой картинки. Если это число не совпадает с ожидаемым, то выдается сообщение об ошибке. Если же совпадает, то считается, что программа правильна.

Теоретически для двух разных картинок при подсчете может получиться одно и то же число, так что совпадение чисел еще не гарантирует совпадения картинок. На практике же вероятность совпадения столь мала, что ею можно пренебречь.

Возможно, у учеников возникнет вопрос: «А почему бы не запомнить всю ожидаемую правильную картинку целиком и не сравнивать ее с тестируемой картинкой?» Ответ очень прост: «Понадобилось бы слишком много места на диске для хранения всех правильных ответов».

Домашнее задание.

Прочитать пункт 2.5.

Урок 6

Тема урока: Цикл «N раз».

Программное обеспечение: Страница *Цикл «N раз»* гипертекста НС1.

Цель урока: Формирование навыков записи и использования составных команд.

План урока:

1. Обсуждение домашнего задания.
2. Практическая работа в гипертексте.

Домашнее задание.

1. Обсуждение домашнего задания.

Урок рекомендуется начать с обсуждения термина ***составная команда*** и короткого обсуждения формы записи команды «цикл "N раз"» (пункт 2.5). Параллельно нужно объяснить два способа ввода составной команды в системе КуМир.

Способ 1. Если ввести в начале строки

```
нц N раз
```

и нажать Enter, то автоматически на экране появится и конец команды.

Способ 2. Если нажать последовательно клавиши Esc и затем Ц (от слова Цикл), то на экране появятся начало и конец команды «цикл»

```
нц
|
кц
```

после чего можно вписать в первую строку «N раз» (6 раз или 4 раз).

Замечание. Слова алг, нач, нц, раз и т.д. являются ***служебными***, и их нельзя склонять по правилам русского языка. В частности, нужно писать 4 раз, а не 4 раза.

2. Практическая работа в гипертексте.

Далее следует перейти к работе в гипертексте со страницей *Цикл «N раз»*. Демонстрации и упражнения подробно прокомментированы в Приложении 3.

Домашнее задание.

Прочитать пункты 2.6—2.8.

Урок 7

Тема урока: Команды ветвления и повторения.

Программное обеспечение: Страницы *Команды обратной связи у Робота*, *Цикл «пока»* и *Команда ветвления «если»* гипертекста НС1.

Цель урока: Формирование навыков записи и использования условных команд ветвления и повторения.

План урока:

1. Команды обратной связи у Робота.
2. Цикл «пока».
3. Команда ветвления «если».

Домашнее задание.

Основной принцип этого урока — поменьше объяснений, побольше самостоятельной работы в гипертексте. Ученики должны поработать с тремя страницами гипертекста: *Команды обратной связи у Робота*, *Цикл «пока»* и *Команда ветвления «если»*.

Учителю придется проконтролировать фронтально, как ученики справились с заданиями С3 «Поиск закрашенной клетки» и D2 «Разметка коридора».

Домашнее задание.

Прочитать пункт 2.9.

Урок 8

Тема урока: Величины.

Программное обеспечение: Страница *Величины в алгоритмическом языке* гипертекста НС1.

Цель урока: Ознакомление с общим понятием *величина*.

План урока:

1. Обсуждение домашнего задания.
2. Практическая работа с гипертекстом.

Домашнее задание.

1. Обсуждение домашнего задания.

Урок 8 начинается с обсуждения начала пункта 2.9. Основной упор делается на тезис:

каждая величина имеет имя, тип, значение

Понимание этого тезиса будет необходимо при практической работе с гипертекстами. Дело в том, что при составлении алгоритмов ученики (а значит, и учитель) будут часто встречаться с двумя типами ошибок:

1. Если в алгоритме не задан тип величины, то в каждой строке, где встречается величина, на полях программы КуМир выводит предупреждение "Нет описания", а имя величины выделяется в строке цветом.
2. Если величина описана и тип ее правильно задан, при выполнении программы тем не менее на полях некоторой строки может появиться сообщение об ошибке "Не определено!!", и имя тоже будет выделено цветом. Такое сообщение означает, что величина к моменту выполнения строки не имела никакого определенного значения, ее значение было неопределенным.

Если ученик не знает, как исправить ошибку, прежде всего нужно помочь ему разобраться, к какому из двух типов она принадлежит.

2. Практическая работа с гипертекстом.

Далее начинается работа со страницей гипертекста *Величины в алгоритмическом языке*. Первые три задания этой страницы нужны только для более наглядного введения понятия *величина*. Этот материал

не будет использован в дальнейшем, поэтому можно ограничиться выполнением одного из заданий V2 и V3 или даже опустить оба, ограничившись просмотром демонстрации V1.

А вот задание V4 «Сумма вклада» должно быть разобрано в деталях. В учебнике приведен алгоритм, который только вычисляет сумму вклада, но не строит график роста суммы вклада. В гипертексте этот алгоритм доделан так, чтобы он строил такой график с помощью Чертежника. Можно рекомендовать ученикам запустить этот алгоритм несколько раз при разных начальных вкладах и разных месячных процентах. В пояснении к гипертекстам приведена одна из возможных дополнительных задач по мотивам алгоритма «Сумма вклада».

Домашнее задание.

Прочитать пункт 2.10.

Урок 9

Тема урока: Табличные величины.

Программное обеспечение: Страница *Табличные величины и цикл «для»* гипертекста НС1.

Цель урока: Ознакомление с табличными величинами.

План урока:

1. Обсуждение домашнего задания.
2. Практическая работа с гипертекстом.

Домашнее задание.

Материал темы *Табличные величины и цикл «для»* будет постоянно использоваться в модулях 9 и 10, поэтому должен быть пройден обстоятельно. Работа с табличными величинами может быть начата уже в конце урока 8.

1. Обсуждение домашнего задания.

Табличные величины очень важны для понимания того, как ЭВМ обрабатывает большие массивы информации. Кроме того, табличные величины являются важным техническим средством при решении многих задач курса, непосредственно не связанных с программированием. Таким образом, требуется, чтобы школьники «набили руку» при написании простейших алгоритмов работы с таблицами. Этому посвящена страница гипертекста *Табличные величины и цикл «для»*.

2. Практическая работа с гипертекстом.

На уроке должны быть выполнены все задания E1–E6.

Домашнее задание.

Прочитать пункт 2.11.

Урок 10

Тема урока: Литерные величины.

Программное обеспечение: Страница *Литерные величины* гипертекста HC1.

Цель урока: Ознакомление с литерными величинами, закрепление знаний, полученных при изучении главы 1.

План урока:

1. Обсуждение домашнего задания.
2. Практическая работа с гипертекстом.
3. Обзор основных понятий и конструкций главы 1.

Домашнее задание.

1. Обсуждение домашнего задания.

Урок начинается с обсуждения материала пункта 2.11. Этот материал будет многократно использован в модулях 9 и 10.

2. Практическая работа с гипертекстом.

После короткого обсуждения можно перейти к работе со страницей *Литерные величины* гипертекста. На этой странице 4 коротких контрольных вопроса, и работа с ней не должна занять много времени.

3. Обзор основных понятий и конструкций главы 1.

Этот урок завершает изучение главы, и его конец следует посвятить обзору основных понятий и конструкций главы 1.

Домашнее задание.

Прочитать пункты 3.1–3.5 из § 3.

Глава 2. КОДИРОВАНИЕ ИНФОРМАЦИИ БЕЗ КОМПЬЮТЕРОВ

Потоки информации, циркулирующие в современном обществе, все в большей степени переводятся в компьютерную форму, обрабатываются компьютерами, циркулируют по компьютерным сетям, преобразуются компьютерами и людьми из одной формы в другую. Любая такая информация по необходимости представлена в некоторой стандартизованной, закодированной форме. Как правило, существует несколько форм представления, кодировки одной и той же информации.

Задача этой главы — познакомить учеников с несколькими формами кодирования информации, возникшими в докомпьютерную эпоху. Это должно подготовить их к пониманию различных методов кодировки информации на компьютерах, а также подготовить психологически к особенностям задач на

кодирование информации, которые, как правило, не имеют единственного решения.

Урок 11

Тема урока: Запись чисел в древности.

Цель урока: Формирование представлений о записи чисел в древности и об алгоритмах работы с такой записью.

План урока:

1. Обсуждение домашнего задания.
2. Разбор египетских алгоритмов умножения и деления.

Домашнее задание.

1. Обсуждение домашнего задания.

Начало урока может быть посвящено обсуждению материала пунктов 3.1–3.5, прочитанных дома. Основной тезис этого урока

> *десятичная позиционная система — колоссальное достижение Человечества — возникла в результате разработки, «эксплуатации» и «конкуренции» многих других систем счисления.*

2. Разбор египетских алгоритмов умножения и деления.

На второй половине урока следует разобрать египетские алгоритмы умножения и деления на примерах, приведенных в учебнике в пункте 3.6.

Если учитель сочтет целесообразным, то при разборе египетских алгоритмов можно упомянуть, что в них неявно используется представление чисел в двоичной системе счисления. Например, представление 19 в виде суммы, составленной из чисел 1, 2, 4, 8, 16, — это просто-напросто запись этого числа в двоичной системе счисления (см. конец пункта 3.13).

Решение упражнения 1. Удвоим 30 три раза:

1	30
2	60
4	120
8	240

Дальше удваивать не нужно, так как из чисел в левом столбце уже можно составить 13, а именно $13 = 1 + 4 + 8$. Отметим соответствующие строчки вертикальными черточками:

1 |	30
2	60
4 |	120
8 |	240

Сумма отмеченных чисел в правом столбце и даст искомое произведение: $30 + 120 + 240 = 390$.

Решение упражнения 2. Будем удваивать 13:

1	13
2	26
4	52
8	104
16	208

и попробуем составить из чисел правого столбца 340 или, если точно составить не удастся, максимальное число, меньшее, чем 340. Удается составить число $338 = 208 + 104 + 26$, оно на 2 меньше 340. Значит, остаток от деления будет 2. Чтобы получить частное, отметим те строки, которые дали вклад в сумму 338:

1	13
2 |	26
4	52
8 |	104
16 |	208

и сложим отмеченные числа левого столбца: $16 + 8 + 2 = 26$. Значит, при делении 340 на 13 частное

будет равно 26, а остаток — 2.

Домашнее задание.

Прочитать пункты 3.7—3.9.

Урок 12

Тема урока: Славянская, римская и десятичная системы записи чисел.

Программное обеспечение: Страница *Запись чисел в древности* из гипертекста к главе 2 (НС2).

Цель урока: Формирование базы для рассмотрения позиционных систем счисления.

План урока:

1. Обсуждение домашнего задания.
2. Десятичная позиционная система счисления.

Домашнее задание.

1. Обсуждение домашнего задания.

Материал пунктов 3.7 и 3.8 учебника имеет в основном общекультурное значение и не будет использоваться в дальнейшем. Скорее всего, достаточно будет уделить 10—15 минут работе с соответствующими заданиями гипертекста к главе 2. Если учитель решит научить школьников читать даты, записанные римскими цифрами, то это может потребовать еще 15—20 минут дополнительной работы на уроке и соответствующего домашнего задания.

Упражнение 3 из пункта 3.8 не предназначено для выработки навыков сложения римских чисел. Его основная цель — демонстрация сложности работы с римскими числами. Самый простой способ его решения — перевод чисел в десятичную систему, а затем перевод результата обратно.

2. Десятичная позиционная система счисления.

Материал пункта 3.9 начинает тему позиционных систем счисления, которая будет продолжена на

следующем уроке (п.п. 3.10—3.11 и 3.13). Следует добиться от учеников полного понимания примера разложения четырехзначного десятичного числа по степеням числа 10.

Домашнее задание.

Прочитать пункты 3.10–3.13.

Урок 13

Тема урока: Позиционные системы счисления.

Программное обеспечение: Страница *Запись чисел в древности* гипертекста НС2.

Цель урока: Формирование представления о позиционных системах счисления.

План урока:

1. Позиционные системы счисления.
2. Двоичная система счисления.

Домашнее задание.

1. Позиционные системы счисления.

Разбираться в деталях в общей формуле для числа в p-ичной системе счисления (пункт 3.10) не обязательно. Но общая концепция позиционной системы счисления должна быть понята, и в этом могут помочь задания гипертекста по вавилонской системе счисления.

2. Двоичная система счисления.

Материал пункта 3.13 к концу 13 урока также должен быть освоен, хотя бы вчерне. Этот материал — стартовая площадка для важного материала пунктов 3.14, 3.15 и 3.16.

Домашнее задание.

Выбор домашних заданий остается на усмотрение учителя. Например, после урока 13 можно дать задания по материалу пункта 3.13 и такое задание:

Подсчитать общее количество
а) двухзначных десятичных чисел;
б) четырехзначных десятичных чисел;
в) не более, чем четырехзначных десятичных чисел;
г) десятизначных двоичных чисел.

Решение.
а) Минимальное такое число — 10, а максимальное — 99. Получается всего $99-10+1=90$ чисел.
б) Минимальное число 1000, а максимальное — 9999. Получается всего 9000 чисел.
в) Минимальное число 0, а максимальное — 9999. Всего 10 000 чисел.
г) Минимальное число — 2^9, а максимальное — $2^9+2^8+2^7+2^6+2^5+2^4+2^3+2^2+2^1+2^0=2^{10}-1$. Получается всего $(2^{10}-1)-2^9+1=2^9=512$ чисел.

Урок 14

Тема урока: Алгоритмы перевода в двоичную систему счисления.

Программное обеспечение: Страница *Двоичная система счисления* гипертекста НС2.

Цель урока: Закрепление представления о позиционных системах счисления (на примере работы с двоичной системой счисления).

План урока:
1. Пример перевода числа в двоичную систему счисления.
2. Просмотр общего алгоритма перевода в двоичную систему счисления.
3. Обсуждение алгоритма перевода.
4. Модификация алгоритма перевода.
Домашнее задание.

Нужно отдавать себе отчет в том, что материал по двоичной системе счисления проходится в два этапа.

На первом этапе ученики должны освоить алгоритмы перевода в двоичную и десятичную системы на интуитивном уровне подобно тому, как в младших классах они осваивали действия с многозначными числами.

На втором этапе ученики должны осмыслить и формализовать навыки, полученные на первом этапе, должны понять формальное описание освоенных ранее навыков на алгоритмическом языке. Второй этап становится возможным только после освоения первого.

Второй этап труден.

> Предание гласит, что однажды сороконожку попросили объяснить, как она ходит. Она начала было бойко объяснять, потом запуталась и, о ужас, тут же разучилась ходить вообще.

1. Пример перевода числа в двоичную систему счисления.

Урок 14 можно начать с разбора алгоритмов перевода чисел в двоичную систему счисления. Интуитивно проще рассмотренный на предыдущем уроке алгоритм из пункта 3.13, последовательно отщепляющий от числа максимально возможную степень двойки. Алгоритм из пункта 3.14 труднее понять, но легче записать на алгоритмическом языке.

Этот алгоритм можно объяснить по аналогии с десятичной системой. Пусть один из учеников задумает пятизначное десятичное число, а учитель попробует его отгадать. Пусть задуманное число, неизвестное учителю, было 23 095. На первом шагу учитель просит ученика разделить число на 10 и сообщить остаток. Ученик сообщает остаток 5 и помнит частное 2309. На втором шагу учитель просит ученика поделить частное на 10 и приписывает сообщенный ему остаток 9 слева к предыдущему остатку 5. В этот

момент учитель уже получил 95, а ученик помнит трехзначное число 230. На очередном шагу учитель просит ученика разделить очередное частное на 10 и приписывает сообщенный ему остаток к ранее полученным цифрам числа. После третьего шага учитель получит 095, после четвертого — 3095, после пятого — задуманное число 23 095.

Остаток любого числа от деления на 2 равен 0 или 1 и не зависит от того, в какой системе счисления записано число. Правда, если число уже записано в двоичной системе, то нет никакого труда найти остаток и частное при делении его на 2. Остатком будет последняя цифра числа, а частным — число, полученное отбрасыванием последней двоичной цифры числа. Если это частное снова поделить с остатком на 2, то мы получим следующую двоичную цифру числа. Таким образом,

двоичные знаки числа, начиная с младшего, могут быть получены вычислением последовательных остатков от деления числа на 2.

А деление на 2 можно проводить не в двоичной, а в десятичной системе счисления. В этом и состоит алгоритм перевода числа в двоичную систему счисления, приведенный в пункте 3.14.

2. Просмотр общего алгоритма перевода в двоичную систему.

После завершения первого этапа можно перейти к работе в гипертексте. Алгоритм перевода следует «прокрутить» для небольших N, для которых результат перевода очевиден или легко вычисляем. Например, N = 4, N = 5, N = 6, N = 7, N = 16, N = 17.

3. Обсуждение алгоритма.

Далее следует подробно обсудить запись алгоритма, остановившись на следующих вопросах:

1. Аргументом алгоритма является положительное целое число. Оно обозначается через N. Если на вход алгоритма будет ошибочно подано отрицательное число или 0, то алгоритм откажется работать и немедленно выдаст на полях строки «дано» сообщение об ошибке.

2. Результатом алгоритма будет вовсе не число, а строка из символов "1" и "0" — *двоичное разложение* числа N. Эта строка обозначается через t. Длина строки t заранее неизвестна и зависит от исходного числа N. Чем больше было N, тем длиннее будет эта строка.

3. В начале работы алгоритма будет сделана копия числа N. Величина, в которую будет скопировано N, обозначается x. В процессе работы алгоритма величина x будет меняться, а величина N останется неизменной.

4. Результат t получается последовательно, цифра за цифрой, начиная с младших цифр. В начале работы алгоритма строка t пустая, в ней нет ни одной цифры.

5. В алгоритме используются две операции над целыми числами: mod(x,2) и div(x,2). Первая операция дает остаток от деления x на 2, а вторая — частное от деления x на 2. Если представить себе мысленно двоичное разложение числа x, то mod(x,2) есть не что иное, как последняя цифра этого разложения, а команда x:=div(x,2) есть не что иное, как отбрасывание последней цифры в этом разложении.

6. Приписывание к t слева нуля записывается как t:="0"+t, а приписывание единицы записывается как t:="1"+t.

7. При выполнении цикла от x последовательно отбрасывается последняя цифра, пока все они не окажутся отброшенными и x не станет равным нулю. Отбрасываемые цифры последовательно припи-

сываются слева к строке t, и по завершении цикла эта строка будет содержать все двоичные цифры исходного числа x, т.е. числа N.

4. Модификация алгоритма перевода.

Рекомендуется после разбора алгоритма в классе еще раз запустить гипертекст и снова понаблюдать за работой алгоритма. В этот момент можно заняться модификацией этого алгоритма для перевода в другие системы счисления. Например, в троичную. Это позволит лучше понять алгоритм.

Переделанный алгоритм может выглядеть так:

```
алг перевод в троичную (арг цел N, рез лит t)
 дано  N>0
 надо | t=строка троичного разложения числа N
нач цел x,остаток
| x:=N
| t:=""
| нц пока x>0
| | остаток:=mod(x,3)
| | если остаток=0 то t:="0"+t все
| | если остаток=1 то t:="1"+t все
| | если остаток=2 то t:="2"+t все
| | x:=div(x,3)
| кц
кон
```

Общая идея алгоритма остается прежней, только вместо деления на 2 проводится деление на 3. Поскольку при делении на три могут быть три различных остатка: 0, 1 или 2, одной конструкцией «если» обойтись не удается и используются три такие конструкции.

Замечание. На современном компьютере целые числа 0, 1, 2 и символы "0", "1" или "2" — это объекты совершенно разного типа. Преобразовать цифры в символы можно операцией лит: лит(0)="0", лит(1)="1", лит(2)="2". Так что вместо трех «если» в алгоритме можно коротко записать:

```
t:=лит(остаток)+t,
```

но такая запись будет для учеников менее наглядной.

Домашнее задание.

Прочесть и разобрать пункт 3.16, прочесть 3.17.

Урок 15

Тема урока: Системы счисления, родственные двоичной.

Программное обеспечение: Страницы *Системы счисления с другим основанием* и *Смешанные системы счисления* гипертекста НС2.

Цель урока: Ознакомление с другими позиционными системами счисления.

План урока:

1. Обсуждение алгоритма перевода из двоичной системы счисления в десятичную.
2. Системы счисления, родственные двоичной.
3*. Смешанные непозиционные системы счисления.

Домашнее задание.

1. Обсуждение алгоритма перевода из двоичной системы счисления в десятичную.

Урок начинается с обсуждения прочитанного дома материала. Алгоритм в пункте 3.16 достаточно прост. Перебираются все цифры двоичного разложения, начиная со старшей, и суммируются все степени двойки, соответствующие единицам разложения. Если число девятизначное, то его старшей цифре соответствует 2**8. В общем случае старшей цифре соответствует 2**j, где j=длин(t)-1.

При обсуждении алгоритма следует обратить внимание учеников на то, что в алгоритме перевод из двоичной отсутствует защита от ввода некорректных данных (любой символ, отличный от 1, будет восприниматься в нем как 0).

В качестве дополнительной задачи можно дать переделку этого алгоритма для перевода из троичной системы. Вот одно из возможных решений:

```
алг перевод из троичной (арг лит t, рез цел N)
 дано | t строка из нулей, единиц и двоек
 надо | найти такое N, что t = (троичное
      | разложение N)
нач цел x, i, j
| N:=0
| нц для i от 1 до длин(t)
| | j:=длин(t)-i
| | если t[i]="1" то N:=N+  3**j все
| | если t[i]="2" то N:=N+2*3**j все
| кц
кон
```

2. Системы счисления, родственные двоичной.

Далее можно коротко обсудить материал пункта 3.17 и перейти к работе с гипертекстами по пунктам 3.16 и 3.17.

3*. Смешанные непозиционные системы счисления.

В конце урока можно порекомендовать желающим заглянуть в гипертексты по пунктам 3.18* и 3.19*. Материал этих пунктов дополнительный и может быть использован во внеклассной работе или для индивидуальных заданий.

Основная сложность использования непозиционных систем счисления заключается в том, что числа в этой системе невозможно сравнивать. Глядя на непозиционную запись числа, нельзя сказать, какое

число больше — $001_{2-3-7} = 36$ или $125_{2-3-7} = 5$. Именно поэтому такие системы счисления не получили широкого распространения.

Решение упражнений:

5.

$$1 = 1111_{2-3-5-7},$$
$$2 = 0222_{2-3-5-7},$$
$$10 = 0103_{2-3-5-7},$$
$$2 \cdot 3 \cdot 7 = 42 = 0020_{2-3-5-7},$$
$$2 \cdot 5 \cdot 7 = 70 = 0100_{2-3-5-7}$$

(в двух последних примерах по три цифры непозиционной записи можно получить, не вычисляя произведение).

6. После прибавления 1 (по правилам, рассмотренным в пункте 3.18*) получится $0000_{2-3-5-7}$. Эта запись соответствует числам 0, $2 \cdot 3 \cdot 5 \cdot 7 = 210$, $2 \cdot 210$, $3 \cdot 210$ и т.д. Если рассматривать только неотрицательные числа от 0 до 209, то получается единственное решение задачи: $1246_{2-3-5-7} = 209_{10}$.

> Общий алгоритм перевода из смешанной непозиционной системы счисления в десятичную довольно сложен, поэтому в данном курсе он не разбирается.

Домашнее задание.

Прочитать пункт 3.20 из § 3 и § 4.

Урок 16

Тема урока: Геометрическое представление чисел. Механические вычислители.

Программное обеспечение: Страница *Геометрическое представление чисел* гипертекста НС2.

Цель урока: Ознакомление с преимуществами геометрического представления чисел. Знакомство с

историей развития докомпьютерных средств вычисления.

План урока:

1. Геометрическое представление чисел.
2. Механические вычислители.

Домашнее задание.

1. Геометрическое представление чисел.

Тема геометрического представления чисел завершает разговор о числах. Геометрическое представление (по сравнению с алгебраическим) обладает повышенной наглядностью и надежностью и может использоваться в тех случаях, когда большая точность представления не нужна. Особенность десятичного представления чисел состоит в том, что ошибка в восприятии всего лишь одной цифры может привести к громадной ошибке в восприятии всего числа. Кроме того, близкие по величине числа могут представляться в десятичной системе совершенно разными наборами цифр.

Этой теме посвящено упражнение 7. Оно подчеркивает тот факт, что далекие друг от друга числа 900 и 600 графически гораздо более похожи, чем отличающиеся на 1 числа 600 и 599. Последние два числа вообще не имеют ни одной общей цифры.

Решение упражнения 8. Вот как эта проблема была решена в авиации. Метры по-прежнему кодируются с помощью стрелки на круговом циферблате, а вот километры — с помощью линейной шкалы:

Глядя на альтиметр, летчик на самом деле получает информацию не только о высоте полета, но и о скорости ее изменения. Быстрое вращение стрелки

метров указывает на большую скорость, медленное вращение или остановка стрелки говорит о том, что полет идет почти горизонтально, без потери и набора высоты.

> Конкуренцию между геометрическим и цифровым методами представления чисел интересно проследить на примере обычных часов. Появившись на рынке впервые, электронные часы с цифровым выводом времени мгновенно стали популярными, символизируя в глазах их обладателей новые технологии XX века. В пору расцвета этой моды в ответ на вопрос «Который час?» можно было услышать что-то вроде «Двадцать один двадцать три». Но постепенно выяснилось, что во многих случаях, когда мы смотрим на часы, такая детальная информация нам не нужна. А более грубую информацию удобнее получить, взглянув на положение стрелок на циферблате часов. Так возникла новая мода — комбинированные часы: вы одновременно видите и циферблат со стрелками, и цифры, показывающие время. Интересно, что никаких механических стрелок в таких часах уже нет. Стрелки изображаются электронно на миниатюрном дисплее — циферблате часов.

Многие современные компьютерные программы выводят числовую информацию в геометрическом виде: позиция курсора в файле, время, оставшееся до завершения какой-либо операции (например, копирования файла) и т.д.

Геометрическое представление чисел используется и для задания числовых параметров программы: на экране имитируются различные механические регуляторы, изображающие эти параметры, и их можно двигать мышкой. Так в современных компьютерах меняются: громкость звучания звукового сопровождения, скорость выдачи информации и т.д.

2. Механические вычислители.

К теме геометрического задания чисел примыкает тема механического задания чисел и механических устройств для проведения арифметических действий (пункт 4.1).

Остаток параграфа 4 посвящен истории вычислительной техники. Исходя из целей курса, нужно подчеркнуть в обсуждении этой темы два примера устройств для массовой автоматической переработки информации: станки Жаккарда и перфокарточные машины Холлерита.

Домашнее задание.

Прочитать вступление к § 5 и пункты 5.1—5.2.

Урок 17

Тема урока: Дискретные координаты.

Программное обеспечение: Страница *Координаты* гипертекста HC2.

Цель урока: Формирование представления о координатах как о способе *однозначного* задания объекта некоторого заранее описанного класса. Примеры координат на шахматной доске и в кинозале должны помочь ученику избавиться от «математического привкуса» термина *координаты*.

План урока:

1. Введение координат на шахматной и шашечной досках.
2. Задание координат в кинозале.

Домашнее задание.

1. Введение координат на шахматной и шашечной досках.

Урок 17 начинается коротким введением буквенно-цифровых координат на шахматной доске. Далее

во время урока начинает решаться упражнение 9 про международные стоклеточные шашки. Одно из возможных решений этого упражнения таково.

Решение упражнения 9. В международных шашках все черные клетки занумерованы подряд от 1 до 50, начиная с левой верхней (если смотреть со стороны играющего белыми).

Эту систему ученики еще раз увидят при работе с гипертекстом. Такая система удобна тем, что охватывает только «игровые» черные клетки, не затрагивая вовсе «лишние» белые. Неудобство ее в том, что по номеру не так легко определить положение клетки.

Конкурировать с такой системой могло бы обобщение шахматной системы: горизонтали нумеруются от 1 до 10, а вертикали обозначаются латинскими буквами a, b, ... h, i, j. Достоинства такой системы — легкое определение положения клетки на доске. Недостаток — кодировка «лишних» белых клеток, которые в игре фактически не участвуют.

При обозначении клеток шашечной доски общепринят один способ задания координат. Но во многих ситуациях одновременно применяются несколько способов.

При каждом удобном случае должна обсуждаться возможность выбора, конкуренции и сосуществования нескольких способов задания координат. Ученики без труда приведут несколько таких примеров.

Например, на температурной оси общеприняты несколько систем координат. Их называют шкалами: шкала Цельсия, шкала Кельвина, шкала Фаренгейта.

2. Задание координат в кинозале.

После обсуждения координат на шахматной доске можно перейти к заданию координат кресел в ки-

нозале. Как занумеровать кресла? Подряд от первого до последнего, как в международных шашках? Или ввести двойную нумерацию, как в шахматах?

Эти вопросы не такие уж игрушечные. Похожие проблемы возникают при задании координат на магнитном диске. Что выбрать: нумеровать все сектора подряд или воспользоваться двойной нумерацией цилиндр/сектор?

Задача 1. В кинозале 30 рядов по 28 мест в каждом. Все 840 кресел занумерованы начиная с первого ряда слева направо. Как определить номер ряда и места по порядковому номеру кресла и обратно?

Решение. Пусть x — порядковый номер кресла. Если $x \le 28$, то кресло расположено в первом ряду и номер места совпадает с x. Если $x = 28+1$, то это кресло во втором ряду на первом месте. Если $x = 28 + 2$, то это кресло во втором ряду на втором месте. В этих случаях номер места равен остатку от деления x на 28:

$$\text{номер места} = \text{mod}(x,\ 28)$$

Легко сообразить, что эта формула действует во всех возможных случаях. Например, 15-е кресло 6-го ряда имеет порядковый номер $5 \times 28 + 15$ и остаток от деления этого номера на 28 как раз и равен $\text{mod}(5 \times 28 + 15,\ 28)=15$.

Номер ряда в последнем случае равен частному от деления x на 28, увеличенному на 1:

$$\text{номер ряда} = \text{div}(x,\ 28)+1$$

Эта формула также действует во всех возможных случаях.

Найти порядковый номер кресла по номеру ряда и номеру места можно так:

$$x = (\text{номер ряда} - 1) \times 28 + \text{номер места}$$

Теперь не составит труда записать формулы в общем случае, когда в кинозале n рядов по m мест в каждом:

$$\text{номер места} = \text{mod}(x, m)$$

$$\text{номер ряда} = \text{div}(x, m) + 1$$

$$x = (\text{номер ряда} - 1) \times m + \text{номер места}$$

Заметьте, что общее число рядов в эти формулы не входит.

Задача 2. В отеле 30 этажей по 28 комнат на каждом этаже. Поскольку клиенты не любят число 13, то этажи занумерованы от 1 до 31 с пропуском 13-го этажа. Аналогично на каждом этаже комнаты занумерованы от 1 до 29 с пропуском номера 13. Кроме того, комнаты занумерованы подряд начиная с первого этажа в порядке возрастания их номеров на этаже. Как найти номер этажа и номер комнаты на этаже по порядковому номеру комнаты?

Решение. Пусть x — порядковый номер комнаты. Вычислим сначала i — истинный номер этажа (без пропуска 13) и j — истинный номер комнаты на этаже. Согласно предыдущему упражнению, это делается так:

```
j:=mod(x,28)
i:=div(x,28)+1
```

Теперь уже легко учесть суеверие клиентов:

```
если j>=13 то комната:=j+1 иначе комната:=j все
если i>=13 то этаж   :=i+1 иначе этаж   :=i все
```

Домашнее задание.

Учитель сам определит объем материала по координатам на шахматной доске и в кинозале, достаточного на один урок, равно как и домашнее задание по этой теме.

Урок 18

Тема урока: Непрерывные координаты.

Программное обеспечение: Страница *Координаты* гипертекста НС2.

Цель урока: Формирование представления о способах задания непрерывных координат в пространстве.

План урока:

1. Практическая работа в гипертексте.
2. Введение координат на плоскости.
3. Разбор упражнений на полярные координаты.

4*. Координаты на местности и карте.

Домашнее задание.

1. Практическая работа в гипертексте.

Урок 18 можно начать с 10-минутной работы в гипертексте по темам урока 17.

2. Введение координат на плоскости.

Далее можно перейти к координатам на плоскости. Необходимо подчеркнуть три тезиса:

1. Есть разные типы систем координат (декартова прямоугольная, косоугольная, полярная и другие).
2. Даже если тип и выбран, на плоскости можно ввести множество систем координат данного типа.
3. Нет самой удобной системы координат на все случаи жизни. В одних ситуациях удобно использовать одну систему координат, в других — другую. Выбор системы координат — первый этап решения многих задач.

Далее нужно разобрать примеры, когда информация о точке недостаточна для однозначного задания точки и, стало быть, эту информацию нельзя назвать координатами точки. Один такой пример дает упражнение 10.

Решение упражнения 10. Длина отрезка OM и площадь прямоугольника $OAMB$ не определяют точку M однозначно. Например, для двух разных точек M_1 и M_2, симметричных относительно биссектрисы первой четверти, будут совпадать и длины отрезков OM_1 и OM_2, и площади OAM_1B и OAM_2B.

3. Разбор упражнений на полярные координаты.

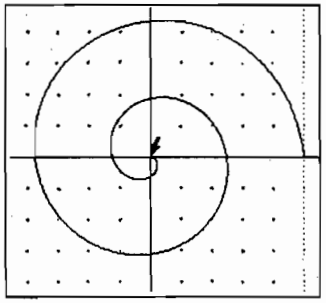
Наконец, если останется время, в конце урока 18 можно разобрать решение упражнений на полярные координаты.

Решение упражнения 11. Перевод из полярной системы координат r, ф в декартову производится по формулам:

```
x:=r*cos(ф/57.3)
y:=r*sin(ф/57.3)
```

Деление на 57.3 нужно для того, чтобы перейти из градусной меры в радианную. При этом углу в 180° будет соответствовать $180/57.3 = 3.141 \approx \pi$ радиан.

Решение упражнения 12. Кривая представляет собой спираль, делающую два оборота вокруг начала координат и заканчивающуюся в начале координат (см. рисунок). Эту спираль можно нарисовать с помощью следующего алгоритма:



```
алг спираль
нач вещ r, x,y, цел ф
  задать поле(-10,10,-7,7)
  сместиться в точку(5,0)
  опустить перо
```

```
нц для ф от 0 до 720
  r:=5*(720-ф)/720
  x:=r*cos(ф/57.3)
  y:=r*sin(ф/57.3); сместиться в точку(x,y)
кц
кон
```

Работу этого алгоритма можно увидеть, запустив гипертекст. Если алгоритм будет выполняться слишком медленно, запустите его не командой F9, а командой Ctrl F9. При этом на полях не будут изображаться значения величин и скорость исполнения повысится.

4*. Координаты на местности и карте.

Материал пунктов 5.5* и 5.6* необязательный. Он может быть рассмотрен на этом уроке, использован для организации заключительной конференции на уроке 21 или вовсе опущен. Материал пункта 5.7* также необязательный и может быть просмотрен учащимися самостоятельно.

Решение упражнения 13. Кратчайшие линии на поверхности шара — это дуги ***больших кругов***, т.е. окружностей, на поверхности шара, центр которых совпадает с центром шара. Все меридианы являются такими большими кругами. Так что полет вдоль меридиана на 2000 км и обратно был бы кратчайшим маршрутом заправщика. А вот параллели, за исключением экватора, большими кругами земного шара не являются и полет вдоль параллели Гринвича не дает кратчайшего маршрута. Пересеките мысленно глобус плоскостью, проходящей через его центр, точку взлета и точку заправки. Эта плоскость пересечет поверхность по окружности. Взяв дугу этой окружности от начальной до конечной точки маршрута, мы получим оптимальный маршрут заправщика.

Таким образом, при полете по кратчайшему маршруту заправщик будет в первую половину полета «забирать на север», а во вторую — возвращаться к исходной параллели.

Домашнее задание.

Прочитать § 6.

Урок 19

Тема урока: Нотная азбука.

Программное обеспечение: Страница *Нотная азбука* гипертекста НС2.

Цель урока: Формирование представления о способах кодирования музыкальных произведений.

План урока:

1. Звуковая гамма. Ноты и октавы.
2. Практическая работа в гипертексте.
3. Обсуждение способа записи мелодий.

Домашнее задание.

1. Звуковая гамма. Ноты и октавы.

Нотная запись — это еще одна интернациональная, широко известная система кодирования информации. Материал этой темы нагляден и хорошо поддержан гипертекстовыми заданиями. Урок можно начать с короткого обсуждения пункта 6.2.

2. Практическая работа в гипертексте.

Далее следует перейти к работе в гипертексте. Возможность изменить алгоритм и немедленно «услышать» результат его работы привлекательна для учеников, и они с удовольствием выполнят задания гипертекста и придумают собственные алгоритмы.

3. Обсуждение способа записи мелодий.

Нужно обратить внимание учеников, что, договорившись о способе записи мелодии в файле, мы

ввели новый способ кодирования музыкальной информации, новый стандарт. Его можно как-нибудь назвать. Скажем, СКОМ — «Стандарт Кодирования Одноголосых Мелодий». Стандарт СКОМ поддерживается программным обеспечением — имеется программа «проиграть», проигрывающая файлы, записанные в этом стандарте. Есть неформальное соглашение об именах файлов, содержащих мелодии, записанные в этом стандарте. Имена таких файлов имеют вид *.MUZ.

Хотя весь этот пример носит шуточный характер, многие реальные стандарты кодирования информации родились из практики схожим образом.

Домашнее задание.

Прочитать § 7.

Урок 20

Тема урока: Кодирование бухгалтерской информации.

Цель урока: Знакомство со способами кодирования больших объемов информации в бухгалтерии.

План урока:

1. Кодирование бухгалтерской информации.
2. Решение упражнений.
3*. Обработка бухгалтерской информации.
Домашнее задание.

1. Кодирование бухгалтерской информации.

Еще одна область, в которой стандарты кодирования информации были выработаны задолго до появления компьютеров, — это бухгалтерская и банковская деятельность. Начало урока 20 следует построить в виде рассказа учителя по материалу учебника или по дополнительным материалам либо в виде обсуждения этого материала.

2. Решение упражнений:

14.

40	"Готовая продукция"	актив	1500000
80	"Прибыль"	пассив	800000

15.

23457	2	10.12.1995	300000
23458	1	11.12.1995	500000

16. Таблица документов может выглядеть так:

номер документа	номер проводки	содержание

при этом для разных документов можно использовать разные таблицы. Поле номер проводки указывает, какая проводка была сделана на основании данного документа.

3*. Обработка бухгалтерской информации.

Материал пункта 7.3* необязательный и может быть пропущен. К нему можно будет вернуться позднее, при изучении кодирования информации на компьютере или при построении информационных моделей.

Решение упражнения 17. Содержательная часть алгоритма может выглядеть следующим образом:

```
сумма:=0
нц для i от 1 до n
| если дебет[i] = a и кредит[i] = b
| |то
```

```
    нц для j от 1 до m
      если 1 <= дата[j] <= 31 и тип1[j] = тип[i]
      то сумма:=сумма+колич[j]
      все
    кц
  все
кц
```

Домашнее задание.

Подготовиться к конференции.

Урок 21

Тема урока: Конференция.

Цель урока: Закрепление материала главы 2.

Предполагается, что к этому уроку школьники готовят короткие (5—10 минут) доклады о различных способах кодирования информации. Темы этих докладов распределяются заранее, по мере прохождения главы 2. Возможные темы докладов:

- Запись чисел у древних майя.
- Смешанные непозиционные системы счисления.
- Выполнение операций в смешанных непозиционных системах.
- Алгоритмы счета на счетах.
- Счетное устройство Непера.
- Координаты на местности и карте.
- Координаты вблизи поверхности Земли.
- Абстрактные координаты.
- Нотная азбука.

Материалы для докладов могут быть взяты из учебника и любых других источников.

Заключительное замечание к главе 2

Мы познакомились с различными методами кодирования информации, выработанными в докомпьютерную эпоху. Знание этих методов, их истории

помогает разрабатывать новые методы кодирования информации для современных компьютеров, ставших основным инструментом работы с информацией в современном мире.

Глава 3. КОДИРОВАНИЕ ИНФОРМАЦИИ НА КОМПЬЮТЕРЕ

Значение этой главы трудно переоценить. Ее изучение должно снабдить ученика достаточным набором взятых из практики примеров кодирования и перекодирования информации, научить простейшим приемам прикидочных подсчетов, связанных с информацией.

По существу, речь идет о базовых навыках, необходимых для повседневной жизни в XXI веке. Пояснить это можно таким примером.

Школьник середины XX века должен был уметь применять полученные в школе знания к решению такой задачи:

> Сколько рулонов обоев шириной 0,9 метра и длиной 12 метров нужно купить, чтобы отремонтировать квадратную комнату площадью 14 кв. метров и высотой 2,8 метра.

А школьнику конца XX — начала XXI века помимо этого надо будет уметь решать и задачки такого типа:

> На дискету объема 2.88 Мб требуется записать речь политического деятеля. Сколько минут высококачественной записи речи уместится на дискету?

Урок 22

Тема урока: Что такое информация.

Цель урока: Ознакомление с понятиями *информация* и *количество информации*.

План урока:

1. Нужно ли строго определять понятие «информация».
2. Информационный объем сообщения.
3*. Знакомство с другими определениями количества информации.

Домашнее задание.

Основной материал этого урока — существование двух разных мер для определения количества информации. Внешняя, техническая характеристика количества информации в сообщении легко определяется и измеряется и потому является очень важной на практике. Внутренняя (семантическая) мера количества информации в сообщении должна отвечать интуитивным представлениям о том, что некоторые весьма длинные сообщения в каком-то смысле несут мало информации или не несут вообще никакой информации.

Оказывается, что единого определения внутреннего количества информации в сообщении, отвечающего этому интуитивному представлению, не существует. А те частные определения, которые удается сформулировать в разных ситуациях, на практике не очень полезны. Этот вывод будет подкреплен на уроке 23 рассмотрением примера подсчета внутреннего количества информации по Колмогорову.

1. Нужно ли строго определять понятие «информация».

Урок можно начать с обсуждения материала пункта 8.1. Основная его мысль в том, что, хотя различные определения понятия «информация» и полезны, гнаться за каким-то сверхправильным определением не стоит. Достаточно примерно понимать, о чем идет речь.

2. Информационный объем сообщения.

Далее следует по возможности коротко обсудить нестрогое понятие «информативность сообщения» и остановиться подробнее на понятии «информационный объем сообщения». Заканчивается урок определением сложности сообщения по Колмогорову (детальному разбору этого определения будет посвящен следующий урок).

При желании в урок может быть включен материал пункта 10.1.

3*. Знакомство с другими определениями количества информации.

В популярной литературе (да и в научной тоже) можно встретить другие определения количества информации. Конец урока при желании можно посвятить обзору наиболее интересных из них (построенному, например, в форме рассказа учителя по приведенным ниже материалам).

Игра в вопросы и ответы

Рассмотрим пример. Кто-то загадал число в заранее определенном диапазоне (скажем, от 0 до 1000). Требуется отгадать это число, используя лишь вопросы, на которые надо отвечать «да»–«нет». По определению, информация равна минимальному количеству вопросов, которое нужно задать для получения ответа. Этот способ был предложен академиком Колмогоровым. По существу, это определение характеризует не одно сообщение, а весь класс допустимых в игре сообщений. По форме это определение включает двух играющих, различные сценарии их

поведения и потому в целом оказывается довольно запутанным.

Логарифм числа возможных сообщений

Этот подход к определению количества информации принадлежит американскому инженеру-связисту Р.Хартли.

Всякое поступающее сообщение выбирается из некоторого конечного набора. Чем богаче такой набор, тем труднее угадать, какое именно сообщение будет получено. Следовательно, тем больше информации оно несет с собой. Проще всего было бы приравнять количество информации количеству исходных наборов, т.е. считать, что количество информации $f(n)$ в сообщении, выбранном из n различных сообщений, равно числу n. Но тут возникает одно расхождение с интуицией.

Пусть существуют десять различных типов сообщений. Рассмотрим сложное сообщение, составленное из двух простых. При этом различных сложных сообщений будет уже $10 \times 10 = 100$, то есть информативность одного простого равна 10, а двух — уже 100. Естественным было бы, чтобы количество информации при комбинировании сообщений складывалось, а не перемножалось.

Так что откажемся от идеи, что информативность сообщения $f(n)$ равна n, и попробуем подобрать другую формулу. К этой формуле выдвигаются такие требования.

Пусть $f(n)$ — количество информации, полученной при принятии одного сообщения из n различных. Тогда $f(1) = 0$ (если сообщение всего одно, то, очевидно, его информативность нулевая). Далее, как было сказано выше, мы хотим, чтобы $f(ab) = f(a) + f(b)$. Наконец, естественно потребовать, чтобы при возрастании аргумента n значение

функции $f(n)$ также возрастало. При дополнительном условии неотрицательности $f(n)$ единственной (с точностью до множителя) функцией, удовлетворяющей этим условиям, будет двоичный логарифм n.

При $n = 2$ получаем $f(2) = 1$, а количество информации в сложном сообщении, состоящем из m элементарных, оказывается равно m. Таким образом, это определение для последовательностей из 0 и 1 совпадает с определением информационного объема сообщения.

Вероятностный подход

Этот подход был развит американским инженером и математиком Клодом Шенноном.

Пусть a_i, где $i = 1, .., N$ — различные сообщения, p_i — вероятности этих сообщений. Тогда величину

$$H(a) = \sum_{i=1}^{n} p_i \log_2 \frac{1}{p_i}$$

будем называть ***энтропией*** сообщения a или его мерой неопределенности. $H(a)$ — количество информации, которое сообщение a несет о самом себе. Единица измерения — бит.

Рассмотрим русский текст. Одну русскую букву можно рассматривать как сообщение, взятое из алфавита. Для простоты условимся не различать ъ и ь, а также е и ё, и дополним алфавит пробелом. Таким образом, если считать, что все буквы равновероятны, то информативность одной — $\log_2 32 = 5$ бит. На самом деле относительная частота появления различных букв неодинакова. Наибольшая она у пробела — 0,175, затем идет о — 0,09 и так далее. С учетом этих частот информативность (энтропия по Шеннону) уменьшается до 4,35 бит.

Попросту говоря, поскольку букв в алфавите у нас 32, то каждую из них можно записать 5-значным двоичным числом и это гарантирует кодирование с числом бит на букву, равным 5. Но если учесть, что одни буквы появляются чаще, а другие реже, а также учесть вероятности появления комбинаций букв, то можно придумать такой способ кодирования русских текстов, при котором на 100 букв потребуется не 500 бит, а всего 435.

Еще одно использование вероятностей букв в русском языке затронуто в пункте 11.8 учебника.

Домашнее задание.

Прочитать и разобрать пункт 8.3.

Урок 23

Тема урока: Количество информации по Колмогорову.

Цель урока: Закрепление представления о понятии *количество информации*.

План урока:

1. Обсуждение домашнего задания.
2. Разбор примера программы расшифровки.
3. Решение упражнений.

Домашнее задание.

1. Обсуждение домашнего задания.

Определение было сформулировано на уроке 22, и в начале урока 23 его стоит повторить. Обратите внимание, что в этом определении присутствует договоренность о том, что программа расшифровки будет записана на определенном языке программирования и передана как последовательность символов, составляющих программу. Отправитель телеграммы не может односторонне изменить язык программирования или способ записи программы. Получатель

в этом случае не сможет расшифровать сообщение. Все это означает, что определение Колмогорова зависит от выбранного языка. При выборе школьного алгоритмического получается одно значение количества информации, а при выборе Бейсика — другое. Идея Колмогорова была в том, что отличия эти будут незначительные.

Знакомство с определением можно начать с такого примера. Требуется передать последовательность, в которой парами идут одинаковые числа. Скажем, 11 11 237 237 2 2 23 23 23 23 и т.д. Как выбрать зашифрованное сообщение и программу расшифровки? Ответ очевиден. Зашифрованное сообщение будет состоять из 2, 4, 6-го и т.д. элементов последовательности, а программа расшифровки будет просто удваивать каждый элемент.

Можно еще сказать, что в определении Колмогорова содержится простая практическая рекомендация: вместо результатов работы какой-то программы над исходными данными часто выгоднее послать эту программу и исходные данные. Все остальное будет получено при запуске программы.

2. Разбор примера программы расшифровки.

В заключение урока следует подробно разобрать пример, приведенный в учебнике, и начать разбор упражнений.

3. Решение упражнений:

19. Вместо 2**18 можно написать 4**9.

20. Вместо *посимвольной* записи текста программы можно записать протокол нажатий на клавиши при наборе программы в КуМире. Каждую клавишу при этом можно обозначить одним символом, но для на-

глядности ниже это не сделано. Для примера приведем протокол набора программы расшифровки:

р [Enter] **ц е л** [пробел] **і** [Enter] [Esc] **д і** [пробел] **о т** [пробел] **4 * * 9** [пробел] **д о** [пробел] **7 6 3 5 0 5** [Enter] **в ы в о д** [пробел] **і**

Поскольку в КуМире конструкцию алг–нач–кон набирать не придется, а конструкция нц–для–кц вставляется по нажатии на две клавиши, такая «поклавишная» запись программы окажется короче, чем «посимвольная».

21. *Замечание.* В условии предполагается, что программа печатает ровно одно 1000-значное число.
Основная идея: 1000-значных чисел много, а 333-символьных программ слишком мало.

Действительно, существует меньше чем 256 различных символов, которые используются при записи программы. Заменим каждый из них десятичной записью кода этого символа. Получится не более чем 996-значное десятичное число. Таким образом, число программ, записываемых не более чем 332 символами, не превосходит 10^{997}. Некоторые из этих программ печатают одно 1000-значное число. Число таких программ, а следовательно, и печатаемых чисел не превосходит общего числа программ, т.е. 10^{997}. А поскольку количество 1000-значных чисел равно 9×10^{999}, подавляющее число из них не может быть напечатано никакой из этих программ.

Домашнее задание.

Прочитать пункты 8.4—8.6.

Урок 24

Тема урока: Кодирование чисел и текста.

Программное обеспечение: Страница *Двоичное кодирование* из гипертекста к главе 3 (НСЗ).

Цель урока: Формирование представления о способах двоичного кодирования чисел и текста на компьютере.

План урока:

1. Обсуждение домашнего задания.
2. Двоичное кодирование чисел.
3. Кодирование текста.
4. Азбука Морзе.
5. Домашнее задание.

1. Обсуждение домашнего задания.

Урок начинается с разбора пунктов, заданных на дом. Основная мысль пункта 8.4: грядет большая научно-техническая революция. Основная мысль пункта 8.5: трудно предсказать результат новых достижений компьютерных технологий, но несомненно, что их воздействие на общество будет весьма сильным.

Основная тема урока — двоичное кодирование. Трудное место тут — понять правила игры. А они таковы: нельзя использовать ничего, кроме нулей и единиц. Нельзя использовать знак "-" при кодировании чисел. Нельзя использовать пробел при кодировании текстов.

Материал пункта 9.1 «Почему все-таки двоичное кодирование» не требует разбора на уроке.

2. Двоичное кодирование чисел.

Первый пример двоичного кодирования — кодирование чисел. Поскольку ранее уже шла речь о двоичной системе счисления, материал во многом знаком ученикам. Надо только обратить внимание на то, что ячейки имеют фиксированную длину, поэтому приходится дополнять нуль и положительные числа нужным количеством нулей.

Кодирование отрицательных чисел может быть разобрано коротко, на одном примере.

Решение упражнений:

22. *0000 0000, 0000 0001, 0001 0100, 0100 0000, 1111 1111, 1111 1110, 1000 0001.*

23. 100 = *0110 0100*, −100 = *1001 1100*, числа 200 и −200 в знаковый байт не помещаются. В беззнаковую ячейку нельзя записать ни одно отрицательное число, но можно записать 200 = *1100 1000*.

3. Кодирование текста.

Кодирование текстов должно быть разобрано в объеме, достаточном для решения упражнений 24 и 25.

Решение упражнения 24в. Пользуясь таблицей, получим ответ в 16-ричном виде:

66 28 78 2B 31 29

Его можно перевести в двоичную форму, заменив каждую 16-ричную цифру на 4 двоичных:

0110 0110 0010 1000 0111 1000
0010 1011 0011 0001 0010 1001

Отступление

При изучении двоичного кодирования информации важно иметь в виду то, что большинство современных компьютеров (и практически все персональные) построены по *архитектуре фон Неймана*. Это означает, что память компьютера представляет собой просто последовательность байтов. Никакой информации о том, какой смысл имеют эти байты, не хранится. Байт со значением *10100100* может означать 164 (число без знака) или −92 (число со знаком), символ "д" (при текстовой интерпретации) или код команды MOVSB (процессора IBM PC).

С одной стороны, изобретение такого подхода в 1945 г. позволило сильно упростить устройство компьютеров и добиться большой гибкости. В самом деле процедура копирования информации будет работать одинаково, независимо от того, что копируется: текст, числа или коды программы.

Однако, с другой стороны, подход фон Неймана может приводить (и часто приводит) к ошибкам, когда коды команд неверно интерпретируются как последовательность чисел или наоборот.

В настоящее время уже разработаны компьютеры с другой организацией памяти (в них для указания того, что записано в памяти, используются ***теги***), однако пока такие машины не получили широкого распространения.

4. Азбука Морзе.

Рассмотрение азбуки Морзе служит в основном для того, чтобы подчеркнуть особенность двоичного кодирования — отсутствие разделителей между кодами (в двоичном кодировании используются **только** символы *0* и *1* и ничего больше).

Решение упражнений:

26. Можно попытаться кодировать точку нулем, тире — единицей и рассматривать полученную последовательность как двоичную запись числа. Однако при этом буквы Е, И и С будут кодироваться одним и тем же числом (т.к. *0*, *00* и *000* все означают число 0). Точно так же нельзя кодировать нулем тире, а единицей точку (нулевой код получится для Т, М и О). Правильным решением будет, например, кодирование точки цифрой 1, тире — цифрой 2 и использование троичной системы счисления.

28. Перенумеруем все буквы (всего их 31) и пробел, а потом воспользуемся двоичным кодированием (обо-

значив, например, *0* точкой, а *1* — тире). Для того чтобы код можно было однозначно декодировать, дополним более короткие последовательности слева нулями (т.е. точками) до 5 символов. Последовательность из 5 двоичных цифр позволяет закодировать как раз $2^5 = 32$ различных элемента.

Это решение не единственное, годятся и другие решения. Главное требование — ни один код не должен быть началом другого. Это условие заведомо выполняется, если все коды имеют одинаковую длину.

Домашнее задание.

Прочитать пункты 9.5 и 9.6.

Урок 25

Тема урока: Кодирование изображений и звука.

Программное обеспечение: Страницы *Кодирование изображений* и *Кодирование звука* гипертекста НСЗ.

Цель урока: Формирование представления о способах кодирования изображений и звука на компьютере.

План урока:

1. Обсуждение способов кодирования изображений.
2. Просмотр демонстраций в гипертексте.
3. Решение упражнений.
4. Обсуждение способа кодирования звука.

Домашнее задание.

1. Обсуждение способов кодирования изображений.

Материал пункта 9.5 не содержит ничего сложного. Рекомендуется поставить ученикам задачу кодирования изображений и решить ее вместе с ними в ходе обсуждения.

При разборе кодирования цветных изображений полезно вспомнить цветные телевизоры и их органы управления.

2. Просмотр демонстраций в гипертексте.

Идея разложения цветов на элементарные воспринимается значительно лучше после просмотра приведенных в гипертексте примеров. Желательно также продемонстрировать формирование цветов на бумаге на примере какого-нибудь цветного оттиска (по возможности плохого качества, с нарушением наложения цветов).

3. Решение упражнений:

29. Будем, например, кодировать тройку чисел (x_1, x_2, x_3) одним числом

$$X = 8^2 x_1 + 8x_2 + x_3.$$

Алгоритм декодирования фактически сводится к записи числа X в восьмеричной системе счисления. Заметим, что реально никаких арифметических операций выполнять не нужно — точно такой же результат получится, если записать подряд двоичное представление трех чисел (дополненное нулями до трех цифр) и рассматривать полученную запись как двоичное представление числа X.

Вместо 8 можно взять любое большее число (например, 10). Кодирование при этом будет менее эффективным, но, возможно, более понятным.

30. Закодируем каждый прямоугольник пятью числами: две координаты одного из углов, размеры и цвет. Всю картинку закодируем как последовательность таких пятерок чисел.

4. Обсуждение способа кодирования звука.

Разбираться детально в физической картине процесса дискретизации звука нет необходимости. До-

статочно, чтобы ученики усвоили конечный результат: звук записывается несколько тысяч раз в секунду. Чтобы получить максимально возможное качество, нужно записывать 44 тыс. раз в секунду. Для записи речи достаточно и 4 тыс. раз в секунду. Каждая запись занимает от 4 до 16 бит, в зависимости от требований к качеству.

Домашнее задание.

Прочитать пункты 9.7–9.9.

Урок 26

Тема урока: Кодирование звука, музыки и фильмов.

Программное обеспечение: Страница *Кодирование звука* гипертекста НСЗ.

Цели урока: Ознакомление с алгоритмами цифровой обработки звука. Формирование представления о способах кодирования музыки и фильмов на компьютере.

План урока:

1. Обсуждение возможностей цифровой обработки звука.
2. Решение упражнений.
3. Кодирование музыки.
4. Кодирование фильмов.

Домашнее задание.

1. Обсуждение возможностей цифровой обработки звука.

Как только способ кодирования звука усвоен, можно переходить к пункту 9.7 «Цифровая обработка звука». Это очень важный пункт. Он показывает на наглядном примере, какую гибкость при обработке дает перевод данных в цифровую форму. Алгоритм «нарастание звука» важен не сам по себе, а как первый пример цифровой обработки данных, с

которым столкнулись ученики. То же верно и для упражнений. Затронутые ими алгоритмы важны как примеры цифровой обработки.

2. Решение упражнений:

31. Решением может служить такая программа:

```
алг затухание звука к концу записи (арг цел N)
 дано | в файле in.xxx записано N значений звука
 надо | в файле out.xxx записан тот же звук, плавно
      | затухающий от начала к концу записи
нач цел i,x
| начать чтение("in.xxx")
| начать запись("out.xxx")
| i:=1
| нц пока не конец файла
| | ввод x
| | x:=x*(N-i)/N
| | вывод x
| | i:=i+1
| кц
| кончить запись
| кончить чтение
кон
```

32. Таблицу зв3[1:N] нужно заполнить так:

```
нц для i от 1 до N
| зв3[i]:=(зв1[i]+зв2[i])/2
кц
```

Просто сложить значения из таблиц зв1 и зв2 нельзя, в таблице зв3[1:N] при этом может получиться недопустимо большое число. Например, если кодирование было 8-разрядным, то может оказаться, что зв1[1]=127 и зв2[1]=127. Но сумма этих чисел (254) не может быть результатом 8-битного кодирования звука.

3. Кодирование музыки.

Стандарту MIDI достаточно уделить несколько минут. По сути дела, он развивает идею, которая была освоена учениками на примере *Чижика-пыжика*: мелодии можно программировать и хранить в виде программы. Сложность MIDI-синтезатора по сравнению с одноголосым синтезатором, встроенным в компьютер, не принципиальна. Но следует добиться от учащихся четкого понимания разницы между двумя методами записи звука: 1) указанием интенсивности 44 тыс. раз в секунду и 2) указанием, на какие клавиши синтезатора в какие моменты нужно нажимать.

4. Кодирование фильмов.

Фактически сводится к кодированию отдельных кадров. Поскольку принципы устройства кино и телевидения в настоящее время общеизвестны, этот материал не может вызвать каких-либо затруднений у учеников.

Решение упражнения 33. Теоретически можно. Запись звука и изображения независимы, а значит, и способы кодирования могут не зависеть друг от друга. Однако на практике стандарт MIDI при кодировании фильмов никогда не используется, поскольку фильмы с чисто музыкальным сопровождением очень редки.

Домашнее задание.

Прочитать § 10.

Урок 27

Тема урока: Измерение информации.

Цель урока: Формирование представлений о характерных объемах информации в окружающем мире.

План урока:

1. Обсуждение домашнего задания.
2. Подсчеты объемов информации.

Домашнее задание.

1. Обсуждение домашнего задания.

Материал параграфа 10 не содержит ничего сложного, и на его обсуждение достаточно затратить несколько минут.

2. Подсчеты объемов информации.

Всю оставшуюся часть урока следует посвятить разнообразным подсчетам объемов информации, скоростей их передачи и т.д. Годятся любые примеры и ситуации.

Решение упражнений:

34. 44 032 равно 172×256. С учетом того что одно значение занимает 2 байта и записывается два канала, получается 172 Кб в секунду. Минута займет $172 \times 60 = 10\,320$ Кб $= 10,078$ Мб. Запись в течение 80 минут займет $172 \times 60 \times 80 = 825\,600$ Кб $= 806,25$ Мб ≈ 800 Мб.

35. *Ответ:* Для записи всех учебников по всем предметам и за все годы обучения вполне хватит одного компакт-диска. Решение можно провести по образцу решения упражнения 36.

36. На странице БСЭ в среднем 75 строчек по 95 символов: $75 \cdot 95 = 7125$. В томе 650—700 страниц; получается около 5 Мб на один том. В энциклопедии 1950 г. было 50 томов. $5 \cdot 50 = 250$ Мб.

Домашнее задание.

Прочитать введение к § 11 и пункты 11.1—11.5.

Урок 28

Тема урока: Упаковка информации.

Программное обеспечение: Страница *Упаковка информации* гипертекста НСЗ.

Цель урока: Ознакомление со способами упаковки информации на компьютере.

План урока:

1. Разбор примеров упаковки информации.
2. Знакомство с алгоритмом Хафмана.
3. Разбор RLE-кодирования.
4. Методы упаковки с потерей информации.

Домашнее задание.

Постановки задач здесь привлекательны и интересны. Задача учителя — не потерять интерес учеников, углубившись в детали алгоритмов. Достаточно, если ученики поймут общую идею каждого из них.

1. Разбор примеров упаковки информации.

Урок 28 начинается с разбора примеров упаковки с потерей информации и упаковки без таких потерь. Ученики могут предложить собственные примеры. Скажем, замена списка учеников с именами и отчествами на список только с инициалами есть пример упаковки с потерей информации.

2. Знакомство с алгоритмом Хафмана.

Далее можно перейти к методу Хафмана упаковки информации и разобрать приведенный в учебнике пример. На идее учета повторений основан, в частности, очень популярный в коммерческих приложениях алгоритм LZ (названный так по первым буквам фамилий его создателей — Abraham Lempel и Jacob Ziv). Однако этот алгоритм слишком сложен, чтобы его можно было разбирать в школе.

3. Разбор RLE-кодирования.

В заключение урока нужно детально разобрать RLE-алгоритм. Этот метод упаковки информации из-за своей простоты используется при кодировании изображений довольно широко (хотя он дает далеко не лучшее качество упаковки), например, в графическом формате PCX или при записи фильмов в формате FLI.

Решение упражнений:

37. *Ответ:* $127 + 64 + 2 = 193$.

38. Такую длинную последовательность нужно кодировать двумя парами байтов. Например, 130 байтов со значением X в упакованном виде могут выглядеть как

11111111 X *10000011* X

Используется то, что $127 + 3 = 130$. Заметим, что это решение не единственное. Годится любая пара чисел, меньших 128 и дающих в сумме 130.

39. Это приведет к тому, что управляющий байт будет ошибочно принят за байт данных (или наоборот). В любом случае весь остаток сообщения за пропущенным байтом будет распакован неверно (и скорее всего, не будет иметь ничего общего с исходным изображением).

4. Методы упаковки с потерей информации.

Пункты 11.4 и 11.5 учебника по техническим причинам не поддержаны гипертекстовыми демонстрациями (продемонстрировать отбрасывание информации о цвете можно только при наличии в изображении не менее 256 цветов, а курс «Информационная культура» в качестве минимального требования к аппаратуре включает поддержку адаптера

EGA, в котором количество цветов ограничено шестнадцатью). Поэтому на уроке придется ограничиться кратким обсуждением этих пунктов.

Алгоритм упаковки звука в классе не разбирается и может быть задан на дом.

Домашнее задание.

Прочитать пункты 11.7 и 11.8.

Урок 29

Тема урока: Шифрование информации.

Программное обеспечение: Страница *Шифрование информации* гипертекста НСЗ.

Цель урока: Ознакомление со способами шифрования информации на компьютере.

План урока:

1. Разбор примера шифрования простой подстановкой.
2. Практическая работа по подсчету частот.
3. Пример расшифровки текста.

Домашнее задание.

1. Разбор примера шифрования простой подстановкой.

Урок 29 целиком посвящается шифрованию. Начинается он с разбора примера шифрования простой подстановкой. Объедините в алфавите буквы ё и е, а также ь и ъ, включим в алфавит пробел (на последнее, 32-е место). Зададим алгоритм шифрования так: а заменяется на б, б заменяется на в и т.д. по кругу: я заменяется на пробел, а пробел заменяется на а. Попросите каждого ученика зашифровать его собственную фамилию.

2. Практическая работа по подсчету частот.

Далее можно перейти к таблице частот в пункте 11.8. Попросите учеников подсчитать долю пробелов

и букв о в каком-нибудь тексте и сравните результаты с таблицей.

3. Пример расшифровки текста.

Вначале следует кратко разобрать содержание пункта 11.9. После этого можно переходить к практической работе по расшифровке текста, приведенного в гипертексте. В конце урока полезно обратить внимание учеников на то, что расшифровка все-таки осталась творческим процессом, а также обсудить, чем еще компьютер мог бы помочь при расшифровке. (Следующим шагом был бы учет частот появления двух- и трехбуквенных сочетаний и целых слов.)

Домашнее задание.

Прочитать пункт 11.10 из § 11, а также введение и начало пункта 12.1 (до подпункта «Взаимодействие процессора и памяти») из § 12.

Урок 30

Тема урока: Шифрование с открытым ключом. Память компьютера (начало).

Программное обеспечение: Страницы *Шифрование информации* и *Устройство памяти компьютера* гипертекста НСЗ.

Цели урока: Ознакомление со способами шифрования информации на компьютере. Формирование представления о понятии *адрес*.

План урока:

1. Обсуждение шифрования с открытым ключом.
2. Принципы устройства памяти компьютера.

Домашнее задание.

1. Обсуждение шифрования с открытым ключом.

В шифровании с открытым ключом неожиданна сама идея публикации, казалось бы, секретной

информации. Общая схема работы с ключами при шифровании и расшифровке сообщений может быть разобрана без всякого анализа самих алгоритмов шифрования.

Эту общую схему следует разобрать на уроке. Изложенная ниже реализация алгоритмов шифрования системы RSA для учеников слишком сложна.

Для применения системы RSA надо:

1) выбрать простые числа p и q и вычислить их произведение $n = p \times q$;
2) выбрать число d, которое должно быть взаимно простым с результатом умножения $(p-1)(q-1)$;
3) выбрать число e, удовлетворяющее условию:

$$e \times d \equiv 1 \ mod\left((p-1)(q-1)\right)$$

Открытый ключ E устанавливается в виде пары (e, n), а закрытый D — в виде пары (d, n). Шифрование/дешифрование числа M производится по формуле

$$C = M^m \mathrm{mod}(n)$$

где m — одно из чисел d или e. Число десятичных знаков p и q должно быть порядка 100.

2. Принципы устройства памяти компьютера.

В конце урока следует начать обсуждение темы «Принципы устройства памяти компьютера», которое будет продолжено на следующих двух уроках. Важнейшее понятие, которое должно быть разобрано на уроке 30, — понятие ***адреса***.

Домашнее задание.

Прочитать пункты 12.1 (начиная с подпункта «Взаимодействие процессора и памяти») и 12.2.

Урок 31

Тема урока: Память компьютера (продолжение).

Программное обеспечение: Страница *Устройство памяти компьютера* гипертекста НСЗ.

Цель урока: Ознакомление со структурой организации внутренней и внешней памяти компьютера

План урока:

1. Взаимодействие процессора и памяти.
2. Внешние запоминающие устройства.

Домашнее задание.

1. Взаимодействие процессора и памяти.

Заключительный параграф, посвященный структуре организации памяти компьютера, носит обзорный характер. Не нужно заставлять учеников разбираться во всех тонкостях организации памяти, важно только, чтобы у них сложилось общее впечатление о способах взаимодействия процессора и памяти.

2. Внешние запоминающие устройства.

В дополнение к обсуждению материалов книги для ученика на уроке можно провести обзор последних новинок компьютерной технологии по материалам свежей компьютерной прессы. Этот обзор может быть подготовлен как учителем, так и самими учениками (и проведен в форме мини-конференции).

Основными параметрами внешнего устройства являются емкость, скорость передачи данных, время поиска данных и стоимость хранения данных. Последнюю величину обычно выражают в долларах (или рублях) на 1 мегабайт данных. Следует, однако, иметь в виду, что единой методики расчета стоимости хранения 1 Мб для устройств со сменными носителями (к которым относится все, кроме винчестеров) не существует: стоимость дисковода тоже

надо как-то учитывать наряду со стоимостью дискеты, но это можно делать по-разному.

Домашнее задание.

Прочитать пункты 12.3–12.5. Полезно дать к следующему уроку задание: подтвердить (или уточнить) приведенные в пункте 12.3 («Что такое 100 мегабайт?») цифры самостоятельными расчетами. Данные для расчетов следует взять из параграфов 9 и 10.

Урок 32

Тема урока: Память компьютера (окончание).

Программное обеспечение: Страница *Устройство памяти компьютера* гипертекста НСЗ.

Цель урока: Формирование представлений о принципах организации памяти и структуре хранения информации в памяти компьютера.

План урока:

1. Что такое 100 мегабайт?
2. Принципы организации кэш-памяти.
3. Структура хранения информации в памяти.

Домашнее задание.

1. Что такое 100 мегабайт?

Формирование интуитивного представления о том, что же такое 100 Мб, очень важно для общекультурного развития. Однако подсчетам объемов информации был посвящен урок 27, поэтому на уроке 32 материал может быть рассмотрен очень кратко.

2. Принципы организации кэш-памяти.

Самым сложным вопросом в организации кэш-памяти является выбор и реализация метода очистки кэш-памяти при полном заполнении. Фактически этот метод должен «угадать», какие данные потребуются процессору в следующий момент и не стирать их из кэш-памяти. Реализация метода должна быть достаточно простой и, самое главное, очень быстрой (иначе все преимущество будет потеряно).

3. Структура хранения информации в памяти.

Основные понятия заключительного пункта главы (*файл* и *файловая система*) уже неоднократно встречались в курсе «Информационная культура». Хорошей задачей на этом уроке будет самостоятельное формулирование учениками определения понятия *файл*. Следует только иметь в виду, что добиваться получения абсолютно точного и полного определения бессмысленно — его просто не существует (как и у любого другого столь общего и фундаментального понятия, например, *информация* или *алгоритм*).

Приложения

1. Сводка используемых в КуМире клавиш

[Ctrl] [?] – вызов подсказки

[Alt$_{пр}$] [*буква*] или [F10] – переход в меню

[Esc] [End] – выход из задания

Редактирование программы

[Ctrl] [Bs] – **откатка** — восстанавливает предыдущее состояние текста; повторные нажатия возвращают ко все более ранним состояниям текста

[Ctrl] [Enter] – **накатка** — вновь вносит откаченные изменения в текст

Переключение режимов клавиатуры:

[Caps] [Alt] – долговременная смена (фиксация) алфавита (с русского на латинский и наоборот)

[Caps] [Shift] – долговременная смена (фиксация) регистра (с малых букв на большие и наоборот)

[Shift] [*X*] – смена регистра (большие/малые) для буквы *X*

[Alt$_{лев}$] [*X*] – смена алфавита (русский/латинский) для буквы *X*

Перемещения курсора по тексту:

[←],[↑],[↓],[→] – передвинуть курсор на одну позицию в соответствующем направлении

[Enter] – передвинуть курсор в начало следующей строки

[Tab] – табуляция вправо

[Shift] [Tab] – табуляция влево

[PgUp] – передвинуть курсор на экран вверх

[PgDown] – передвинуть курсор на экран вниз

[Ctrl] [←] – передвинуть курсор в начало строки

[Ctrl] [→] – передвинуть курсор в конец строки

Просмотр текста:

[Ctrl] [↑] – просмотр текста вверх (до нажатия любой клавиши или достижения верхней границы)

[Ctrl] [↓] – просмотр текста вниз (до нажатия любой клавиши или достижения нижней границы)

[ScrLock] – заблокировать изменение экрана в командах просмотра текста вверх/вниз

Вставка/удаление символов:

[Ins] – вставить пробел в позицию курсора

[Del] – удалить символ в позиции курсора

[Bs] – удалить предыдущий символ (слева от курсора) и сместить курсор влево на одну позицию

Вставка/удаление строк:

Shift Ins – вставить пустую строку

Shift Del – удалить строку или конструкцию

Копирование символов и строк:

F1 – запомнить символ

Shift F1 – запомнить и удалить символ из текста

F2 – вспомнить символы

F3 – запомнить строку

Shift F3 – запомнить строку и удалить ее из текста

F4 – вспомнить строки

Вставка шаблонов конструкций:

Esc А – вставить шаблон алгоритма

Esc Е – вставить шаблон команды "если"

Esc И – вставить конструкцию "иначе"

Esc В – вставить шаблон команды "выбор"

Esc С – (Случай) – вставить конструкцию "при"

Esc Ц – вставить шаблон команды цикла (нц-кц)

Esc П – вставить шаблон команды "пока"

Esc Д – вставить шаблон команды "для"

Esc У – вставить/убрать условие в конце цикла

Работа с памятью имен:

+ – дописать имя (используется клавиша на дополнительной клавиатуре)

Перемещение строки по тексту:

Caps ↑ – продвинуть текущую строку вверх

Caps ↓ – продвинуть текущую строку вниз

Выполнение программы

F8 – начать выполнение по шагам/выполнить очередной шаг

F7 – начать выполнение вспомогательного алгоритма по шагам

Alt F8 – непрерывно выполнить вспомогательный алгоритм до конца

F9 – непрерывно выполнить всю программу

Break – перейти к пошаговому выполнению или прекратить выполнение

F11 – непрерывно выполнять программу без вывода на поля

F12 – выполнить непрерывно до строки, в которой находится курсор (не включая эту строку)

ScrLock – при непрерывном выполнении включить/выключить отображение результатов на полях

Ctrl F2 – сбросить изменения, внесенные в обстановки исполнителей при предыдущем запуске программы, и восстановить начальные обстановки

Ctrl F8 – поставить/убрать точку останова

Ctrl I – посмотреть значение переменной

2. Что такое "КуМир–Гипертекст"

КуМир–Гипертекст — это редактор-компилятор школьного алгоритмического языка, погруженный в гипертекстовую оболочку.

Школьный алгоритмический язык был придуман академиком А.П.Ершовым и изложен в первом советском учебнике информатики 1985 года. В последующих учебниках он дорабатывался, но устоявшегося названия так и не получил. Первая версия языка поддерживалась системой Е-практикум (в честь А.П.Ершова). Окончательная версия языка описана в учебнике А.Г.Кушниренко и др. и поддерживается системами семейства КуМир.

Что такое гипертекст

Само слово *гипертекст* состоит из двух частей — приставки *гипер-* и корня *текст*. И действительно, в основе гипертекста лежит текст, но этот текст устроен сложнее, чем напечатанный в книге, и работа с ним ведется на компьютере.

Первое отличие от обычного текста (которое есть во всех гипертекстовых системах) — удобство работы со ссылками, примечаниями и другими связями между отдельными частями текста. Конечно, примечания и ссылки используются и в обычном, напечатанном тексте, но обычно авторы стараются не злоупотреблять этим. В самом деле обилие ссылок сильно замедляет чтение печатного текста — процесс поиска нужного места довольно утомителен, да еще надо не забыть заложить место, куда вернуться после чтения ссылки.

Другое дело работа за компьютером: достаточно подвести курсор к выделенному месту и нажать всего одну кнопку — и гипертекстовая система мгновенно перенесет вас куда надо. Текущее место в

тексте автоматически запомнится, и нажатие другой кнопки вернет вас в то место, откуда вы пришли.

Что такое активный гипертекст

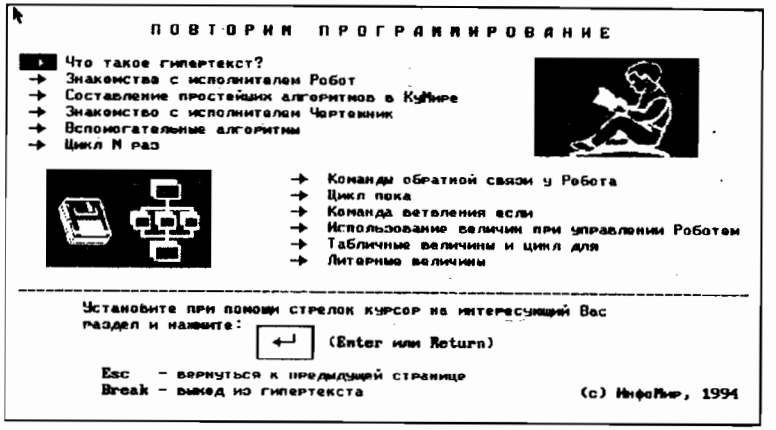
Одно из наиболее распространенных применений гипертекстов — это справочные системы и встроенные подсказки. Подсказка — это простейший, так называемый *пассивный*, гипертекст. В пассивном гипертексте представляемая информация не меняется в процессе просмотра, обратная связь отсутствует, единственная активность пользователя — чтение текста с экрана и выбор очередной страницы.

Активные гипертекстовые системы предоставляют пользователю более «живую», более привлекательную среду, дают ему возможность просматривать «мультфильмы», отвечать на контрольные вопросы, не только получать, но и создавать информацию; дают возможность быть активным, действовать самостоятельно, оставаясь тем не менее в одной и той же среде.

Разумеется, система с активным гипертекстом должна понимать какой-нибудь язык программирования или уметь подключать к гипертексту готовые программы (быть может, оформленные особым образом). КуМир умеет и то и другое. Программы, задающие активность в гипертексте, можно писать на школьном алгоритмическом языке (используя при этом все преимущества редактора-компилятора). Такие программы можно писать и на производственном языке (например, Си), затем компилировать их и подключать к гипертексту. А можно совмещать эти подходы: критические для эффективности модули писать на производственном языке, а все остальное — на школьном алгоритмическом.

3. Гипертекст к главе 1

Гипертекст начинается с ***головной страницы***, содержащей перечень из 12 ***тем***. Каждая из 12 тем представлена отдельной страницей, содержащей перечень заданий. Ниже приведены указанные 12 страниц с кратким описанием каждого задания.



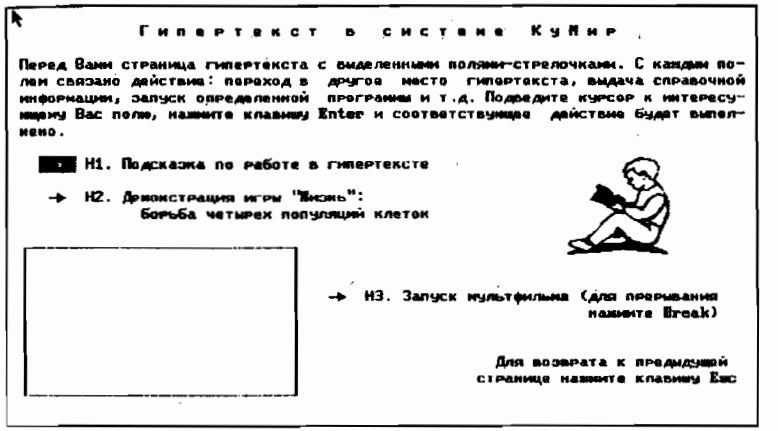
Тема «Что такое гипертекст?» (гипертекст в системе КуМир)

Н1. Подсказка по работе в гипертексте

Это задание демонстрирует выдачу небольшой порции справочной информации поверх исходной страницы. Возврат — клавиша Esc.

Н2. Демонстрация игры «Жизнь»

Это задание демонстрирует возможность запуска из гипертекста программы, работающей на полном экране. В качестве такой программы выбрана весьма привлекательная графически демонстрация игры



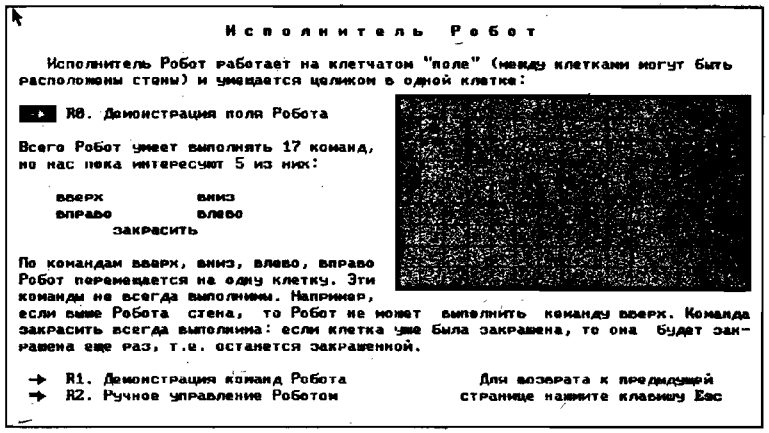
«Жизнь»*). Демонстрация показывает борьбу четырех популяций клеток за жизненное пространство. На экране сменяют друг друга фантастические цветные узоры, и наблюдать за этим — одно удовольствие. Начальные данные выбираются случайно при каждом новом запуске. Так что, если школьнику понравилось, он может запустить игру снова и увидеть новое развитие событий.

Графики, нарисованные в нижней части экрана разными цветами, показывают число «живых» клеток каждого цвета. Бледные цвета на игровом поле отмечают цвет клетки, которая последний раз побывала на этом поле. Основная часть программы, делающая один ход в игре «Жизнь», написана на языке Си по достаточно быстрому алгоритму. Выбор начальных условий и рисование графиков написаны на школьном алгоритмическом языке. Возврат — Esc.

Н3. Запуск мультфильма

Это задание демонстрирует возможность запуска из гипертекста программы, работающей в окне. При

*) Описание игры «Жизнь» можно найти, например, в книге: *Гарднер М.* Крестики – нолики. — М.: Мир, 1988.



запуске программы в окне рисуется сложная пространственная кривая. Возврат — Esc.

Тема «Исполнитель Робот»

R0. Демонстрация поля Робота

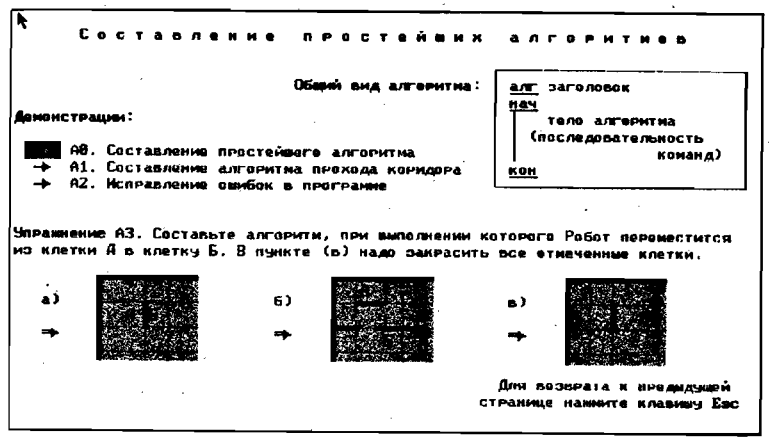
Это бесхитростная статическая демонстрация. Поверх поля Робота появляются пояснения: «это Робот», «это стена». Возврат — клавиша пробела.

R1. Демонстрация команд Робота

Примерно с секундным интервалом в окне появляется команда, которую ЭВМ дает Роботу, и через полсекунды Робот на поле выполняет эту команду. В конечном счете Робот приходит в точку, помеченную буквой Б, а на экране в окне остается протокол команд, поступавших от ЭВМ к Роботу. Этот протокол делает в дальнейшем естественным написание программ по управлению Роботом. Возврат — клавиша пробела.

R2. Ручное управление Роботом

На экране появляется пульт управления Роботом. Нажимая (мышью) кнопки на этом пульте,



можно дать Роботу любую из 5 команд. Никакая конкретная задача по управлению Роботом не сформулирована. Ее должен поставить учитель. Возможные задачи:

- перевести Робота в точку Б;
- закрасить все клетки, прилегающие к стене;
- с помощью закраски клеток изобразить что-нибудь на поле Робота, скажем, дату, свои инициалы и т.д.

Эта демонстрация должна показать «самостоятельность» Робота, возможность работы с ним в режиме «ручного управления», без всяких программ.

Возврат — нажатие кнопки **Выкл** на пульте управления Роботом.

Тема «Составление простейших алгоритмов»

В этой теме три демонстрации и три однотипных упражнения.

А0. Составление простейшего алгоритма

Речь идет о составлении следующего алгоритма:

```
алг Переход в точку Б
 дано |
 надо | Робот перешел в клетку Б
нач
|  вниз
|  вниз
|  вправо
|  вправо
|  вправо
|  вверх
кон
```

(Это тот самый алгоритм, который выполнялся в демонстрации R1 предыдущей темы.)

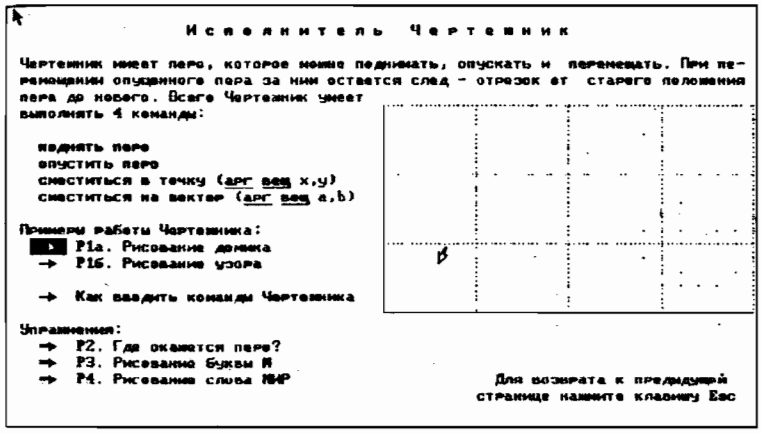
Демонстрация показывает процесс написания и пошагового исполнения алгоритма. На экране появляются указания, какие клавиши и зачем нужно нажимать. Ученик должен следовать этим указаниям буквально. При нажатии неправильной клавиши раздается предупреждающий звуковой сигнал. В процессе демонстрации осваиваются команда редактора «дописывание имени» и команда F8 пошагового исполнения алгоритма.

А1. Составление алгоритма прохода коридора

Демонстрация аналогична предыдущей. Осваиваются команды запоминания и вспоминания строк F3 и F4 и команда непрерывного исполнения алгоритма F9.

А2. Исправление ошибок в программе

Демонстрируется программа с ошибками; объясняется, как КуМир показывает ошибочные места. Демонстрируется исправление ошибок и запуск исправленной программы.



Упражнение А3

В любом из заданий А3 а)—в) ученик должен самостоятельно составить алгоритм. Законченный алгоритм следует выполнить по шагам или непрерывно и убедиться в его правильности. Однако в конце концов ученик должен «сдать алгоритм на проверку» компьютеру. Для этого нужно нажать Ctrl T. На правильный алгоритм компьютер реагирует сообщением «Замечательно!». После этого рядом с заголовком задания в гипертексте появляется «плюсик». Когда сделаны все три варианта упражнения, на экране видны три «плюсика».

Тема «Исполнитель Чертежник»

В этой теме три демонстрации, один проверочный вопрос и два упражнения на изменение алгоритмов.

P1a. Рисование домика

Примерно с секундным интервалом в окне появляется команда, которую ЭВМ дает Чертежнику, и через полсекунды Чертежник выполняет эту команду. В конечном счете на поле Чертежника оказывается нарисованным домик, а на экране в окне остаются последние команды, поступавшие от ЭВМ. Этот

протокол показывает наглядно вид команд с числовыми аргументами.

Р1б. Рисование узора

При выборе этого задания Чертежник рисует на поле криволинейный узор. Нужно обратить внимание учеников на то, что, хотя Чертежник умеет рисовать только прямолинейные отрезки, за счет рисования большого числа коротких отрезков можно рисовать и кривые.

Как вводить команды Чертежника

Команды Чертежника длинные, набирать их целиком при составлении алгоритма непрактично. Демонстрация показывает, как ускорить ввод команд Чертежника за счет использования команд редактора «дописать слово» и «запомнить строку»/«вспомнить строки».

Р2. Где окажется перо?

Это упражнение проверяет понимание команды Чертежника сместиться на вектор. При неверном решении предлагается новый вариант вопроса. Если нужно выйти из упражнения, так и не получив верного ответа, нажмите Break.

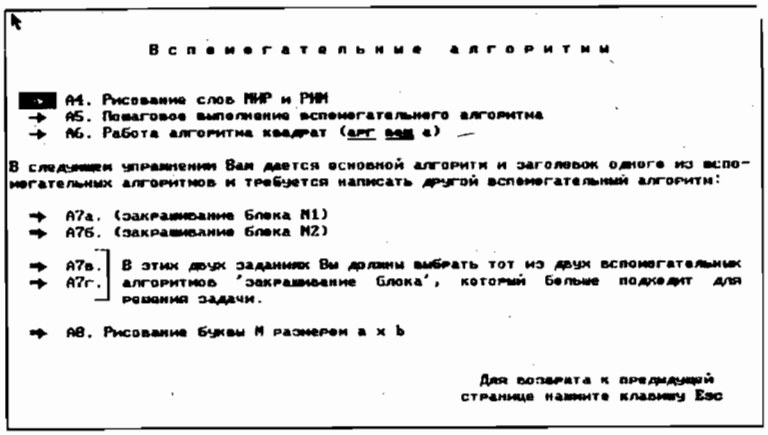
Р3. Рисование буквы М

На экране появляется заголовок алгоритма буква М, а на поле Чертежника — образец буквы, которую нужно нарисовать. Задача ученика — написать алгоритм. Одно из возможных решений приведено в пункте 2.3 книги для ученика, и, таким образом, это задание сводится к вводу в компьютер уже готового алгоритма. При выполнении задания рекомендуется пользоваться командами, показанными в демонстрации «Как вводить команды Чертежника». Законченный алгоритм нужно «сдать на проверку компьютеру», нажав Ctrl T. Возврат — Esc End.

Р4. Рисование слова МИР

В этом задании требуется не составить новый алгоритм, а лишь немного изменить заданный. Изменение в этом задании сводится к увеличению на 1 первых аргументов трех команд сместиться на вектор, как показано ниже:

```
алг МИР
 дано |перо поднято
 надо |вправо от точки с координатами (3,3)...
нач
 сместиться в точку(3,3)
 |
 | рисование буквы М:
 опустить перо
 сместиться на вектор( 0,  4 )
 сместиться на вектор( 1, -2 )
 сместиться на вектор( 1,  2 )
 сместиться на вектор( 0, -4 )
 поднять перо
 сместиться на вектор( 2,  0 )
 |
 | рисование буквы И:
 сместиться на вектор( 0,  4 )
 опустить перо
 сместиться на вектор( 0, -4 )
 сместиться на вектор( 2,  4 )
 сместиться на вектор( 0, -4 )
 поднять перо
 сместиться на вектор( 2,  0 )
 |
 | рисование буквы Р:
 опустить перо
 сместиться на вектор( 0,  4 )
 сместиться на вектор( 2,  0 )
```




```
| сместиться на вектор( 0, -2 )
| сместиться на вектор(-2,  0 )
| поднять перо
| сместиться на вектор( 4, -2 )
кон
```

(Измененные числа в программе подчеркнуты.)

Первые два изменения нужны, чтобы увеличить расстояние между буквами слова. Последнее изменение нужно, чтобы после рисования перо оказалось на 2 единицы правее конца последней буквы (это требуется в условии).

Тема «Вспомогательные алгоритмы»

А4. Рисование слов МИР и РИМ

Задана программа из основного алгоритма РИМ и трех вспомогательных алгоритмов. В основном алгоритме вспомогательные вызываются в неправильном порядке. Поэтому вместо слова «РИМ» Чертежник рисует слово «МИР». Требуется исправить программу. Обратите внимание учащихся, что решение задачи можно и нужно получить, не исправляя и даже

не заглядывая внутрь вспомогательных алгоритмов. Достаточно изменить основной алгоритм так:

```
алг РИМ
 дано | перо поднято
 надо | вправо от точки с координатами (3,3)...
нач
| сместиться в точку(3,3)
| буква М |                        | буква Р
| буква И | <— заменить на       | буква И
| буква Р |                        | буква М
кон
```

Это упражнение демонстрирует относительную независимость основного и вспомогательного алгоритмов.

А5. Пошаговое выполнение вспомогательного алгоритма

Эта демонстрация подкрепляет понятие вспомогательного алгоритма. При выполнении программы по шагам (F8) выполнение вспомогательного алгоритма считается одним шагом, вызов вспомогательного алгоритма выполняется как единое целое.

Если нужно детализировать процесс выполнения вспомогательного алгоритма, то используется команда F7.

А6. Работа алгоритма квадрат (арг вещ а)

Демонстрация показывает работу алгоритма при а=4 и при а=6.

```
алг квадрат (арг вещ а)
 дано | перо Чертежника в левом нижнем углу
      | будущего квадрата и поднято
 надо | карисован квадрат с длиной стороны а,
      | перо Чертежника в исходной точке и поднято
нач
| опустить перо
```

```
    сместиться на вектор( a, 0)
    сместиться на вектор( 0, a)
    сместиться на вектор(-a, 0)
    сместиться на вектор( 0,-a)
    поднять перо
кон
```

Упражнение А7

В этом упражнении дается основной алгоритм и заголовок одного из вспомогательных алгоритмов и требуется написать другой вспомогательный алгоритм. Основной алгоритм оперирует двумя вспомогательными алгоритмами, предназначенными для обхода стены и закрашивания блока. Ученик должен самостоятельно составить алгоритм обход стены. Особенность упражнения в том, что алгоритм закрашивания блока задан заранее, но ученику недоступен. Доступно лишь краткое описание этого алгоритма в строках дано–надо.

Пункты (а) и (б) этого упражнения парные, их рекомендуется выполнять подряд. В (а) дан алгоритм закрашивания блока N1, переводящий Робота в правый нижний угол блока. Алгоритм обход стены будет довольно длинным. Этот алгоритм должен перемещать Робота из правого нижнего угла блока размером 3×5 в левый верхний угол следующего блока. Вот одно из возможных решений задания А7а:

```
алг обход стены | задание А7а
 дано | Робот в конце блока (в правом нижнем углу)
 надо | Робот в начале следующего блока
нач
| вверх;вверх;вверх;вверх;вверх;вверх;
| вправо;вправо;вправо
| вниз;вниз
кон
```

В А7б дан алгоритм закрашивания блока N2, переводящий Робота в правый верхний угол блока. Алгоритм обход стены будет в этом случае короче, вместо 6 команд вверх понадобятся только две:

```
алг обход стены | задание А7б
 дано | Робот в конце блока (в правом верхнем углу)
 надо | Робот в начале следующего блока
нач
| вверх;вверх
| вправо;вправо;вправо
| вниз;вниз
кон
```

В задании А7в больше подходит закрашивание блока N1. В этом случае обход стены может быть таким:

```
алг обход стены | задание А7в
 дано | Робот в конце блока (в правом нижнем углу)
 надо | Робот в начале следующего блока
нач
| вниз;вниз
| вправо;вправо;вправо
| вверх;вверх;вверх;вверх;вверх;вверх
кон
```

А в задании А7г больше подходит закрашивание блока N2. В этом случае обход стены может быть таким:

```
алг обход стены | задание А7г
 дано | Робот в конце блока (в правом верхнем углу)
 надо | Робот в начале следующего блока
нач
| влево
| вверх;вверх
| вправо;вправо;вправо;вправо;вправо
| вниз;вниз
| вправо
кон
```

А8. Рисование буквы М размером *a* x *b*

Эта задача годится для проведения провероч-

ной работы по темам «Исполнитель Чертежник» и «Вспомогательные алгоритмы». Вот одно из возможных решений:

```
алг буква М (арг вещ а,б)
 дано | перо поднято
 надо | вправо от начального положения...
нач
| опустить перо
| сместиться на вектор(0, б)
| сместиться на вектор(а/2,-б/2)
| сместиться на вектор(а/2, б/2)
| сместиться на вектор(0,-б)
| поднять перо
| сместиться на вектор(а/2,0)
кон
```

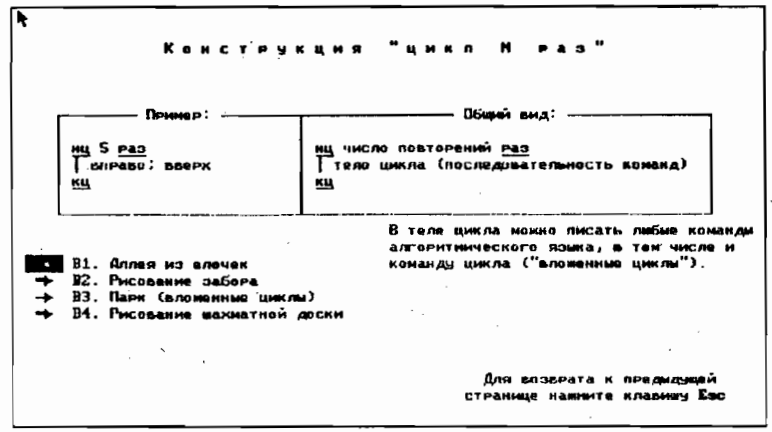
Тема «Цикл N раз»

B1. Аллея из елочек

Демонстрация показывает работу цикла «6 раз» при выполнении алгоритма аллея:

```
алг аллея
 дано | перо Чертежника в начале будущей аллеи
 надо | нарисована аллея из 6 елочек, перо
      | в начале следующей аллеи
нач
| сместиться в точку(2,11)
| нц 6 раз
| | елочка
| кц
| сместиться на вектор(-18,-5)
кон
```

Разбираться в том, как устроен вспомогательный алгоритм елочка, при выполнении этого задания вовсе не обязательно.



B2. Рисование забора

Поскольку алгоритм доска начинает работу в левом нижнем углу доски и заканчивает в правом нижнем углу, при последовательных вызовах алгоритма доска будут нарисованы доски, плотно прилегающие друг к другу. Так что для решения задачи достаточно вставить в основной алгоритм команду

```
нц 18 раз
| доска
кц
```

Для этого нужно подвести курсор в пустую строку за строкой нач, ввести строку

```
нц 18 раз
```

и нажать Enter. После этого на экране появятся строки

```
нц 18 раз
|
кц
```

и останется только ввести команду доска.

В3. Парк (вложенные циклы)

Эта демонстрация показывает, как выполняются вложенные циклы на примере алгоритма парк:

```
алг парк
 дано | перо Чертежника в начале верхней аллеи
      | будущего парка
 надо | нарисован парк из трех аллей
      | по 6 елочек в каждой
нач
| сместиться в точку(2,11)
| нц 3 раз | ВНИМАНИЕ: нельзя писать "нц 3 раза"
| | нц 6 раз
| | | елочка
| | кц
| | сместиться на вектор(-18,-5)
| кц
кон
```

В4. Рисование шахматной доски

В этом задании ученик должен самостоятельно составить алгоритм закрашивания фрагмента поля Робота размером 8 × 8 в шахматном порядке, т.е. в порядке, совпадающем с положением черных (или белых) клеток на шахматной доске. Задача эта не простая, поэтому в задании предусмотрена (по нажатии Ctrl U) выдача указания.

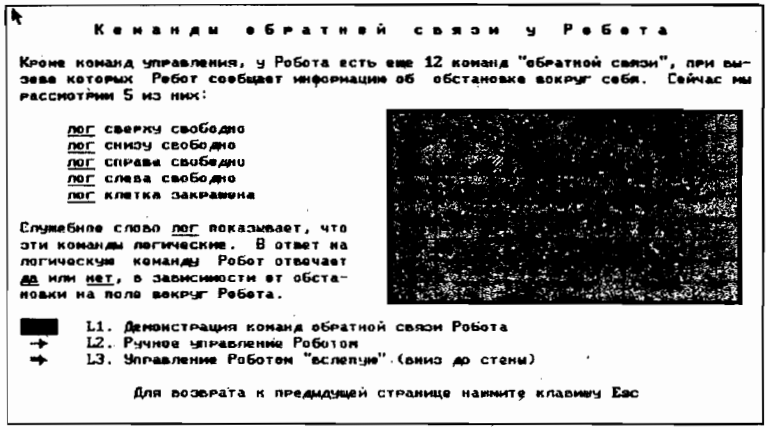
Указание к заданию В4

Заметим, что шахматная доска состоит из 4 × 4 одинаковых фрагментов, каждый размером 2 × 2 клетки. Поэтому при решении удобно использовать вложенные циклы, и общий вид программы получается таким:

```
алг шахматная доска
 дано | Робот в левом верхнем углу квадрата 8x8
 надо | клетки квадрата закрашены
      | в шахматном порядке
нач
| нц 4 раз
| | нц 4 раз
| | | « команды закрашивания фрагмента 2x2 »
| | кц
| | « команды перехода
| |   к следующему ряду фрагментов »
| кц
кон
```

Следуя этому указанию, получаем такой алгоритм:

```
алг шахматная доска
 дано | Робот в левом верхнем углу квадрата...
 надо | клетки квадрата закрашены...
нач
| нц 4 раз
| | нц 4 раз
| | | закрасить; вправо; вниз;
| | | закрасить; вверх; вправо
| | кц
| | вниз; вниз;
| | нц 8 раз
| | | влево
| | кц
| кц
кон
```



Тема «Команды обратной связи у Робота»

L1. Демонстрация команд обратной связи Робота

В этой простейшей демонстрации для двух положений Робота приводятся вопросы ЭВМ и ответы на них Робота. Например:

```
В точке А диалог ЭВМ-РОБОТ может быть
таким:
  сверху свободно?  -> Робот: нет
  справа свободно?  -> Робот: да
  клетка закрашена? -> Робот: нет
```

L2. Ручное управление Роботом

Эта демонстрация очень похожа на ручное управление Роботом (пункт R2) со страницы гипертекста «Исполнитель Робот». На экране появляется пульт управления Роботом, который теперь дополнен командами обратной связи. Нажимая (мышью) кнопки на этом пульте, можно дать Роботу любую из 10 команд. Выполнение одной из первых пяти команд, как и раньше, приводит к перемещению Робота (или закрашиванию клетки). Результат выполнения команд обратной связи (строка «Да» или «Нет») появляется в окне сообщений.

Возврат — нажатие кнопки **Выкл** на пульте управления Роботом.

L3. Управление Роботом «вслепую»

В задании требуется подвести Робота вплотную к стене, которая расположена где-то ниже Робота. Особенность этого задания заключается в том, что положение стены вначале не показано (стена становится видимой только после того, как на вопрос «снизу свободно» будет получен ответ «нет»). Положение стены выбирается случайно при каждом новом запуске, так что без использования команд обратной связи это задание выполнить нельзя.

Для выполнения задания надо последовательно нажимать на пульте кнопки «снизу свободно» и «вниз» до тех пор, пока Робот не окажется в нужном положении. Возврат — нажатие кнопки **Выкл** на пульте управления.

Тема «Цикл "пока"»

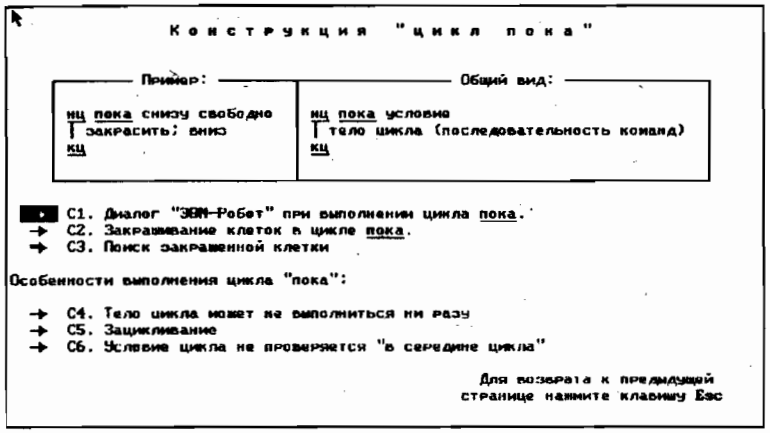
C1. Диалог «ЭВМ-Робот» при выполнении цикла пока

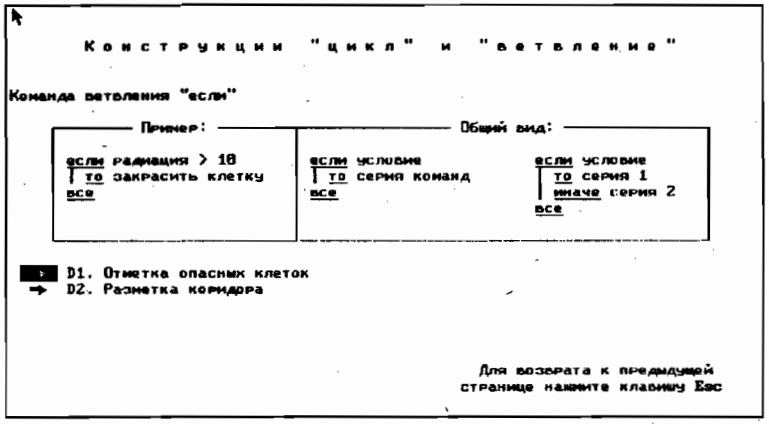
C2. Закрашивание клеток в цикле пока

Эти две демонстрации показывают выполнение двух простейших программ с циклом «пока» и диалог ЭВМ—Робот, который при этом происходит.

C3. Поиск закрашенной клетки

Простое упражнение на использование цикла «пока». Особенность этого упражнения состоит в том, что закрашенная клетка при каждом новом запуске появляется в новом месте (на самом деле ее положение определяется перед каждым выполнением программы заново, с использованием датчика случайных чисел).





C4–C6. Особенности выполнения цикла "пока"

Эти три демонстрации дают наглядное представление о процессе выполнения цикла «пока».

Тема «Команда ветвления "если"»

D1. Отметка опасных клеток

Каждая клетка на поле Робота имеет невидимые простым глазом температуру и радиацию. В этом задании речь идет об алгоритме, отмечающем все клетки коридора с повышенным уровнем радиации. Демонстрация показывает работу команды «если», вложенной в команду «пока», и является подготовительной к следующему заданию.

D2. Разметка коридора

В этом задании ученик должен составить алгоритм, выводящий Робота из коридора и отмечающий клетки, из которых есть выход вверх. Хотя в задании дан конкретный коридор, требуется, чтобы ученик написал универсальный алгоритм, работающий для коридора произвольной длины.

Вот одно из возможных решений:

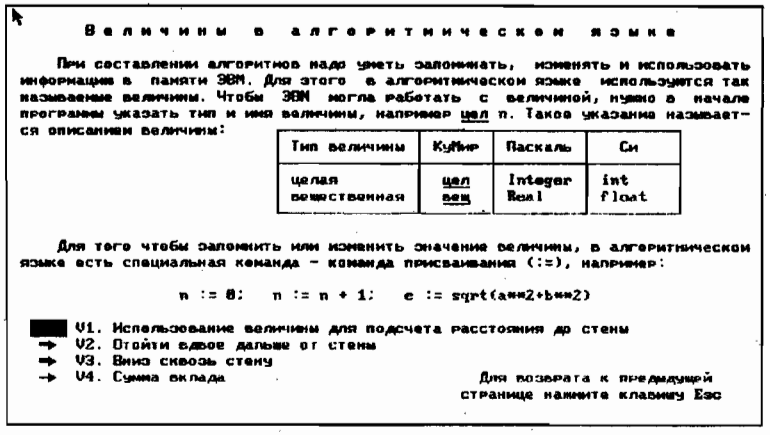
```
алг разметка выходов из коридора
 дано | Робот в левой клетке коридора
 надо | Робот вышел из коридора вправо...
нач
| нц пока снизу стена
|  | если сверху свободно
|  | то
|  |  закрасить
|  | все
|  | вправо
| кц
кон
```

При вводе алгоритма команда «пока» вводится путем последовательного нажатия двух клавиш Esc и П, а команда «если» — по нажатии Esc и Е.

Тема «Величины в алгоритмическом языке»

V1. Использование величины для подсчета расстояния до стены

Демонстрация использования целой величины для подсчета пройденного Роботом расстояния.



V2. Отойти вдвое дальше от стены

В этом задании требуется самостоятельно составить алгоритм. Хотя на экране и задано конкретное расположение Робота и стены, требуется алгоритм, который правильно работает в любой ситуации, для любого расстояния Робота до стены. По нажатии на Ctrl U на экран выдается дополнительное указание по условию задачи:

> Если Робот стоит вплотную к стене, то он должен остаться в исходном положении. Если Робот попадает вплотную к стене за один шаг, то он должен отойти от стены на два шага, и т.д.

Вот одно из возможных решений:

```
алг отойти вдвое дальше от левой стены
 дано | где-то левее Робота есть стена...
 надо | Робот отошел вправо от стены...
нач цел н
| н:=0
| нц пока слева свободно
| | влево; н:=н+1
| кц
| нц 2*н раз
| | вправо
| кц
кон
```


V3. Вниз сквозь стену

В этом задании требуется самостоятельно составить алгоритм. Хотя на экране и задано конкретное расположение Робота и стены, требуется написать алгоритм, который правильно работает в любой ситуации, для стены любой длины. По нажатии на Ctrl U на экран выдается дополнительное указание по условию задачи:

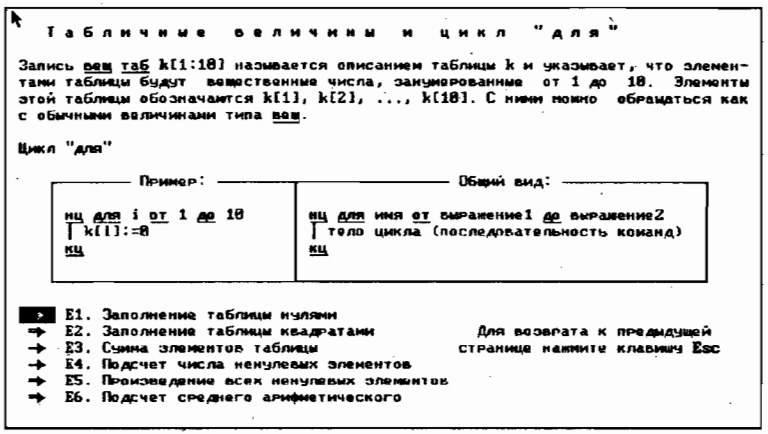
> Будем командовать Роботу вправо до тех пор, пока ниже его стена, запоминая количество сделанных шагов. Затем один раз скомандуем вниз и нужное количество раз — влево. Для запоминания количества сделанных шагов введем переменную типа цел.

Вот одно из возможных решений:

```
алг вниз сквозь стену
 дано | Робот над горизонтальной стеной...
 надо | Робот под стеной, на клетку ниже...
нач цел н
| н:=0
| нц пока снизу стена
| | вправо; н:=н+1
| кц
| вниз
| нц н раз
| | влево
| кц
кон
```

V4. Сумма вклада

В учебнике приведен алгоритм, который только вычисляет сумму вклада, но не строит график роста



суммы вклада. В гипертексте этот алгоритм доделан так, чтобы он строил такой график с помощью Чертежника.

При желании на базе этого алгоритма можно сформулировать несколько дополнительных задач. Например, сравнить, когда сравняются суммы двух вкладов. Первый в 0,5 млн рублей под 20% в месяц, а второй в 1 млн рублей под 12% в месяц.

Тема «Табличные величины и цикл "для"»

E1. Заполнение таблицы нулями

Демонстрация выполнения простейшей программы с использованием табличных величин:

```
алг заполнение нулями (рез цел таб a[1:8])
 дано | a - таблица из 8 элементов
 надо | все элементы таблицы равны 0
нач цел i
| нц для i от 1 до 8
| | a[i]:=0
| кц
кон
```


В процессе демонстрации в окне ввода-вывода показываются последовательные состояния табличной величины a.

E2. Заполнение таблицы квадратами

Простое упражнение на составление программы по образцу демонстрировавшейся в пункте E1. Вот пример правильного решения:

```
алг заполнение квадратами (рез цел таб a[1:8])
 дано | a - таблица из 8 элементов
 надо | a[i] = i**2 для всех i
нач цел i
| нц для i от 1 до 8
| | a[i]:=i**2
| кц
кон
```

E3. Сумма элементов таблицы

E4. Подсчет числа ненулевых элементов

Две демонстрации — подсчет суммы элементов таблицы и числа ненулевых элементов — дают примеры алгоритмов обработки табличной информации. Следует обратить внимание на использование во втором примере вложенных управляющих конструкций (команда ветвления «если» внутри цикла «для»).

E5. Произведение всех ненулевых элементов

В этом упражнении, разумеется, нельзя просто перемножить все элементы таблицы. Правильное решение следующее:

```
алг цел произведение ненулевых (...)
 дано | a - таблица из N элементов
 надо | значение = произведение...
нач
| знач:=1
```

```
  нц для i от 1 до N
  | если a[i]<>0 то знач:=знач+1 все
  кц
кон
```

Трудным местом при выполнении этого упражнения может стать используемый здесь стандартный прием — доопределение функции на таблицу нулевого размера (вначале, до рассмотрения первого элемента, предполагается, что произведение равно 1). Хотя этот же прием был использован в демонстрации Е3, он мог пройти незамеченным (то, что сумма нулевого количества чисел равна нулю, кажется вполне очевидным; с произведением это не так, хотя с математической точки зрения эти случаи ничем не различаются).

Е6. Подсчет среднего арифметического

В этом упражнении надо сначала найти сумму элементов таблицы, а затем разделить ее на число элементов.

Тема «Литерные величины»

S0. Пусть t="кильватер"...

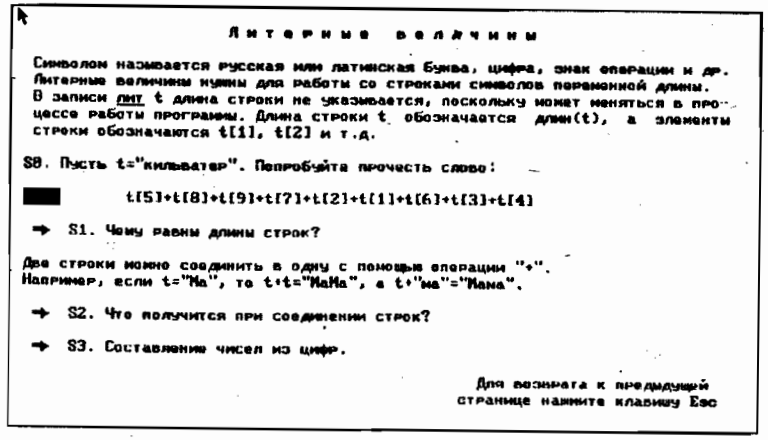
Ответ: "вертикаль". Если ученикам задача понравится, предложите им записать в таком же виде слово "спаниель", где t="апельсин".

S1. Чему равны длины строк?

Ученику последовательно даются строки:

"победа"	"да"
"обеда"	"а"
"беда"	""
"еда"	" "

и он должен указать длину каждой из них. Назначение этого задания — дать представление о пустой



строке и о ее отличии от строки из любого числа пробелов.

S2. Что получится при соединении строк?

S3. Составление чисел из цифр

Это элементарные упражнения на операцию соединения строк "+". Последнее упражнение готовит учеников к записи алгоритмов перевода чисел в двоичную систему на школьном алгоритмическом языке.

Оглавление

Учебное издание

Кушниренко Анатолий Георгиевич
Леонов Александр Георгиевич
Эпиктетов Михаил Геннадьевич
и другие

ИНФОРМАЦИОННАЯ КУЛЬТУРА
Модуль
Класс 9

Методическое пособие

Ответственный редактор *М. Г. Циновская*
Художник обложки *А. В. Кузнецов*
Оригинал-макет подготовил *М. Г. Эпиктетов*
Корректор *Л. А. Александрова*

ЛР № 061622 от 23 сентября 1992 г.
Подписано к печати 03.08.95 г. Формат 84×108 1/32.
Бумага тип. № 2. Гарнитура «Школьная». Печать высокая.
Усл. печ. л. 5,8. Уч.-изд. л. 6,5. Тираж 10 000 экз.
Заказ № 1172.

Издательский дом «Дрофа».
105318, Москва, ул. Щербаковская, 3

Смоленская областная ордена «Знак Почета» типография
им. Смирнова. 214000, г. Смоленск, пр. им. Ю. Гагарина, 2