ISSN 0234—0453

ИНФО

ИНФОРМАТИКА И ОБРАЗОВАНИЕ

2 1991

О новых программах центра читайте на стр. 12, 119

Научно-методический журнал
Государственного комитета СССР
по народному образованию

Издается с августа 1986 г.
Выходит шесть раз в год
Москва «Педагогика» 1990

ИНФ
ИНФОРМАТИКА И ОБРАЗОВАНИЕ

2 '91
март
апрель

Содержание

Редактор отдела *А. Кравцова*
Научный редактор *Н. Копытина*
Зав. редакцией *Н. Игнатова*
Художественный редактор *Л. Коновалова*
Корректор *О. Пурлова*

Сдано в набор 23.01.91. Подписано в печать 5.03.91. Формат 70×100 1/16. Бумага офсетная № 1. Печать офсетная. Усл. печ. л. 10,40. Усл. кр.-отт. 42,88. Уч.-изд. л. 13,04. Тираж 55770 экз. Заказ 82. Цена 1 руб. 20 коп. Издательство «Педагогика» Академии педагогических наук СССР и Государственного комитета СССР по печати.

Почту направлять по адресу: 119034, Москва, Смоленский б-р, д. 4.

Издательство «Педагогика». Журнал «Информатика и образование». Адрес редакции: Лефортовский пер., д. 8.
Телефон: 261-11-29.

Ордена Трудового Красного Знамени Чеховский Полиграфический комбинат Государственного комитета СССР по печати.
142300, г. Чехов Московской обл.

Выпускные экзамены по информатике в XI классе

Процесс становления курса «Основы информатики и вычислительной техники» естественным образом поставил вопрос о введении выпускного экзамена по информатике. Это особенно важно, так как Государственная программа по информатике 1985 г. рассчитана на безмашинный вариант преподавания и морально устарела. Преподавание курса по машинному варианту ведется, по крайней мере, по трем различным учебникам (В. А. Каймин, А. Г. Кушниренко, А. Г. Гейн). Экзаменационные билеты, содержащие базовый инвариант курса, могли бы стать необходимым ориентиром для учителей, определяющим требования к знаниям и умениям учащихся.

Введение определенного государственного стандарта на уровень знаний и умений учащихся по информатике необходимо также и для согласования уровня требований школы и вузов. Уже в прошлом году некоторые вузы ввели вступительный экзамен по информатике, и очевидно, что этот процесс будет расширяться.

В 1989/90 учебном году по желанию учащихся и инициативе учителей информатики были проведены пробные выпускные экзамены в некоторых школах Москвы. Результаты экзаменов подтвердили правильность данного шага.

В соответствии с объяснительной запиской Министерства народного образования РСФСР (Информационный сборник МНО РСФСР. 1990. № 31) в перечень экзаменов по выбору официально введен экзамен по основам информатики и вычислительной техники.

Там же опубликованы экзаменационные билеты, ориентированные на преподавание курса по безмашинному варианту по учебнику А. П. Ершова «Основы информатики и вычислительной техники», ч. I и II (1985, 1986 гг.). Во многих школах преподавание курса информатики ведется с учетом использования ПЭВМ, что требует разработки варианта выпускных экзаменационных билетов, рассчитанных на машинный вариант преподавания курса.

Ниже приводятся билеты, разработанные в МГИУУ, которые были обсуждены среди учителей и методистов Москвы и прошли апробацию на выпускных экзаменах в 1989/90 учебном году.

Экзамен проводится в устной форме, в каждом билете три вопроса. Первый вопрос выявляет уровень знаний учащихся по наиболее существенным, узловым проблемам информатики, второй — требует определенных знаний в области алгоритмизации и программирования, а третий — проверяет умение учащихся применять знания на практике. В процессе решения задач учащиеся отлаживают программы на компьютерах.

Билеты снабжены комментариями, в которых указаны параграфы по различным учебникам, содержащие рекомендации учащимся при подготовке к экзаменам. Здесь же приведена подборка примерных экзаменационных задач.

Содержание большинства экзаменационных вопросов изложено во всех трех учебниках информатики. Некоторые вопросы внесены дополнительно, так как содержание курса меняется быстрее, чем сами учебники. Так, первый вопрос билета № 18 касается понятия файла, являющегося, по нашему мнению, одним из центральных понятий курса информатики.

Литература

1. *Каймин В. и др.* Основы информатики и вычислительной техники: Пробное учебное

пособие для X—XI классов средней школы. М.: Просвещение, 1989.
2. *Ершов А., Кушниренко А. и др.* Основы информатики и вычислительной техники: Пробный учебник для средних учебных заведений. М.: Просвещение, 1988.
3. *Гейн А. и др.* Основы информатики и вычислительной техники: Пробное учебное пособие: Свердловск: Изд-во Уральск. ун-та, 1989.

Билет 1

1. ЭВМ: устройство, назначение, применение.

[1] § 9. Состав персональных ЭВМ.
§ 10. Правила работы.
[2] § 2. ЭВМ — универсальная машина для обработки информации.
§ 16. Основные устройства ЭВМ.
[3] § 1. Что умеет ЭВМ.
§ 35. Как устроена ЭВМ. Микропроцессорные системы.
§ 36. Где применяются ЭВМ.

2. Переменная: имя и значение. Идентификатор.

[1] § 17. Алгоритмы и решение задач.
[2] § 11. Величины в алгоритмическом языке.
[3] § 34. Представление информации в памяти ЭВМ.

3. Задача на вычисление корня уравнения на данном интервале.

Определить корень уравнения $2x-\cos x=0$.

Билет 2

1. Информация. Единицы измерения информации.

[1] § 2. Информация и ее виды.
§ 3. Количество информации.
[2] § 1. Информация, ее передача, запоминание и кодирование.
[3] § 28. Информация и измерение ее количества.

2. Типы переменных (целый, вещественный, символьный, логический).

[1] § 31. Переменные и массивы в Бейсике.
[2] § 11. Величины в алгоритмическом языке.
[3] § 26. Символьные переменные и операции над ними в Бейсике.

3. Задача на построение графика функции. Построить график функции $y=x^2$.

Билет 3

1. Понятие алгоритма. Исполнители алгоритмов.

[1] § 17. Алгоритмы и решение задач.
[2] § 3. Алгоритмы управления Роботом.
[3] § 4. Понятие алгоритма.
§ 5. Исполнители алгоритмов.

2. Функции обработки символьных переменных.

[1] § 31. Переменные и массивы в Бейсике.
[2] § 14. Символьные и литерные переменные.
[3] Глава 8. Символьные переменные.

3. Задача на обработку символьного массива.

Заполнить массив буквами русского алфавита и распечатать его на экране.

Билет 4

1. Разветвляющийся алгоритм.

[1] § 20. Выбор действий в алгоритмах.
[2] § 10. Условия, команды ветвления и контроля.
[3] Глава 3. Ветвления в алгоритмах. Блок-схемы и словесные описания ветвлений.

2. Операторы ввода-вывода информации. Дополнительные возможности вывода на экран.

[1] § 19. Средства ввода-вывода информации.
[2] § 15. Команды ввода-вывода информации в алгоритмическом языке.
[3] § 20. Основные команды Бейсика.

3. Задача на заполнение двумерного массива и вывод его на экран.

Заполнить массив (3×3) произвольной числовой последовательностью и распечатать его в виде таблицы на экране.

Билет 5

1. Циклический алгоритм.

[1] § 21. Повторение действий в алгоритмах.
[2] § 8. Команды повторения ДЛЯ.
§ 9. Команды повторения ПОКА.
[3] Глава 4. Циклическая форма организации действий в алгоритмах. Блок-схемы и словесные описания циклов.

2. Оператор присваивания. Арифметические выражения.

[1] § 17. Алгоритмы и решение задач.
§ 25. Язык программирования Бейсик.

[2] § 11. Величины в алгоритмическом языке, команда присваивания.

[3] § 20. Основные команды языка Бейсик.

3. Задача на вычисление с проверкой правильности исходных данных.

Вычислить координату и скорость тела, брошенного с высоты H_0 со скоростью V_0 вертикально вверх, в заданный момент времени. Предусмотреть проверку правильности исходных данных.

Билет 6

1. Конструирование алгоритмов методом последовательной детализации. Вспомогательный алгоритм.

[1] § 22. Вспомогательные алгоритмы.
§ 38. Пошаговая детализация алгоритмов.

[2] § 5. Вспомогательные алгоритмы.

[3] Глава 5. Вспомогательные алгоритмы.

2. Условный оператор, полный и неполный варианты.

[1] § 21. Выбор действий в Бейсике.

[2] § 10. Условия, команды ветвления и контроля.

[2] § 21. Ветвления в Бейсике.

3. Задача на нахождение элементов массива, отвечающих заданному условию.

Заполнить массив $M(9)$ числами в интервале от 0 до 100, найти элементы, значение которых меньше 50.

Билет 7

1. Этапы решения задач на ЭВМ.

[1] § 33. Этапы решения задач.
§ 34. Математическая постановка задачи.
§ 35. Методы решения задач.
§ 36. Сценарии работы с ЭВМ.

[2] § 20. ЭВМ — универсальное устройство обработки информации.

[3] § 2. Этапы решения задач с помощью ЭВМ.
§ 3. Пример математической модели.

2. Цикл, тело цикла, параметры цикла.

[1] § 29. Повторение действий в Бейсике.

[2] § 8. Команды повторения ДЛЯ
§ 9. Команды повторения ПОКА

[3] § 22. Циклы в Бейсике.

3. Задача на создание диалоговой программы.

Создать программу-словарь терминов по информатике.

Билет 8

1. Логическое сложение. Базовый логический элемент «ИЛИ».

[1] § 6. Элементы математической логики.
§ 50. Логические элементы ЭВМ.

2. Стандартные функции. Определение функции пользователя.

[1] § 25. Язык программирования Бейсик.

[2] § 12. Результаты алгоритмов и алгоритмы-функции.

[3] Глава 7. Язык программирования Бейсик.

3. Задача на использование подпрограмм.

Нарисовать на экране многоэтажный дом.

Билет 9

1. Логическое умножение. Базовый логический элемент «И».

[1] § 6. Элементы математической логики.
§ 50. Логические элементы ЭВМ.

[2] § 17. Взаимодействие памяти и процессора. Принцип хранимой программы.

2. Подпрограмма, обращение к ней, возврат из подпрограммы.

[1] § 30. Организация подпрограмм в Бейсике.

[2] § 9. Результаты алгоритмов и алгоритмы-функции.

[3] § 23. Подпрограммы и определяемые функции.

3. Задача на моделирование случайного процесса.

Нарисовать на экране «звездное небо».

Билет 10

1. Логическое отрицание. Базовый логический элемент «НЕ».

[1] § 6. Элементы математической логики.
§ 50. Логические элементы ЭВМ.

2. Условия с логическими связками.

[1] § 28. Выбор действий в Бейсике.
[2] § 10. Условия, команды ветвления и контроля.
[3] § 21. Ветвления в Бейсике.

3. Задача на движение графических объектов.

Нарисовать на экране движущуюся точку.

Билет 11

1. Информатизация общества. Развитие вычислительной техники.

[1] Глава 8. ЭВМ в развитии общества.
§ 56. История развития ЭВМ.
[2] Глава 3. Применения ЭВМ.
§ 19. История вычислительной техники. Поколения ЭВМ.
[3] Глава 9. Информатика — отрасль производства.

2. Операторы графики.

[1] § 27. Графические программы на Бейсике.
[2] § 4. Чертежник и работа с ним.
[3] § 16. Последовательное построение алгоритмов.

3. Задача с использованием логических функций.

Распечатать на экране таблицу истинности логической функции «И».

Билет 12

1. Представление об операционной системе.

[1] § 54. Программное обеспечение ЭВМ.
[2] § 18. Программное обеспечение ЭВМ.

2. Массивы. Различные способы заполнения массивов.

[1] § 23. Организация массивов в алгоритмах.
§ 31. Переменные и массивы в Бейсике.
[2] § 13. Табличные величины и работа с ними.
[3] Глава 6. Организация данных.

3. Задача на отыскание площади геометрической фигуры.

Определить площадь треугольника по известным сторонам.

Билет 13

1. Машинная графика. Назначение и основные функции.

[1] § 12. Графика и черчение.
[2] § 25. Компьютерное проектирование и производство.

2. Операторы хранения и чтения данных в программе. Повторное чтение.

[1] § 31. Переменные и массивы в Бейсике.

3. Задача на обработку символьных переменных.

Создать программу, генерирующую слова-«перевертыши».

Билет 14

1. Информационные сети. Компьютерная почта.

[1] § 64. ЭВМ и информационные коммуникации.
[2] § 32. Информационные системы.
§ 26. Компьютеры в системах связи.

2. Представление о десятичной и двоичной системах счисления.

[1] § 2. Информация и её виды.
[2] § 1. Информация, ее передача и кодирование.
[3] § 33. Системы счисления.

3. Задача на обработку числового массива.

Заполнить массив последовательностью чисел и подсчитать их сумму.

Билет 15

1. Системы управления базами данных (СУБД). Назначение и основные функции.

[1] § 13. Поиск информации.
[2] § 23. Информационные системы.
[3] § 29. Информационно-поисковые системы.

2. Программные средства организации диалоговых и игровых программ (управление с помощью клавиш).

[1] § 32. Диалоговые программы.

3. Задача на создание программы контроля знаний.

Создать программу-тренажер английского языка.

Билет 16

1. Электронные таблицы. Назначение и основные функции.

[1] § 15. Численные расчеты.

[2] § 22. Электронные таблицы.
[3] § 31. Электронные таблицы.

2. Внешняя память (дисководы, диски).

[1] § 52. Внешние устройства ЭВМ.
[2] § 16. Основные устройства ЭВМ.

3. Задача на вывод информации на экран.

Вывести на экран текст в заданном формате.

Билет 17

1. Текстовый редактор. Назначение и основные функции.

[1] § 11. Редактирование текстов.
[2] § 21. Обработка текстов с помощью компьютера.
[3] § 30. Редактирование текстов с помощью ЭВМ.

2. Матричный принцип печати. Принтер.

[1] § 52. Внешние устройства ЭВМ.
[2] § 16. Основные устройства ЭВМ.

3. Задача на моделирование физического процесса.

Смоделировать на экране атом водорода. Предусмотреть движение электрона по орбите.

Билет 18

1. Представление о файле.

2. Датчики случайных чисел, моделирование случайных процессов.

3. Задача на сортировку массива.

Отсортировать числовой массив по возрастанию.

Н. УГРИНОВИЧ, зав. кабинетом информатики МГИУУ

И. ОРЕШКОВ
председатель секции ССО Научно-методического совета информатизации образования Гособразования СССР
Н. ЗЕЛЕНСКАЯ
Государственный комитет по делам науки и высшей школы РСФСР

Новому этапу информатизации — новую программу

Прошедшее пятилетие в сфере образования характеризуется существенным прогрессом в области информатизации учебного процесса. Уже можно говорить об известном насыщении техникумов и училищ вычислительной техникой, правда, разнотипной, неунифицированной. Но, во всяком случае, акценты сместились, и весьма заметно, в сторону программного обеспечения, хотя эффективное использование имеющегося парка вычислительной техники тормозится отсутствием педагогических программных средств, а также слабым знанием возможностей системных программных средств. Ниже публикуется для обсуждения проект программы, во введении которого достаточно подробно излагаются мотивы, которыми руководствовались разработчики программы, определяя ее содержание.

Следует отметить, что проект программы по предмету ОИВТ был рассмотрен и одобрен членами секции среднего специального образования Научно-методического совета информатизации образования Гособразования СССР на очередном заседании (14—17 мая 1990 г., г. Новосибирск). Коллектив разработчиков обращается к преподавателям средней специальной школы с просьбой направить свои замечания и предложения в адрес редакции.

Основы информатики и вычислительной техники

Проект программы для средних специальных учебных заведений на базе неполного среднего образования

1. Введение

В соответствии с концепцией информатизации средней специальной школы выпускник среднего специального учебного заведения должен быть подготовлен теоретически и практически для информатизации общественных структур в сфере своей профессиональной деятельности, знать и активно применять в работе средства электронно-вычислительной техники.

В настоящее время в нашей стране выбрано в качестве основного направление на использование PC-совместимых ЭВМ и сетей на их базе.

Данная программа рассчитана на практическое изучение и применение программного обеспечения ПЭВМ и может быть использована для реализации общего, первого и второго уровней базовой подготовки учащихся средних специальных учебных заведений в области применения средств вычислительной техники.

В зависимости от уровня базовой подготовки определены следующие цели компьютеризации:

общий уровень — дать специалисту начальные сведения по информатике, вычислительной технике и программированию, научить пользоваться универсальными пакетами прикладных программ;

первый уровень — дать специалисту необходимые сведения по информатике и вычислительной технике, методике математического моделирования и решения задач на ЭВМ; привить навыки формализации задач для их дальнейшей реализации на ЭВМ, использования пакетов прикладных программ по специальности, самостоятельного освоения нового прикладного математического обеспечения;

второй уровень — дать специалисту широкие знания в области информатики и вычислительной техники; научить программированию, работе с операционными системами, самостоятельному освоению новых типов компьютеров; научить разрабатывать прикладное математическое обеспечение.

Так как курс ОИВТ является пока единственным предметом, непосредственно связанным с изучением информатики для специальностей, относящихся к общему уровню, то целесообразно при его изучении больше внимания уделить практической работе с пакетами прикладных программ по специальности. В соответствии с этим необходимо перераспределить выделенные часы по разделам.

Например, для специальностей второй группы «Право и документирование» больше внимания уделить изучению текстовых редакторов и работе с системами управления базами данных. Еще более дифференцированный подход необходим при выборе изучаемого материала для специальностей третьей группы «Просвещение». Так, для специальности 0303 «Физическая культура» целесообразно изучить работу с системами, обслуживающими спортивные соревнования, обработки результатов тестирования спортсменов. Для специальности 0305 «Музыкальное воспитание» и для специальностей пятой группы, связанных с изучением музыки, следует больше внимания уделить программированию мелодий, ввести раздел «Работа с музыкальными редакторами», а для специальностей изобразительного цикла — работе с графическими редакторами и программированию графических изображений. Для специальности 0520 «Библиотечное дело» можно рекомендовать изучение работы с различными информационно-справочными системами, СУБД. Для специальностей шестой, девятой, десятой групп упор можно сделать на работу с пакетами обработки данных, полученных путем измерений и передаваемых автоматически в ЭВМ. Для специальностей, связанных с техническим обслуживанием и ремонтом машин, приборов, других видов техники, рекомендуется уделить внимание работе с различного рода диагностическими программами, тренажерами, позволяющими обнаруживать и локализовывать неисправности, моделировать те или иные нестандартные ситуации и производить контроль действий обучаемого в этих ситуациях.

Изучение пакетов прикладных программ по специальности для первого уровня базовой подготовки целесообразно отнести в специальные предметы, а в данном курсе акцент сделать на изучение методики математического моделирования и решения задач на ЭВМ — от постановки задачи, алгоритмизации до разработки программ на языках высокого уровня и методике испытаний, с тем чтобы эти знания, умения и навыки учащиеся могли применить на лабораторных работах по предметам специального цикла в курсовом и дипломном проектировании.

Для многих специальностей первого и второго уровней базовой подготовки предусмотрена и специальная подготовка в области информатики. Это реализуется введением различных специальных курсов, связанных с программированием, математическим моделированием, изучением системы программного обеспечения, технических средств ЭВМ и автоматизированных систем. Поэтому курс ОИВТ должен стать базой для изучения соответствующих специальных предметов.

Здесь можно рекомендовать более глубокое изучение арифметических и логических основ программирования, принципов организации и работы узлов ЭВМ, методов разработки алгоритмов, программирования на одном из языков высокого уровня.

Особенностью данной программы является ее ориентация на постоянное использование ПЭВМ, обучающих систем, хорошо разработанной программной поддержки лабораторного практикума. Акцент сделан на практическое знакомство с системой программного обеспечения современных ПЭВМ.

С целью своевременного включения вопросов, связанных с последними достижениями в области использования средств ВТ, учета специфических потребностей предприятий отрасли и возможностей учебного заведения, предметным (цикловым) комиссиям необходимо вносить изменения в примерный тематический план.

2. Примерный тематический план

№ п/п	Наименование раздела*
	Введение
1.	Архитектура
2.	Программное обеспечение ПЭВМ, назначение ее компонента
3.	Операционные системы
4.	Редакторы текстов
5.	Графические редакторы
6.	Основы алгоритмизации и решение задач с помощью ПЭВМ
7.	Системы программирования
8.	Электронные таблицы
9.	Базы данных
10.	Перспективы использования средств ЭВМ

* Общее количество часов — 100, в том числе на лабораторные и практические занятия — 50.

3. Содержание предмета

Введение

Цель и задачи предмета. Что такое информатика. Информация и ее обработка. Виды информации, обрабатываемые с помощью ЭВМ: числа, тексты, рисунки, чертежи, таблицы, документы. История развития вычислительной техники и программного обеспечения. Изменение характера использования ЭВМ от поколения к поколению. Современный уровень производства и применения вычислительной техники.

Раздел 1. Состав ПЭВМ, назначение и технические характеристики ее устройств.

Состав ПЭВМ. Назначение процессора. Три вида памяти: постоянная, основная (оперативная), внешняя, их назначение. Единицы измерения информации: бит, байт, килобайт, мегабайт. Понятие адреса памяти. Внешняя память на сменных гибких магнитных дисках (дискетах). Основные характеристики дискет, правила обращения с ними. Назначение жестких магнитных дисков, их характеристики. Клавиатура. Группы клавиш клавиатуры, их назначение.

Дисплей. Алфавитно-цифровые и графические дисплеи, цветные и монохромные, их назначение и характеристики.

Печатающие устройства, синтезаторы на дисплеи, их возможности печатать текстовую и графическую информацию. Управление печатающим устройством с помощью клавиш на панели устройства.

Лабораторная работа № 1

Порядок включения и выключения ПЭВМ. Меры безопасности при работе на ПЭВМ. Работа с клавиатурным тренажером.

Раздел 2. Системное и прикладное программное обеспечение ПЭВМ.

Компоненты программного обеспечения ПЭВМ: операционные системы, системы программирования, системы диагностики и контрольно-наладочные тесты, пакеты прикладных программ, их назначение и краткая характеристика.

Раздел 3. Операционная система

Назначение операционной системы. Состав операционной системы: файловая система, драйверы внешних устройств, процессор командного языка.

Понятие файла. Спецификация файла, имя файла, расширение имени, шаблоны имен файлов. Каталоги файлов. Просмотр и печать каталога. Копирование файлов. Просмотр содержимого файла. Вывод содержимого файла на печатающее устройство. Переименование файла. Удаление файлов. Создание и удаление подкаталога, вход в подкаталог и выход из него, указание маршрута поиска файла. Командные файлы. Загрузка операционной системы. Назначение программ-знакогенераторов. Основная и расширенная таблица кодов. Подготовка дисков к работе: форматирование рабочих и системных дисков, их проверка. Копирование дисков.

Диалоговые надстройки над операционной системой.

Лабораторная работа № 2

Изучение команд операционной системы. Создание подкаталогов, командных файлов.

Лабораторная работа № 3

Работа с диалоговой надстройкой над ДОС.

Лабораторная работа № 4

Работа с обучающей программой, позволяющей проконтролировать знание учащимся операционной системы.

Раздел 4. Редакторы текстов

Настольные издательские системы на базе ПЭВМ. Типовой набор команд редактора текстов. Считывание и запись текстового файла. Перемещение курсора по тексту. Стирание, вставка и замена символов, слов, строк. Блочные операции: выделение блока, копирование, перемещение, удаление блоков, запись блока на диск и вывод на печать. Поиск в тексте и замена. Задание формата текстового документа, форматирование. Задание системных параметров работы редактора. Перекодирование файлов. Проверка правописания.

Лабораторная работа № 5

Создание и корректировка текстового файла.

Лабораторная работа № 6

Блочные операции. Поиск в тексте и замена. Форматирование текста.

Лабораторная работа № 7

Вывод документа на печать в требуемом формате и использование возможностей печатающего устройства печатать различными шрифтами.

Раздел 5. Графические редакторы

Возможности графических редакторов. Назначение и работа с манипулятором «мышь». Создание и корректировка изображений и текста. Блочные операции: копирование, перемещение, зеркальное отображение, повороты, растяжение, сжатие, удаление и изменение фрагментов изображения. Создание зрительных и звуковых эффектов. Возможности создания мультипликации.

Лабораторная работа № 8

Создание и корректировка изображения. Выдача изображений на печать.

Лабораторная работа № 9

Выполнение блочных и файловых операций.

Лабораторная работа № 10

Создание зрительных и звуковых эффектов.

Лабораторная работа № 11

Создание кадров мультипликации и их «проигрывание».

Раздел 6. Основы алгоритмизации и решение задач с помощью ЭВМ

Этапы решения задачи: построение математической модели задачи, разработка алгоритма, разработка программы, ее отладка и испытание на контрольных примерах. Примеры задач и их моделей. Понятие алгоритма. Способы описания алгоритмов.

Основные структуры, используемые для описания алгоритмов: следование, ветвление, выбор, цикл **пока**, цикл **повторить**, цикл **для**. Нисходящее проектирование и планирование. Метод пошаговой детализации (последовательного уточнения) разработки алгоритма. Вспомогательные алгоритмы. Подготовка контрольных примеров для проверки алгоритмов в нормальных, экстремальных и исключительных условиях. Выбор программных средств для решения задачи.

Практическое занятие № 1

Построение математических моделей различных задач, определение существенных и несущественных характеристик задачи, определение исходных данных и результатов.

Практическое занятие № 2

Разработка линейных и разветвляющихся алгоритмов.

Практическое занятие № 3

Разработка простейших циклических алгоритмов.

Практическое занятие № 4

Использование вспомогательных алгоритмов.

Раздел 7. Системы программирования

Классификация языков и систем программирования. Краткая характеристика языков. Изучение основ программирования на одном из языков высокого уровня, например Паскале. Структура программы: раздел операторов, комментарии. Стандартные скалярные типы, отрезки типов, перечисляемые типы. Скалярные типы, определяемые пользователем. Выражения, операнды, операции.

Простые операторы: пустой, присваивания, безусловного перехода, процедуры, ввода-вывода. Логические операции и выражения. Условные операторы. Составной оператор. Тип массива. Операторы цикла. Строки и символьные массивы. Стандартные процедуры и функции: преобразования, арифметические, строковые, множества.

Тип записи. Файлы. Текстовые файлы. Процедуры и функции для работы с типизированными файлами. Файлы без типа. Процедуры и функции для работы с нетипизированными файлами.

Построение графических изображений.

Лабораторная работа № 12

Изучение команд меню и команд редактора Паскаля. Создание текста программы. Трансляция в память и на диск. Выполнение и сохранение программ.

Лабораторная работа № 13

Выполнение линейных программ на Паскале с вставкой комментариев и корректировкой операторов ввода-вывода с целью достижения требуемого формата ввода-вывода, улучшения эксплуатационных характеристик программы.

Лабораторная работа № 14

Отладка и испытание разветвляющихся программ.

Лабораторная работа № 15

Отладка и испытание простейших циклических программ.

Лабораторная работа № 16

Изучение работы программ обработки множеств и строковых величин.

Лабораторная работа № 17

Изучение работы программ обработки массивов.

Лабораторная работа № 18

Изучение работы программ создания и обработки текстовых файлов.

Лабораторная работа № 19

Изучение работы программ создания и обработки типизированных и нетипизированных файлов.

Лабораторная работа № 20

Изучение работы программ построения графических изображений.

Раздел 8. Электронные таблицы

Назначение и возможности электронных таблиц. Электронный банк. Координаты ячеек. Активная ячейка. Перемещение курсора. Перемещение и разделение бланка. Строка состояния бланка. Строка подсказки. Строка ввода команд. Ввод текста, формул, команд и подкоманд. Получение справки. Формат ячейки. Содержимое и значение ячейки. Изменение информации в строке ввода. Переход из режима электронного блока в режим команд. Перечень и назначение основных команд. Стандартные функции.

Лабораторная работа № 21

Создание и корректировка простейших отчетов. Изменение формы отображения отчета. Выполнение операций над строками, столбцами и группами ячеек. Получение итогов. Сохранение отчета в файле на диске.

Лабораторная работа № 22

Группировка информации по строкам и столбцам. Распечатка отчета на печатающем устройстве. Изменение режимов вычислений и вывода.

Лабораторная работа № 23

Использование графических возможностей электронных таблиц. Вывод на экран и на печать различного рода диаграмм.

Раздел 9. Базы данных

Назначение системы управления базами данных. Реляционные, иерархические, сетевые базы данных. Создание

и модификация структуры и содержимого файла. Активизация файла. Перемещение по записям файла. Просмотр отдельных записей, групп записей и всего файла. Вывод структуры файла и его записей на печатающее устройство. Полиэкранное редактирование записей файла, вставка, удаление и добавление записей.

Индексирование и сортировка записей файла, их назначение. Поиск информации в отсортированном и индексированном файлах.

Базовые средства программирования. Создание программных файлов и файлов процедур. Активизация файла процедур, выполнение программных файлов и процедур. Присваивание значений переменным памяти. Стандартные функции. Команды ввода-вывода информации. Команды установки режимов работы СУБД.

Лабораторная работа № 24

Изучение возможностей СУБД и использование команды помощи.

Лабораторная работа № 25

Изучение меню СУБД. Работа в режиме «Ассистент». Создание файла. Просмотр и печать его структуры и содержимого. Модификация структуры файла.

Лабораторная работа № 26

Создание индексных файлов. Поиск записей в файле. Просмотр записей индексированного файла. Корректировка записей файла.

Лабораторная работа № 27

Формирование видеограмм и отчетов с помощью экранного «художника».

Лабораторная работа № 28

Работа с программами создания и корректировки файлов.

Лабораторная работа № 29

Работа с программами, выдачи видеограмм и отчетов.

Раздел 10. Перспективы использования средств ЭВТ

Новые технические и программные средства. Понятие о базах знаний, экспертных системах. Интегрированные системы. Перспективы использования ЭВМ в отрасли, где будет трудиться будущий выпускник.

Лабораторная работа № 30

Демонстрация использования ЭВМ в профессиональной области будущего выпускника.

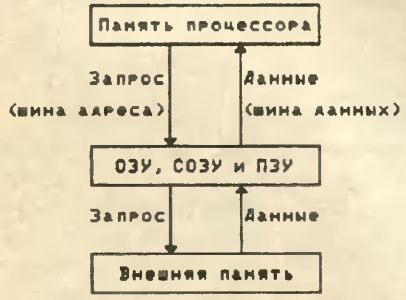
А. АВЕРБУХ

Как от Е-практикума перейти к устройству ЭВМ

7. Разрабатываем структурные принципы организации памяти

Изготавливать всю память из триггеров дорого. К тому же, несмотря на высокое быстродействие, триггер имеет существенный недостаток — он «забывает» информацию при отключении питания. Поэтому при проектировании ОЗУ, ПЗУ и других видов памяти используются физические явления, отличные от тех, что происходят в транзисторе.

Как правило, чем больше объем запоминающего устройства, тем меньше его быстродействие, тем оно дешевле, тем ниже оно располагается в иерархической структуре памяти ЭВМ. На рис. 12 показан процесс **чтения** данных из разных видов памяти. При **записи** данных направления информационных потоков меняются на противоположные.



На схеме видно, что запрос к внешней памяти выполняется ступенчато, как в известном детском стихотворении. «Король, его величество, просил ее величество, чтобы ее величество спросила у молочницы, нельзя ль доставить масло на завтрак королю. Придворная молочница пошла к своей корове...» (рис. 13).

Итак, в ЭВМ применяется много разновидностей запоминающих устройств. Эти устройства отличаются емкостью, быстродействием, работают на разных физических принципах. Однако все запоминающие устройства похожи *логически,* т. е. базируются на однотипных структурных принципах.

Любое запоминающее устройство строится из простейших элементов, каждый из которых хранит один бит информации. Для увеличения быстродействия ЭВМ ее различные части обмениваются информацией большего объема, от 8 до 64 бит у разных типов ЭВМ. Информационный объем сообщения, одновременно передаваемого внутри ЭВМ от одного блока к другому, называется **машинным словом.**

Элементы памяти объединяются в **ячейки,** каждая из которых содержит стандартное число бит, как правило, кратное машинному слову. Каждая ячейка имеет собственный номер — **адрес.** Таким образом, понятия величины в ЭВМ, передачи параметров, исполнения команд присваивания, ввода и вывода наполняются материальным содержанием.

В самом понятии величины открываются новые грани. Мы теперь можем взглянуть на величину «глазами ЭВМ». Как видим, имя величины есть лишь непонятное для ЭВМ, но удобное для человека сочетание букв. Встречая имя величины, ЭВМ *идентифицирует* номер ячейки, содержащей заданную величину. Поэтому имена в информатике часто называют *идентификаторами*.

Значение величины для ЭВМ — это содержимое ячейки памяти с определенным адресом. Вид величины позволяет ЭВМ прогнозировать, каким образом и в какой момент последуют операции записи в память и извлечения из памяти значения величины. Тип величины информирует ЭВМ о формате записи значения в ячейке. Например, для хранения величины типа лог достаточно одного бита.

При использовании языка высокого уровня человек может не знать, в каких ячейках хранятся значения величин алгоритма. Процессор сам следит за распределением памяти, устанавливает соответствия между именами и адресами величин. Но во многих языках программирования есть возможность прямого доступа к памяти, когда человек оперирует не именами, а адресами величин.

Теперь прервем увлекательный процесс проектирования ЭВМ, немного отдохнем и в это время дадим исторический обзор развития вычислительной техники.

8. Поколения, семейства и брак

Первый прообраз современной ЭВМ был построен на лампах в конце 30-х гг. С точки зрения сегодняшнего дня, это был бракованный экземпляр, дающий сбои каждые 7 мин. Впрочем, у него было одно неоспоримое достоинство: он служил прекрасным обогревателем помещения и зимой в машинном зале отопление не включали.

От этого обогревателя и произошли ЭВМ, историю которых принято, как у людей, делить на поколения. Правда, время жизни человеческого поколения примерно 80 лет, а ЭВМ — 10 лет.

Первая «настоящая» ЭВМ была создана в 1945 г. Она получила имя ENIAC (Electronic Numerical Integrator And Calculator), имела 20 ячеек памяти и выполняла 5 тыс. операций в секунду. ENIAC была представителем первого поколения.

Деление ЭВМ на поколения достаточно условно. Все зависит от критерия. В учебнике в качестве критерия взята элементная база. Существуют и другие критерии, например быстродействие. Деление на четыре поколения традиционно, но вряд ли необходимо для подробного изучения в общеобразовательной школе.

Более важно, на наш взгляд, выделить концептуальные этапы развития вычислительной техники (не только электронной). Таких этапов три:

1. Середина XIX в. Чарльз Бэббидж. Разделение арифметико-логического и запоминающего устройства. Появление терминов «адрес» и «код операции». Первые перфокарты. Изобретение команды условного перехода.

2. 1945 г. Джон фон Нейман. Современная архитектура вычислительной машины. Принцип хранимой программы.

3. 1984 г. Отход от традиционной фон-неймановской архитектуры. (Микропроцессор, занимающий площадь менее 1 $см^2$ и состоящий из нескольких связанных между собой процессоров, работающих параллельно.)

Раз есть прародитель и поколения, то должны быть и семейства. Принято выделять четыре семейства. (Как и любая классификация в информатике, деление ЭВМ на семейства весьма условно и устарело в момент принятия.)

Самые мощные — супер-ЭВМ. Длина машинного слова в них составляет 64 бита и более. У них большой объем ОЗУ, измеряемый десятками Мбайт. Второе семейство — большие ЭВМ. Обычно 32-разрядные и с меньшим объемом ОЗУ.

Затем — мини-ЭВМ. Длина машинного слова 16 или 32 бит и сравнительно небольшое ОЗУ. Перечисленные три семейства объединяют ЭВМ, имеющие достаточно большие ресурсы и предназначенные для одновременной работы нескольких пользователей (мультипрограммный режим).

И наконец, микро-ЭВМ. Они, как

правило, являются персональными, т. е. предназначены для одного пользователя. В настоящее время выпускается так много разновидностей ПЭВМ, что трудно обобщить их технические характеристики. Машинное слово содержит от 8 до 32 бит. Объем ОЗУ колеблется от десятков Кбайт до одного Мбайта и даже более того. (Объем ОЗУ связан с разрядностью процессора и размером шины адреса. Для 16-разрядной шины адреса максимальный объем ОЗУ равен 2^{16} байт=2^6К байт=64К байта. Однако имеется техническая возможность увеличить объем адресуемой памяти, что и делается в большинстве ПЭВМ.)

Пришло время вернуться к процессору и разобраться, зачем мы его собрали.

9. Процессор выполняет программу

Наш процессор умеет выполнять арифметические операции, операции сравнения и логические операции. Часть процессора, которую мы спроектировали, назовем **арифметико-логическим устройством (АЛУ).**

Процессор выполняет указанные операции не по собственной инициативе, а повинуясь написанной человеком программе. Она в память процессора, конечно, не поместится, поэтому будем считать, что машинная программа хранится в оперативной памяти (иногда и в ОЗУ вся программа не помещается).

Подобно программе, написанной на алгоритмическом языке, машинная программа представляет собой последовательность команд. Но, в отличие от известных нам команд, машинная команда имеет стандартный формат и строго фиксированную длину, кратную машинному слову.

В машинной команде указывается операция, которую должен выполнить процессор, и адреса величин, над которыми должна быть произведена эта операция (рис. 14).

КОМАНДА	
Код операции	Адреса величин

В зависимости от количества величин, указанных в машинной команде, процессоры делятся на трехадресные, двухадресные, полуторадресные, одноадресные и безадресные (стековые). На рис. 15 а) и б) показаны форматы команд одноадресного и трехадресного процессоров.

КОМАНДА	
Код операции	Адрес аргумента или результата
Что делать?	Откуда взять или куда положить?

а)

КОМАНДА			
Код операции	Адрес первого аргумента	Адрес второго аргумента	Адрес результата
Что делать?	Откуда взять?		Куда положить?

б)

Поскольку во время выполнения программы необходимо часто обращаться к оперативной памяти то возложим на процессор дополнительные функции: чтение команды из памяти; извлечение из памяти значений величин, указанных в команде; преобразование команды в последовательность команд АЛУ; запись результатов выполнения команды в память; вычисление порядкового номера следующей команды. Ответственную за эти функции часть процессора назовем **устройством управления (УУ).**

Устройство управления содержит два важных для нас регистра: Счетчик Команд (СК) и Слово Состояния Процессора (ССП).

В специальной литературе по ЭВМ коллективного пользования (серии ЕС и т. п.) аббревиатура «ССП» иногда расшифровывается как «Слово Состояния Программы». Это специальный регистр, предназначенный для обработки прерываний в мультипрограммном режиме. Как правило, это 64-разрядный регистр, первая половина которого описывает текущее состояние процессора, а оставшиеся 32 бита — состояние программы в момент прерывания.

Нас же интересуют всего два бита регистра ССП, хранящие значение зна-

ка результата последней команды S (Signum). Для определенности можно считать, что соответствие содержимого двух битов ССП и значения S задается табл. 9.

Таблица 9

ССП	Значение S
0 0	+1
0 1	0
1 0	-1
1 1	переполнение

Теперь процессор готов к работе. Разберемся, как он это делает, исполняя готовые программы, а затем предложим учащимся самостоятельно составить простые машинные программы. При этом будем пользоваться табл. 10, в которой приведены несколько команд микропроцессора К1801 ВМ2. Для любителей экзотики даны также мнемонические обозначения этих команд на языке ассемблера.

Таблица 10

КОМАНДА	МНЕМОКОД
Стоп	HALT
Очистить X	CLR X
Увеличить X на единицу	INC X
Уменьшить X на единицу	DEC X
Переслать из X в Y	MOV X,Y
Прибавить X к Y	ADD X,Y
Вычесть X из Y	SUB X,Y
Сравнить X с Y	CMP X,Y
Сравнить X с нулем	TST X
Если меньше - переход на D	BLT +D
Если равно - переход на D	BEQ +D
Если больше - переход на D	BGT +D
Если не меньше - переход на D	BGE +D
Если не равно - переход на D	BNE +D
Если не больше - переход на D	BLE +D
Переход на D	BR +D

Итак, имеются процессор и различные запоминающие устройства. Нам осталось, во-первых, подключить устройства ввода-вывода, и, во-вторых, научить процессор ими управлять.

10. Подключаем периферийное оборудование

Прежде всего договоримся о двух вещах. Во-первых, мы не будем изготавливать внешние устройства (т. е. подробно разбираться в том, как они работают), а станем рассматривать их в качестве готовых исполнителей вроде Робота и Чертежника.

Второе. Мы изучим только то оборудование, которое имеется у нас «живьем» в дисплейном классе. О прочих типах внешних устройств будем лишь коротко упоминать.

Остановимся на внешней памяти. К устройствам внешней памяти относится накопитель на гибких магнитных дисках (ГМД, дискетах, флоппи-дисках). Еще его называют дисководом. Он имеется в классе, мы можем увидеть щель для дискет, можем потрогать дискету, найти прорези для магнитных головок.

Кроме дисковода к устройствам внешней памяти относятся стриммер, накопитель на сменных пакетах, накопитель на магнитном барабане и, наконец, накопитель на жестких дисках, так называемый винчестер.

Кстати, о винчестере. Программисты любят придумывать и употреблять жаргонные словечки, часть из которых русского происхождения. Например, слово обеспечéние, с неправильным ударением на четвертом слоге (как «компáс» у моряков или «дóбыча» у шахтеров). Но большинство слов программистского слэнга имеют английское происхождение.

1. Файл. От английского File — папка, картотека.

2. Принтер. От английского Printer — печатающее устройство.

3. Сканнер. От английского Scanner — блок развертки в телевизоре (scan — пристально разглядывать).

4. Джойстик. От английского Joystick — рычаг управления.

5. Стриммер. От английского Streamer — магнитная лента.

6. Плоттер. От английского Plotter — устройство для построения графиков, диаграмм, чертежей по точкам.

7. Диджитайзер. От английского

Digitizer — устройство для кодирования информации (графической) в цифровом виде. (Вспомним binary digit, bit — двоичная цифра.)

8. Винчестер. Есть несколько версий происхождения этого термина. Приведем две из них. Первая: вначале выпускался герметизированный дисковод с двумя дисками по 30М байт, что обозначалось «30/30», подобно калибру охотничьего ружья «Winchester». Вторая версия: дисковод на жестких дисках был разработан в одном из английских филиалов фирмы IBM в городе Winchester.

Не рассматривая подробно работу устройств внешней памяти, приведем список их технических характеристик (табл. 11).

Таблица 11

УСТРОЙСТВА ВНЕШНЕЙ ПАМЯТИ
НАКОПИТЕЛЬ НА СМЕННЫХ ДИСКАХ
Тип носителя (пакет, дискета) Емкость носителя (М байт, К байт) Размер носителя (дюймы) Число дисков Число рабочих поверхностей Число дорожек на поверхности Скорость передачи данных (К байт/с) Плотность записи (бит/мм)
СТРИММЕР
Тип ленты Емкость (М байт) Скорость передачи данных (К байт/с) Плотность записи (бит/мм)
НАКОПИТЕЛЬ НА ЖЕСТКИХ ДИСКАХ
Тип носителя (барабан, "винчестер") Емкость (М байт) Время доступа к записи (мс) Скорость обмена информацией (К байт/с)

Кроме устройств внешней памяти к периферийному оборудованию относятся устройства ввода-вывода. Из огромного перечня этих устройств рассмотрим клавиатуру, монитор и принтер. Эти устройства есть у нас в дисплейном классе.

Клавиатура позволяет набирать отдельные символы, слова, фразы. С ее помощью мы как бы пишем процессору письма. Для отправки письма служит специальная клавиша ввода. Впрочем, ученики уже прекрасно умеют обращаться с клавиатурой, не забывают нажимать клавишу ввода и не сидят перед дисплеем в ожидании ответа на неотправленное письмо.

Остается добавить, что с точки зрения пользователя клавиатуры отличаются друг от друга лишь расположением клавиш.

У отечественных ЭВМ верхние шесть клавиш образуют слово «ЙЦУКЕН», у зарубежных ЭВМ — «QWERTY». Существуют комбинированные клавиатуры, на которых латинские символы расположены в порядке «QWERTY», а русские — «ЙЦУКЕН».

Такое расположение клавиш традиционно. Так было принято на пишущих машинках для замедления (!) работы человека, чтобы механические детали не задевали друг друга.

Поскольку внутри клавиатуры ЭВМ нет механических деталей, то в настоящее время разрабатываются другие расположения клавиш, ускоряющие ра-

Таблица 12

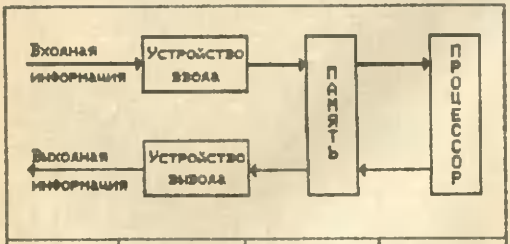
УСТРОЙСТВА ВЫВОДА
МОНИТОР
Тип, число цветов (цветной, монохромный) Размер экрана по диагонали (мм) Формат экрана в текстовом режиме (axb, знаков) Емкость буфера (байтов) Разрешающая способность (число точек на экране в графическом режиме, axb)
ПРИНТЕР (АЦПУ)
Тип (матричный, термографический, струйный, литерный, строчной, лазерный и т.д.) Число цветов Скорость печати (строк/мин или символов/мин) Ширина строки (символов) Способ подачи бумаги (ручной, автоматический) Срок службы красящего элемента (ч) Число одновременных закладок Количество шрифтов Возможность печати букв кириллицы и способ обеспечения этой возможности

боту. Однако широкого распространения эти разработки пока не получили.

В табл. 12 приведен перечень главных технических характеристик монитора и принтера (характеристики монитора определяются параметрами адаптера — блока, соединяющего ЭВМ и монитор).

Кроме того, перечислим еще некоторые устройства ввода-вывода: «мышь», «джойстик», сканнер, диджитайзер, микрофон, световое перо, тактильный экран, плоттер, громкоговоритель. Подробнее мы с ними познакомимся тогда, когда они появятся у нас дома или хотя бы в школьном классе.

Наша ЭВМ почти готова. Имеются процессор, память, устройства ввода-вывода. Осталось научить процессор командовать всей этой аппаратурой. Но предварительно повторим пройденное. Воспользуемся при этом опорным конспектом (табл. 13).



Кодирование информации	Передача информации	Накопление информации	Обработка информации
Виды информации: числа тексты алгоритмы графики рисунки звуки показания приборов нотные знаки химические уравнения	Ввод: клавиатура "мышь" джойстик сканер световое перо дигитайзер микрофон тактильный экран Вывод: монитор принтер плоттер громкоговоритель	Внутренняя память: ОЗУ СОЗУ ПЗУ ППЗУ Внешняя память: флопи-диск "винчестер" стриммер пакет дисков магнитный барабан	арифметико-логическое устройство устройство управления

Решим несколько задач, подобных тем, которые мы решали в самом начале курса. При этом обратим внимание на технические характеристики различных устройств наиболее распространенного сегодня стандарта для персональных ЭВМ.

Кроме того, познакомимся с удобным способом расчетов в информатике, сводящихся к операциям над степенями двойки. При расчетах удобно пользоваться справочной таблицей 14.

Таблица 14

n	2^n	Соотношения между единицами
0	1	
1	2	
2	4	
3	8	1 байт = 2^3 бит
4	16	
5	32	
6	64	
7	128	1К байт = 2^{10} байт
8	256	
9	512	1М байт = 2^{10}К байт = 2^{20} байт
10	1024	1К бит = 2^{10} бит

Задача 1. Информационная емкость человеческой яйцеклетки приблизительно равна 2^{33} бит. На скольких винчестерах (по 20М байт) можно уместить генетическую информацию одного человека?

Решение:

1. Переведем биты в М байты:

2^{33} бит$=2^{30}$ байт$=2^{10}$М байт (1 Гигабайт)

2. Найдем количество винчестеров:

2^{10}М байт / 2×10 М байт$=2^9/10=$
$=51.2$

Ответ: 52 винчестера.

Задача 2. Емкость одного условного печатного листа равна приблизительно 32К байта. Сколько «чистого» времени (без учета смены бумаги) потребуется для распечатки текста одной газеты (2 усл. печ. л.):

а) на матричном (64 символа в секунду) принтере?

б) на лазерном (512 символов в секунду) принтере?

Решение:

1. Представим все числовые данные в виде степеней двойки:

$32=2^5$; $64=2^6$; $512=2^9$

2. На матричном принтере:

$2\times2^5\times2^{10}/2^6=2^{10}=1024$ (с) $\approx$
≈17 (мин)

Ответ: 17 мин.

3. На лазерном принтере:

$2 \times 2^{15} / 2^9 = 2^7 = 128$ (с) ≈ 2 (мин)

Ответ: 2 мин.

Задача 3. Сколько таких газет (без учета иллюстраций) умещает один стриммер емкостью 64М байта?

Решение:

1. Переведем Мбайты в Кбайты:

64М байт $= 2^6$М байт $= 2^{16}$К байт

2. Вычислим число газет (по 64К байта каждая):

$2^{16} / 2^6 = 2^{10} = 1024$

Ответ: 1024 газеты.

Задача 4. Оперативная память IBM PC AT имеет объем 640К байт. Сколько страниц книги поместится в ней, если считать, что на странице 32 строки по 64 символа?

Решение:

1. Число символов на одной странице:

$2^5 \times 2^6 = 2^{11}$ (символов)

2. Информационный объем одной страницы (1 символ требует 1 байт) 2^{11} байт $= 2$К байт.

3. Число страниц: $640/2 = 320$

Ответ: 320 страниц.

Задача 5. Монитор EGA имеет 320 точек по вертикали и 640 точек по горизонтали. Определите объем памяти, необходимой для хранения одного:

а) черно-белого изображения;

б) цветного изображения из 16 цветов.

Решение:

1. Количество точек на экране:

$2^5 \times 10 \times 2^6 \times 10 = 2^{11} \times 100$ (точек)

2. Черно-белое изображение требует 1 бит на каждую точку. Вычислим необходимый объем памяти:

$2^{11} \times 100$ бит $= 2 \times 100$К бит $\times 2^{-2} \times$ $\times 100$К байт $= 25$К байт

Ответ 1: 25К байт.

3. 16 цветов кодируются четырьмя битами. Поэтому объем памяти, необходимый для цветного изображения, вчетверо больше:

$25 \times 4 = 100$К байт

Ответ 2: 100К байт.

Задача 6. Дисковод персональной ЭВМ IBM PS/2 предназначен для флоппи-дисков емкостью 720К байт (3.5 дюйма, около 90 мм). Сколько цветных изображений можно уместить на одном флоппи-диске?

Решение:

1. Так как одно изображение занимает 100К байт, то уместится:

$720/100 = 7.2$

Ответ: 7 изображений.

Итак, мы почти собрали ЭВМ. «Почти», потому что ЭВМ без программ беспомощна, как новорожденный младенец. И подобно тому, как человек умен пропорционально не только «качеству мозгов», но и своему опыту, «умственные способности» ЭВМ определяются не столько качеством аппаратуры, сколько качеством заложенных в нее программ. Вспомните, как дала сбой одна из первых бухгалтерских программ, встретив в ведомости фамилию Стоун-Джексон и принявшись вычитать одно из другого.

Займемся программным обеспечением.

11. ЭВМ = аппаратура + программы

Программное обеспечение — неотъемлемая часть любой ЭВМ, оно продается и поставляется вместе с аппаратурой. Начинающий пользователь, выбирая ЭВМ, руководствуется привычной логикой: чем мощнее, тем лучше. Он внимательно изучает технические показатели — быстродействие, разрядность, объем оперативной памяти. В результате ему часто приходится разрабатывать или заказывать необходимые прикладные программы, затрачивая порой суммы, в несколько раз превышающие стоимость аппаратуры.

Опытный пользователь прежде всего задается вопросом, насколько качественно, развито и доступно работающее на выбранной ЭВМ программное обеспечение. Можно, продав последнюю рубашку, приобрести для школы вычислительный комплекс «Эльбрус», и это будет бесполезная груда неплохой аппаратуры, так как нет подходящих учебных и обучающих программ. А можно выбрать КУВТ на базе БК-0010, технические характеристики которых таковы, что за согласие работать на этой технике надо бы приплачивать. И тем не менее благодаря героическим усилиям членов «Клуба пользователей БК» и отдельных граждан,

создающих все новые и новые программы, пользы от БК в условиях школы будет больше, чем от «Эльбруса».

Наличием разнообразного программного обеспечения объясняется грандиозный успех персональных ЭВМ «Apple-2» на рубеже 70—80-х гг. Фирма не скрывала ни внутреннего устройства, ни операционной системы своей машины, и ежедневно появлялись все новые разработки программистов для «Apple-2».

В настоящее время качество программного обеспечения и распространенность соответствующих ЭВМ настолько взаимозависимы, что невозможно сказать, что важнее. С одной стороны, огромная популярность IBM-совместимых машин объясняется тем, что фирма IBM в специальных журналах регулярно публикует сведения о новых версиях операционной системы, о появлении новых инструментальных средств и приглашает всех желающих создавать новые программы.

С другой стороны, таких желающих становится все больше и больше, так как широкая распространенность стандарта IBM обеспечивает большой рынок сбыта программ. Например, электронные таблицы Lotus стоимостью менее 500 долларов принесли фирме Lotus Development Corporation доход около 1 млд., потому что разошлись в количестве более 2 млн. экземпляров.

Рассмотрим классификацию программного обеспечения (рис. 16).

Системное программное обеспечение включает в себя операционную систему; средства контроля, тестирования и диагностики ЭВМ; «надстройку» над операционной системой, облегчающую пользователю диалог с ЭВМ.

Прикладное программное обеспечение состоит из библиотеки стандартных процедур (тригонометрические, логарифмические и т. д.); пакетов прикладных программ (рассмотрены ниже); систем программирования (средства трансляции, редактирования, загрузки и отладки программ).

Главный инструментарий пользователя — пакеты прикладных программ. Для большинства ЭВМ имеются тысячи различных пользовательских пакетов. Но можно выделить пять основных разновидностей прикладных программ:

1. Программы обслуживания сетей (обеспечивают связь с другими ЭВМ)
2. Электронные таблицы
3. Графические пакеты
4. Системы управления базами данных
5. Системы подготовки текстов

Как правило, современные прикладные программы не только совместимы с определенными ЭВМ, но и *взаимно совместимы*. Например, экономист готовит набор расчетных показателей с помощью электронных таблиц, затем, используя графический редактор, получает те же показатели в виде графиков или диаграмм (не вводя данные повторно), после чего снабжает графическую часть пояснениями с помощью текстового редактора. Нажимая определенную клавишу, он получает готовый материал, отпечатанный на бумаге.

Специальное программное обеспечение предназначено для решения уникальных задач (управление экспериментом, космической аппаратурой, технологическим процессом). Оно разрабатывается под конкретное применение и, как правило, не тиражируется и не продается. Иногда в состав специального программного обеспечения входит уникальная операционная система. Стоимость специального программного обеспечения может в десятки раз превосходить стоимость ЭВМ, для которой оно создано.

После такого общего обзора остановимся подробнее на системе программирования и операционной системе. В состав системы программирования входит три вида трансляторов: ассемблеры, компиляторы и интерпретаторы.

Ассемблер переводит мнемокоды команд в соответствующие коды операций, а имена переменных — в адреса ячеек памяти. Понятно, что это очень простой способ перевода, поэтому ассемблер — это несложная программа, зани-

мающая в памяти мало места.

Компилятор преобразует программу, написанную на языке программирования, в **объектный модуль**, закодированный на машинном языке. Чтобы полученная программа могла работать, необходимо связать ее с другими модулями, преобразовать имеющиеся в ней обращения к стандартным процедурам в конкретные адреса, обеспечить соответствие типов. Поэтому объектный модуль обрабатывается **редактором связей** и превращается в **загрузочный модуль.** Загрузочный модуль можно запускать в работу. Таким образом, компилятор переводит всю программу целиком на язык машин.

Иначе действует интерпретатор. Он прочитывает первую команду алгоритма, переводит ее на язык ЭВМ и тут же дает приказ процессору исполнить эту команду. Затем он делает то же самое со второй командой алгоритма, затем — с третьей и так продолжает до тех пор, пока не дойдет до конца программы. Иными словами, интерпретатор — специалист по синхронному переводу.

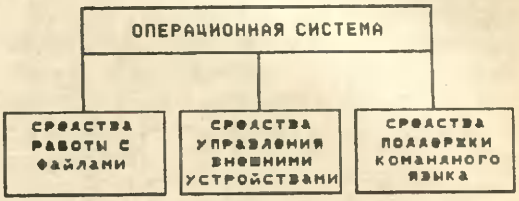
Самая «заметная» для начинающего пользователя часть программного обеспечения — операционная система (ОС). Наибольшее распространение на персональных ЭВМ получили три базовых семейства ОС: CP/M, MS—DOS и UNIX.

Система CP/M (control program for microcomputers) самая простая из названных. **Ядро системы** реализует лишь простейшие команды управления машиной и работы с файлами. Основная нагрузка при работе ЭВМ ложится на **оболочку системы**, т. е. прикладные программы, работающие под управлением ОС и обеспечивающие дружественный диалог с ЭВМ, выполнение различных сервисных операций (разметка дисков, тестирование, работа с сетью и т. д.). Система CP/M хороша для начинающего пользователя, но для опытного программиста часто бывает недостаточно ее возможностей. Для работы этой ОС требуется всего 16K байт памяти, поэтому она используется на дешевых восьмиразрядных персональных ЭВМ.

Более сложна MS — DOS (Micro Soft Disk operating system). Она имеет достаточно развитые сервисные средства, древовидную иерархическую файловую структуру, более мощный командный язык. Система занимает приблизительно 60K байт и, как правило, применяется на шестнадцатиразрядных ЭВМ.

Но наиболее удобны для профессионала системы семейства UNIX. В них сочетаются достоинства MS — DOS и большие дополнительные возможности. Преимущество семейства UNIX объясняется, в частности, наличием развитой библиотеки служебных программ. Система поддерживает многопользовательский режим работы. Командный язык UNIX во многом совпадает с языком СИ, кроме того, почти все модули системы написаны на этом языке. Однако вместе с библиотекой система требует около 5M байт памяти.

Любую операционную систему, независимо от класса, можно условно разделить на стандартные составные части (рис. 17).



Итак, начав с МОП-транзистора, мы с учениками постепенно собрали электронную вычислительную машину. Конечно, специалисты могли бы это сделать и без нас, но не будем жалеть затраченных сил. Зато теперь мы убедились, что в «умных» машинах нет ничего сверхъестественного.

Теперь настало время понять, зачем нам нужна ЭВМ, где ее можно применить. Но это уже совсем другая статья.

М. ГОЛЬЦМАН, А. ДУВАНОВ, Я. ЗАЙДЕЛЬМАН, Ю. ПЕРВИН

Обработка текстов в Роботландии

Редакторам отведено значительное место в курсе раннего обучения информатике.

Во-первых, редакторы непосредственно продолжают информационную линию курса [1]. В самом начале знакомства с информатикой дети говорили об информации, обсуждали примеры различных информационных процессов. Они подготовлены к тому, чтобы подробно поговорить о технологии обработки информации. Редакторы дают для этого богатый материал.

Во-вторых, в редакторах отчетливо выражена одна из основных сущностей информатики — конструирование сложного объекта (текста, рисунка, мелодии) из простых и элементарных составляющих. В этом смысле изучение редакторов может рассматриваться как эффективная пропедевтика фундаментального понятия информатики — процедуры (вспомогательного алгоритма).

В-третьих, тема «Редакторы» имеет глубокую практическую направленность. Каждый из учеников, выйдя из стен школы, встретится с необходимостью обработать текстовый документ или начертить схему. Очень важно, чтобы он умел делать это с помощью основного инструмента информационного общества — компьютера.

Наконец, применение редакторов в учебно-воспитательном процессе при продуманной методике может эффективно помочь формированию творческих способностей учащихся.

В Роботландию включены три редактора: текстовый редактор Микрон, графический редактор Раскрашка и музыкальный редактор Шарманщик. Все они являются программными средствами, адаптированными к условиям начальной общеобразовательной школы. При разработке системы ее авторы исходили из тезиса: адаптированные программные средства, в частности редакторы,— для всеобщего обучения; профессиональные средства — для факультативов, кружков и профориентационной подготовки.

Поскольку редакторы Роботландии принадлежат к одной программной системе, характерной их чертой является единство интерфейса: общие приемы копирования информации, обменов с внешними носителями, рассылки по сети.

Урок III.2. **Тема урока: «Текстовый редактор Микрон»**

План урока:

Редактирование текстов — информационный процесс.

Задачи редактирования текстов — правильность и удобство восприятия. Приемы редактирования.

Правила работы с редактором Микрон, его режимы. Некоторые понятия и приемы машинной подготовки текстов: размер строки, размещение пробелов, переносы, абзац, управление курсором; замена и вставка; удаление символов, слов, конца строки.

Лабораторная работа: **подготовка текста в Микроне.**

Идея программ-редакторов восходит к детским конструкторам, позволяющим собрать любую придуманную конструкцию из набора небольшого числа стандартных элементов с помощью некоторого числа простых инструментов.

Роль инструментов в игрушке-конструкторе велика: они позволяют не только сооружать сложный объект, но и вносить в конструкцию произвольные изменения в любой момент времени. Используя полиграфическую терминологию, такие программы-конструкторы называют *редакторами*.

Дети, безусловно, слышали слова «редактор», «редколлегия», «редакция». Поэтому достаточно рассказать о потребности газетных и книжных издательств в вычислительной технике, чтобы убедить их в актуальности автоматизации редакторской деятельности. Этим целям могут служить § 6.1 и § 6.2 учебника. В частности, полезно зафиксировать в сознании детей присваивание поисковых образов хранимому в архиве (или в памяти компьютера) краткому описанию

(карточке) опубликованной газетной заметки. Этот образ можно использовать на одном из следующих уроков, когда будет идти речь о классификации информации по задаваемым признакам.

Учитель представляет учащимся текстовый редактор Микрон, разослав его по сети, и предлагает вниманию детей лабораторную работу по основному режиму Микрона — набору текстов и исправлению единичных символов. Работа состоит в переписывании фрагмента известного стихотворения С. Я. Маршака «Багаж».

Каждая строка стихотворения вмещается в физическую строку экрана, а длина фрагмента (количество строк) превосходит размер экрана. Это последнее обстоятельство позволяет убедительно продемонстрировать понятие окна (§ 6.4 учебника). Закончив набор текста, дети должны совершить перемещения окна вверх и вниз. С этой целью полезно предложить детям чтение вслух (по очереди) записанного на экране стихотворения. Для повышения активности ученики читают стихотворение по четверостишьям. Остальные в это время следят за ходом чтения. Слежение состоит в перемещении курсора по строкам.

Детей надо попросить принести к следующему уроку любимые книжки, чтобы на следующем уроке они смогли набрать текст в редакторе Микрон.

§ 6.3. Программы-редакторы

Компьютер всегда работает по программе. Так называют задание для компьютера, написанное на понятном ему языке. До сих пор на уроках информатики вы работали с программами-исполнителями. Некоторые из них были очень простыми, как Курсор или Автомат, другие — достаточно сложными, как Машинист или Плюсик. Каждый исполнитель задуман для решения некоторого типа задач.

Среди разнообразных компьютерных программ есть такие, с которыми можно работать, как с обычным детским конструктором.

В отличие от игры-конструктора программа-конструктор работает не с металлическими или пластмассовыми деталями, а с информацией. Такие программы называют *редакторами*.

Редактор, работающий с текстовой информацией, называется *текстовым редактором*.

Графический редактор позволяет собирать рисунок (графическую информацию) из простых частей: квадратов, треугольников и кругов.

Музыкальный редактор собирает из отдельных звуков мелодию песни.

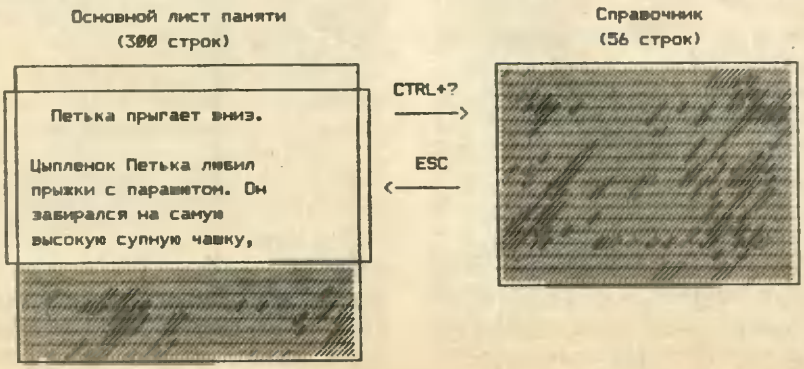
Текстовый редактор Роботландии, с помощью которого можно писать диктанты и сочинения, исправлять ошибки в тексте и красиво оформлять написанные рассказы, называется *Микрон*.

Вопросы и упражнения:

1. Почему программы-редакторы похожи на конструкторы?
2. Какие бывают редакторы?

§ 6.4. Справочник Микрона

О том, как работает Микрон, рассказано на страницах этой книги. Но если вы что-то забудете, можно узнать об этом у самого Микрона. Для этого надо вызвать на экран подска ку (кла-



виши *CTRL*+?). На экране появляется таблица. В ней перечислены операции редактора и клавиши для их выполнения.

Таблица-справочник — большая, она не умещается целиком на экране. Однако, нажимая на стрелки ВНИЗ и ВВЕРХ, можно посмотреть любую ее часть. В этом случае экран монитора подобен окну, через которое видна часть длинного текста справочника.

Выход из справочника — нажатие на клавишу *ESC*.

Вопросы и упражнения:

1. Как перейти к просмотру справочника Микрона?
2. Как посмотреть части справочника, не видимые на экране?
3. Как выйти из справочника?

Урок III.3. **Тема урока: «Исправление ошибок в текстовом редакторе; классификаторы»**

План урока:

Приемы корректировки строк — удаление и вставка, разрезание и объединение строк.

Лабораторная работа: **операции над строками; корректировка стихотворений средствами текстового редактора.**

Задачи на классификацию объектов Выделение существенных признаков объекта.

Игра-классификатор на определение лишнего объекта в множестве.

Лабораторная работа: **игра «Кто лишний?».**

Домашнее задание: **придумать игру-классификатор.**

Центральная тема урока — операции над строками. Приступая к ней, учитель опирается на сформированные умения школьников корректировать ошибки в отдельных символах.

С клавишей верхнего регистра (*SHIFT*) уже приходилось встречаться. Она нужна для «усиления» действия буквенных клавиш.

Сегодня демонстрируются еще две «усиливающие» возможности *SHIFT* при взаимодействии с клавишами удаления (*DEL*) и вставки (*INS*)*. § 6.9 учебника содержит подробное описание операций удаления и вставки строк, а также примеры их использования.

На этом же уроке вводятся две новые операции Микрона: разделение строк и объединение («склеивание») строк.

Теперь можно начать лабораторную работу. Ученики получают на свои экраны тексты, подлежащие корректировке, и задания. Тексты хорошо известны детям — это стихи А. Л. Барто. Тем не менее учитель должен быть готов напомнить учащимся забытые строки стихов (но не правила корректировки ошибок!).

Вот задания лабораторной работы:

1. Удалите лишние слова:

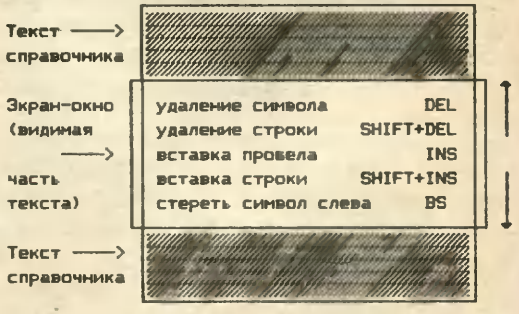
Мишка
Уронили Мишку на пол,
Оторвали Мишке правую лапу.
Все равно его не брошу,
Потому что он очень хороший.

Аналогично первому строятся последующие задания на удаление лишних частей строк, удаление лишних строк и вставку недостающих строк.

§ 6.5. Как устроен Микрон

У Микрона два листа памяти. Один — основной, на нем человек может набирать тексты, другой — всегда хранит текст справочника. Человек рассматривает листы Микрона через окно — экран монитора.

Используя клавиши-стрелки, можно перемещать окно вверх или вниз по листу памяти. В окне помещаются 24 строки текста, по 32 символа в каждой.



При наборе текстов человек, конечно же, совершает ошибки. (Вспомните, какие три вида ошибок совершаются при вводе текста с клавиатуры.)

* Более подробное взаимодействие *SHIFT* с другими клавишами описано в [2].

Все исправления ошибок — замена, удаление и вставка отдельных символов — выполняются по обычным правилам, с помощью клавиш *BS*, *DEL* и *INS*. Впрочем, в Микроне есть и новые возможности для правки текста. О них будет подробно рассказано на других уроках.

Вопросы и упражнения:

1. Можно ли в Микроне набрать текст длиннее 24 строк?
2. Как посмотреть текст, не видимый на экране монитора?
3. Сколько символов, которые можно рассмотреть через окно экрана без его перемещения?

Урок III.4. **Тема урока: «Копирование»**

План урока:

Приемы копирования символов и слов.
Копирование строк.
Лабораторная работа: копирование слов и строк («Два веселых гуся»).
Проверка домашнего задания: игра «Кто лишний?» у доски по сюжетам домашнего задания.

Лабораторная работа: **классификатор «Кто лишний?».**

Домашнее задание: **написать два примера для упражнений на копирование символов и копирование строк.**

Первой из новых возможностей текстового редактора рассматривается копирование символов.

Обсуждается задание: написать несколько строф стихотворения С. Я. Маршака «Багаж».

Вопрос о технологии написания текстов, содержащих повторяющиеся фрагменты, рекомендуется вынести на обсуждение. Дети, безусловно, предложат копировать слова «диван», «чемодан». Возможно, они обратят внимание на то, что предлагаемые для копирования слова не совпадают по падежам с требуемым словом и потому отличаются окончаниями. Однако и в этом случае копирование дает эффект: можно либо целиком скопировать слово и затем удалить (или заменить) последнюю букву, либо скопировать все символы, кроме последнего, а затем дописать и его.

Когда прием копирования символов и слов усвоен школьниками, можно переходить к копированию строк. Иллюстрировать это объяснение удобно тем же стихотворением. Знакомый пример упрощает изложение и первое восприятие нового дидактического материала. Что еще более важно, он позволяет продемонстрировать высокую эффективность копирования целых строк по сравнению с посимвольным.

Завершив обсуждение приемов копирования, дети подходят к лабораторной работе на данную тему. Им предстоит набрать слова известной песенки про двух веселых гусей — серого и белого. Задание и текст песни к началу работы положены около каждой ученической машины. Здесь важно догадаться, что многократно повторяющаяся строка «один серый, другой белый» надо не переписывать, а копировать.

(Этот урок, так же как и все остальные, поддержаны, разумеется, текстами учебника. В настоящей публикации опущены параграфы ученического учебника, относящегося к этому и нескольким следующим урокам. Соответственно сделаны некоторые сокращения в методических комментариях.)

Урок III.5. **Тема урока: «Выравнивание; классификация»**

План урока:

Приемы форматирования текста.
Центрирование.

Лабораторная работа: **форматирование («Снегурочка и лиса»).**

Проверка домашнего задания: игра «Кто лишний?» у доски.

Классификация объектов в статичных множествах.

Лабораторная работа: **классификатор Полянка.**

Домашнее задание: **придумать игру в классификацию.**

Для закрепления пройденного материала детям можно предложить задание: просклонять слово «дом» по падежам.

Предупреждение: каждая новая строка не переписывается, а копируется.

После этого упражнения начинается изучение новой темы, следующего приема обработки текстов — выравнивания. В профессиональной терминологии информатики этот прием называют форма-

тированием. Введение такого сложного термина для учащихся III класса нельзя считать оправданным, тем более что в названии «выравнивание» отражается вся сущность этого приема. Актуальность выравнивания диктуется не только эстетическими соображениями, но и упоминавшимися ранее (§ 6.1) требованиями оформления газетного и книжного материала.

Текст для лабораторной работы на выравнивание записан на диске — это небольшой (в три абзаца) фрагмент сказки.

Перед нажатием на «клавишу выравнивания» (*F5*) важно, чтобы дети увидели неровный правый край текста. Только после этого им можно разрешить самостоятельно выполнить выравнивание по абзацам. В тексте сказки это придется делать трижды. После первого нажатия надо остановиться и сравнить вид первого, выровненного абзаца и второго, имеющего «рваные» края.

Теперь предстоит выравнивание заголовка и подзаголовка. Здесь тоже уместно сделать ссылку на профессиональную терминологию: установка слова в центр строки называется центрированием. Учитель демонстрирует прием центрирования; ученики повторяют его действия на своих машинах.

Урок III.6. **Тема урока: «Справочник редактора; откатка»**

План урока:

Проверка домашнего задания: игра в классификацию.

Актуальность самодокументации программных средств.

Описание справочника Микрона и правила пользования справочником. Откатка.

Классификация объектов в динамическом множестве, игра.

Домашнее задание: **придумать игру в классификацию.**

В большинстве сложных программных систем существует режим подсказки, позволяющий использовать машину как справочник по этой системе. Умение обратиться к компьютеру за подсказкой относится к числу важнейших умений человека в современном информационном обществе.

Прислав на ученические машины редактор Микрон, учитель демонстрирует способ входа в режим подсказки (§ 6.4): он един для всех программ Роботландии.

Один из новых приемов редактирования — восстановление ранее сделанных исправлений. Его часто называют откаткой. Новая операция требует экспериментирования с текстом на экране ученического рабочего места. Ученикам предлагается удалить три нужные строки, а затем восстановить эти строки троекратной откаткой.

Учитель должен использовать откатку как демонстрацию дружественного характера отношений машины с человеком: школьник может не опасаться собственных ошибок. Откатка позволяет вернуться к любому из предшествующих состояний обрабатываемого текста.

Урок III.7. **Тема урока: «Пересылка по сети, сохранение текстов»**

План урока:

Проверка домашнего задания: игра в классификацию.

Актуальность сохранения и передачи информации по сети. Режимы работы с сетью.

Импровизации учеников на заданную тему.

Сбор текстов сочинений на учительскую машину.

Демонстрация учащимся сохраненных текстов.

Лабораторная работа: **обмен текстами.**

Домашнее задание: **описать порядок подготовки текста, собираемого из отдельно написанных частей.**

Основная тема урока — пересылка информации по сети. В § 6.10 дается определение (точнее, описание) сети вычислительных машин. Компьютеры нашего кабинета соединены в сеть, которую называют *локальной* (местной). С помощью специальных устройств можно соединять не только по соседству расположенные машины, но и компьютеры, удаленные друг от друга на многие десятки, сотни и тысячи километров. Тогда образуется большая информационная сеть. Если компьютер подключен к такой сети, то, не отходя от экрана, можно послать сообщение в сибирский Красноярск или даже в американский Сан-Франциско. В недалеком

будущем наша планета будет вся покрыта информационной сетью вычислительных машин. Поэтому так важно нынешнему школьнику научиться работать в такой сети — уметь передавать и принимать информацию с помощью компьютеров.

Демонстрацию возможностей локальной сети класса лучше начать с приема информации, получаемой с центральной машины. Тем более, что фактически режим приема информации уже использовался на уроках.

Затем детям надо освоить обратную операцию — передачу текста со своей машины в сеть, для чего им нужно создать на своих экранах оригинальные тексты. Учитель рекомендует написать на экранах анкеты типа:

```
Фамилия, имя:
      Петя Кук
Дата рождения:
      11 марта 1980 года
```

Написав анкету, ученик передает ее по сути учителю. Выполнив прием очередного текста, учитель приглашает к своему экрану ученика, пославшего текст, чтобы тот убедился в получении анкеты учителем.

Перед тем как начать заполнение анкет, учитель объясняет, как важно сохранять текст на внешних носителях и почему необходимо именовать текст, сохраняемый на диске.

Когда все ученические тексты переданы на центральную машину, надо показать ученикам важный уникальный элемент сети — канал связи учительской машины с принтером.

Сначала учитель показывает ученикам каталог диска. Рассматривая его, они должны узнать имена своих текстов.

Затем учитель поочередно читает ученические тексты: войдя в режим «Чтение текста» и назвав имя нужного, он показывает его на экране. Потом учитель печатает анкету ученика на бумаге.

§ 6.10. Почта в классе

Третий «А» выполнял упражнения на Микроне. Вдруг Петя Кук поднял руку. Глаза его восторженно сияли.

— Сан Саныч! — обратился он к учителю.— Я сейчас сочинил стихи. Про компьютер. Можно, я их прочитаю?

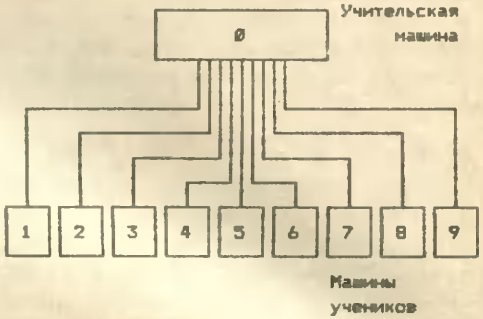
— Пришли их, пожалуйста, мне.

— Прислать? — удивился Петя. — Разве у нас в классе есть почтовый ящик?

И он сам рассмеялся своей шутке.

Но учитель спокойно сказал:

— Конечно, есть. Наша почта — это соединение всех компьютеров класса. Такое соединение машин называют *сетью*. В нашей сети можно послать сообщение с центральной (учительской) машины на все другие машины класса и наоборот — с ученической можно написать «письмо» центральной машине.



— А как послать стихи с моей машины?

— В редакторе Микрон есть режим работы *«обмен информацией»*. Войти в него можно, нажав клавишу *ESC*.

Петя нажал *ESC* и увидел на экране меню из трех «блюд»:

— «Прием текста» — это ожидание «письма» с машины учителя, а «Передача текста» — это как раз то, что тебе сейчас нужно — посылка на центральную машину,— объяснил Сан Саныч.— Выбор в меню нужной операции делается, как обычно, курсором. Чтобы выполнить операцию, надо нажать клавишу выполнения.

```
  Прием текста
■ Передача текста
  Диктант
```

На центральной машине сейчас тоже работает Микрон в режиме обмена информацией. Я выбираю операцию «Прием текста», а ты, Петя, ставишь курсор на строчку «Передача текста».

И когда мы оба нажмем у себя клавиши выполнения (сначала я, моя машина должна быть настроена на прием), то твой текст перешлется по сети в память центральной машины и я увижу его на экране.

Петя отправил свой стихотворный текст по классной «почте». На экране учительского компьютера появились строки:

Зажег зеленый свой экран
Передо мною монитор,
И строчек ровный караван
Погнал вперед курсор.
Даже ошибки не беда,
Когда со мной Микрон.
Легко и просто без следа
Их исправляет он.

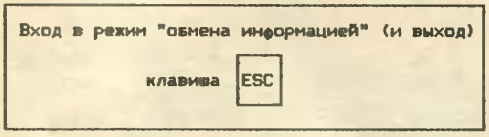
— Молодец, Петя! Хорошие стихи. Ребята! Вы хотите почитать стихи, которые Кук сочинил про редактор Микрон?

— Да! Конечно! Хотим! — раздались голоса.

— Тогда войдите, пожалуйста, все в режим «обмен информацией» и выберите в меню операцию «Прием текста». Готово? Тогда я вхожу в режим «Передача текста».

И центральная машина очень быстро разослала стихи всем ученикам.

В заключение Сан Саныч сказал, что покинуть режим обмена информацией можно, повторно нажав клавишу *ESC*.



Вопросы:

1. Какие действия должны выполнить ученик и учитель, чтобы передать учителю текст с экрана ученика?

2. Какие действия надо выполнить ученикам и учителю, чтобы получить учительский текст на экранах ученических компьютеров?

3. Как с помощью Микрона один ученик может послать сообщение другому?

Урок III.8. **Тема урока: «Машинный диктант»**

План урока:

Проверка домашнего задания.

Лабораторная работа: **классификатор Полянка.**

Режим «Диктант» в текстовом редакторе.

Сообщения редактора об ошибках.

Лабораторная работа: **диктант и последующая работа над ошибками.**

Подготовка учителя к уроку проводится в два этапа: до урока и в начале его*. Диктант должен соответствовать программе по объему и уровню сложности.

Хотя о стилистике уже говорилось, нелишне еще раз напомнить:

переносы делать не надо;

пробелы перед знаками препинания недопустимы;

пробелы после знаков препинания обязательны.

По поводу процесса диктовки есть одна методическая рекомендация: заголовок диктанта надо сообщить детям в самом конце и попросить их вставить заголовок в начало рассказа, поместив его (заголовок) точно в середине строки. Тем самым ребята заодно вспомнят, как можно вернуться к началу текста, как вставить строку, как центрировать заголовок.

Когда диктовка закончена, учитель предлагает детям выровнять текст и проверить его. Сначала проверка выполняется «вручную» — путем перечитывания текста; затем учитель говорит о возможности автоматической проверки, которая включается нажатием клавиши *SELECT*. Если в диктанте есть ошибки, то на экране зажигается надпись ОШИБКА, а курсор указывает ее место.

Ученик имеет полную возможность поразмышлять об ошибке и ее причине. Не каждая ошибка понятна ученикам. Поэтому многим из них надо будет помочь понять ошибку и проконтролировать их умение ее исправлять.

Исправив ошибку, ученик вновь нажимает *SELECT*. Машина зажигает метку ОШИБКА в новом месте. Так продолжается до тех пор, пока не будут исправлены все ошибки. Тогда в конце текста загорается приятное сообще-

* Данный вопрос подробно рассматривался в [2].

ние ОШИБОК НЕТ. Формальная цель урока — получить это сообщение на всех экранах.

§ 6.11. Машинный диктант

Как волнуются школьники, когда на уроке русского языка учитель объявляет: «Диктант!». Надо постараться написать диктант аккуратно, без помарок, а главное — не сделать ошибок.

Впрочем, и учителю предстоит немало забот с диктантом. Ведь придется проверить столько тетрадей, исписанных разными почерками!

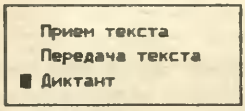
Микрон приходит на помощь и ученикам, и учителю. Что ему стоит проверить диктант, сравнив текст, написанный учеником, с правильным текстом, полученным от учителя!

Вот порядок написания диктанта в Микроне:

1. Получить от учителя правильный текст диктанта.
2. Написать диктант на экране под диктовку учителя.
3. Исправить ошибки.

1. Как получить от учителя правильный текст диктанта?

Можно принять текст диктанта в режиме обмена информацией, выбрав операцию «Прием текста». Но тогда текст появится на экране, и никакого диктанта не получится. Меню обмена информацией содержит специальную операцию «Диктант». Это прием текста в специальную память, содержимое которой нельзя увидеть на экране монитора.



2. Как написать диктант на экране под диктовку учителя?

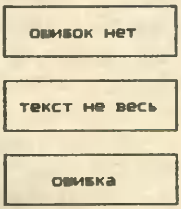
Это делается как обычно, но вместо тетради у вас экран монитора, а вместо ручки — клавиатура. Если вы привыкли к клавиатуре, то будете писать диктант на экране почти с той же скоростью, что и на бумаге. Только машинный диктант пишется красивыми ровными буквами, без всяких помарок. Исправление ошибок не ухудшит внешний вид текста — ведь после замен, вставок и удалений на экране не остается следов.

3. Как найти и исправить ошибки?

Прежде всего их надо найти. Очень хорошо, если вы это сделаете сами. Но если вы затрудняетесь, Микрон поможет вам.

Написав диктант, нажмите клавишу *SELECT*. Эта клавиша отдает Микрону команду сравнить написанный текст с правильным, полученным с центральной машины.

После проверки Микрон выдаст одно из трех сообщений:



Если ошибок нет, вы большой молодец!

Сообщение «текст не весь» означает, что диктант не дописан до конца, но в написанной части ошибок нет.

Сообщение «ошибка» выдается Микроном при обнаружении первой же ошибки, дальше текст не проверяется. Нажмите любую клавишу, и вы увидите курсор. Он указывает место, где Микрон обнаружил расхождение вашего текста с текстом учителя. Теперь нужно подумать, в чем заключается ошибка, исправить ее и для проверки снова нажать клавишу *SELECT*.

Некоторые правила написания текстов

Все основные знаки препинания (точка, запятая, восклицательный и вопросительный знаки) пишутся слитно с предшествующим словом, а после них обязательно оставляется промежуток (один или несколько пробелов).

Это правило очень важно при проверке диктантов на Микроне. Возможно, учитель и не рассердится, если прочитает в вашем тексте

диван, чемодан, саквояж,

но Микрон обязательно «придерется» и сообщит об ошибках, допущенных в конце слов «диван», «чемодан» и т. д.

Вопросы:

1. Почему правильный текст диктанта не передается ученикам при помощи

операций «Прием текста» и «Передача текста»?

2. Может ли случиться, что Микрон выдал сообщение «ошибка», а в вашем диктанте ошибок нет?

3. Почему Микрон «придирается» при проверке диктанта, когда перед запятой обнаруживает пробел?

Урок III.9. **Тема урока: «Применение текстового редактора на уроках русского языка; классификация информации»**

План урока:

Формализованные задачи обработки текстов на уроках родного и иностранного языка.

Технология подготовки к упражнению по русскому языку.

Лабораторная работа: **упражнение на вставку пропущенных букв.**

Классификация динамического потока объектов.

Лабораторная работа: **классификация объектов с помощью классификатора Парашютик.**

Домашнее задание: **подготовить сочинение-роботлашку.**

На сегодняшнем уроке будет получен не только прямой результат урока — закрепление навыков работы с текстовым редактором, но и важный косвенный эффект — расширение кругозора учащихся в прикладных применениях школьного компьютера. Сюжетную линию урока составляет упражнение по русскому языку (и из учебника русского языка!). В этой лабораторной работе-упражнении надо вставить пропущенные буквы*.

Для передачи учебного текста ученику надо войти в режим «Прием текста», а учителю — в режим «Передача текста». На учительском экране в это время должен находиться учебный текст. На уроке можно использовать тексты, созданные учителем, или записанные на диске Роботландии стандартные тексты.

Это упражнение школьники выполняют быстрее, чем диктант: здесь не приходится набирать весь текст, нужно только вставлять пропущенные буквы, да перемещать курсор от одной буквы к другой.

* Технология подготовки и проведения урока на данную тему подробно разобрана в [2].

§ 6.17. На уроке русского языка

Ребята привыкли видеть компьютер на уроках информатики. И когда Петя Кук узнал, что на урок русского языка надо прийти в кабинет информатики, он весьма удивился.

Как только на экране монитора появилось изображение, Петя сразу узнал:

— Да ведь это же Микрон!

— Правильно, — сказал учитель. — Сегодня на уроке можно обойтись без тетрадей. Вы будете делать упражнения на экранах.

Петя прочитал сначала задание:

```
Упражнение 266. Прочитайте текст. Вставьте
пропущенные буквы.
```

а потом текст упражнения:

```
Целый день Митя катался на коньках с л_дяной
горы и по реке. Привезал он домой и г_ворит
отцу: - У нас вес_ло зимой! Я бы хотел, чтобы
всегда была зима! - Зап_ши твое желание в мою
карманную книжку, - ск_зал отец. Митя зап_сал.
```

Кук выполнил упражнение очень быстро: он подводил курсор поочередно к каждой пропущенной букве и нажимал правильную (как ему казалось) букву.

Каково же было его огорчение, когда, закончив упражнение, он нажал клавишу *SELECT*! В слове «*весело*» ярко горело: ОШИБКА. Действительно, у Пети было написано «*весило*».

— Ой! — воскликнул Петя и принялся размышлять над тем, как исправить ошибку.

— Сегодняшнее упражнение,— сказал Петя учителю в конце урока,— очень похоже на машинный диктант. Это все благодаря Микрону. С ним можно часто проводить уроки русского языка в кабинете информатики.

Вопросы:

1. Чем отличается компьютерное выполнение упражнений по русскому языку от машинного диктанта?

2. Подумайте и догадайтесь, как можно организовать работу, описанную в параграфе. Какие операции по пересылке должны быть предварительно выполнены учителем и учеником?

3. Может ли при выполнении упраж-

нений на компьютере появиться сообщение ТЕКСТ НЕ ВЕСЬ?

4. Можно ли использовать Микрон на других школьных предметах? Как?

Урок III.10. **Тема урока: «Сочинение Роботлашек»**

План урока:

Самостоятельная работа учащихся с редактором Микрон над сочинениями Роботлашек.

Сохранение сочинений на диске.

Основное содержание работы учеников на этом уроке — набор на клавиатуре текстов написанных дома сочинений-роботлашек. Если в классе окажутся ученики, не выполнившие домашнего задания, им придется импровизировать.

Основная форма работы учителя — индивидуальные беседы с учениками около экранов. Учитель должен помочь школьнику оформить его замысел, сгладить грубые стилистические шероховатости, отметить и объяснить ошибки, напомнить требования формы.

К этим требованиям относятся наличие заголовка в начале текста и авторской подписи в конце. Набранный текст надо будет выровнять. Заголовок надо центрировать, а подпись — сместить к правому краю экрана.

В конце урока учитель записывает все ученические сочинения на диск. Не беда, если некоторые из сочинений не будут завершены до звонка. Тогда на диск будут записаны только завершенные части текстов. Закончить сочинения можно будет на следующем уроке. Эту возможность следует декларировать в самом начале урока, чтобы дети не испытывали давления времени.

Урок III.11. **Тема урока: «Издание Роботлашек»**

План урока:

Обсуждение технологии выпуска книги.

Распределение работы по подготовке издания.

Печать книги.

Распределение авторских оттисков. Изготовление книги. Обсуждение выпуска.

Заключительное занятие по Микрону посвящено выпуску книжек-роботлашек. Ученики сочиняют рассказы, сказки, стихи, вводят их в компьютер с помощью Микрона, печатают и собирают в книжку, используя специальную программу.

Хотя систематическое изложение Микрона завершается, школьникам еще предстоят интересные встречи с текстовым редактором. Так, после нескольких уроков следующей темы (графический редактор Раскрашка) дети всем классом выпускают стенгазету: свои статьи, свои рисунки!

На одном из уроков учитель рассказывает детям о визитных карточках, и дети с удовольствием изготовляют свои карточки на компьютере (§ 6.18).

А за пределами уроков учитель сопровождает свой класс на экскурсию в редакцию районной газеты или заводской многотиражки. Хотя экскурсию ведет сотрудник редакции, однако она подготовлена учителем, который предварительно встретился с редактором, объяснил цель предстоящего визита школьников, определил объем и содержание демонстрируемого материала.

§ 6.18. Визитная карточка

Прозвенел звонок, и в класс вошел учитель вместе с незнакомым мальчиком.

— Ребята! Это ваш новый товарищ. Он приехал из другого города и теперь будет учиться у нас. На перемене вы с ним познакомитесь. А сейчас начнем урок.

Учитель посоветовал новенькому сесть рядом с Петей Куком: соседнее место сегодня пустовало.

Кук не мог дождаться перемены и сразу же принялся нашептывать новичку вопросы:

— Как твоя фамилия? А как тебя зовут? Где ты живешь?

Новенький, ничего не говоря, вытащил из кармана куртки маленькую карточку и передал ее Куку. На карточке Петя прочитал:

СОЛОВЬЕВ
Василий
ученик 3 класса "а" школы №6
152140
Переславль-Залесский
ул. Лесная, 5, кв. 12 телефон 2-88-65

— Вот это да! — изумился Петя.— Что это?

— Это визитная карточка,— ответил Вася и спросил: — Удобно, правда?

— Еще бы! Не надо даже записывать адрес в записную книжку. К тому же у меня почерк плохой. А тут очень красиво напечатано. Как ты это сделал?

— Очень просто. На компьютере. С помощью Микрона.

— Значит, можно отпечатать сколько угодно таких карточек и раздать их всем своим знакомым?

— Конечно,— подтвердил Вася и добавил: — А особенно незнакомым. Ведь визитные карточки очень полезны при знакомствах. Поэтому ими часто пользуются взрослые.

— Решено. На ближайшем же уроке информатики я попрошу у учителя разрешения сделать себе визитные карточки.

Литература

1. *Дуванов А., Зайдельман Я., Первин Ю., Гольцман М.* Роботландия — курс информатики для младших школьников // Информатика и образование. 1989. № 5.

2. *Дуванов А.* «Микрон» // Информатика и образование. 1989. № 4.

В. БУЦИК

32 Обучение младших школьников началам информатики

Современные тенденции в обучении информатике все более отражают ее как науку, имеющую собственные методы (формализация, информационное моделирование, алгоритмизация...), средства (компьютеры, компьютерные системы, программное обеспечение...) и имеющую широкий спектр применения (прежде всего в автоматизации интеллектуальной деятельности людей) [1, 2].

Поэтому, на наш взгляд, целесообразно формировать у младших школьников не только элементы компьютерной грамотности [4], но и начальные знания основ информатики, осуществлять пропедевтику ее фундаментальных понятий и способов деятельности [3].

В настоящее время нами разрабатывается методическая система, ориентированная на обучение младших школьников началам информатики, в которой формулируются цели обучения, определяется содержание разрабатываются программные средства и методика их использования. В данной статье мы предлагаем эту систему для обсуждения.

Цели обучения

1. Формирование представлений о возможностях компьютера в области обработки информации (компьютер рисует, вычисляет, пишет...), хранения информации (компьютер запоминает рисунки, числа, тексты...) и передачи информации (от человека к компьютеру, от компьютера к компьютеру, от компьютера к человеку).

2. Знакомство с основными понятиями и их отношениями как общими: информация, язык, знания, модель, алгоритм, структура и др. так и частными: данные, таблицы данных, объекты, имена, значения, условия, структуры записей процедурных ответов.

3. Формирование навыков работы с клавиатурой (набор текста, набор чисел, управление движением и фиксирование объектов на экране дисплея, выбор режимов работы по меню).

4. Освоение способов деятельности, отражающих специфические методы информатики: формализацию (описание условий и решений задач с ориентацией на их компьютерное исполнение), алгоритмизацию (запись для некоторого типа задач процедурных ответов простейшей структуры типа «ветвление», «выбор», «цикл»), решение практических задач с применением компьютера (ведение компьютерных дневников при наблюдениях за погодой, жизнью растений и животных), решение задач пред-

лагаемых компьютером, рисование и конструирование на компьютере.

5. Формирование культуры работы в компьютерной инфосфере при активном развитии личности и мышления школьников.

Содержание обучения (*I—IV* класс)

I класс. Учащиеся усваивают, что рассматриваемый объект может иметь имя или за ним можно закрепить имя. Каждому из объектов они дают имя, по имени отыскивают соответствующий ему объект, указывают объекты в задаче, присваивают им имена, описывают решения через имена, решают логические задачи с процедурными ответами простейшей структуры.

Учащиеся приобретают навык создания несложных рисунков на простейших графических редакторах, конструируют несложные объекты из более простых, присваивают имена выполненным изображениям. Работая с таблицами чисел или таблицами имен, отыскивают элементы с заданными значениями, указывают их номера в таблицах, определяют размеры таблицы.

II класс. Учащиеся усваивают, что каждый рассматриваемый объект может иметь несколько имен, а каждому имени может соответствовать несколько значений. Для каждого предлагаемого объекта они придумывают имя, подбирают или строят для него значение (модель объекта), выделяют в каждой задаче, что «дано» и что «надо» получить (рассчитать, вычислить), переводят текстовые задачи на язык краткой записи компьютерного решения, решают задачи на компьютере, решают задачи с процедурными ответами и «проигрывают» их на компьютере.

Учащиеся приобретают навык создания сложных рисунков на простых графических редакторах, конструируют сложные объекты из более простых, присваивают имена выполненным изображениям. Работая с таблицами чисел или таблицами имен, отыскивают элементы с заданными значениями, указывают их номера в таблицах, ведут их подсчет.

III класс. Учащиеся усваивают, что каждый рассматриваемый объект может быть по-разному описан (словесно, рисунком...), в жизненных ситуациях выбор имен связывается со смысловым значением слов.

Учащиеся самостоятельно выбирают объекты, дают им имена, создают их компьютерные модели (изображения). Выделяют в каждой задаче, что «дано» и что «надо» получить (рассчитать, вычислить), переводят текстовые задачи на язык краткой записи компьютерного решения, решают задачи на компьютере, решают задачи со сложными процедурными ответами и «проигрывают» их на компьютере.

Учащиеся приобретают навык создания сложных (композиционных) рисунков на простых графических редакторах, конструируют сложные объекты из более простых, присваивают имена выполненным изображениям. Работая с таблицами чисел или таблицами имен, производят сортировку из элементов по указанному признаку.

IV класс. Учащиеся знакомятся с основными базовыми структурами алгоритмов, назначением основных устройств компьютера. Производят вычисления по схемам, блок-схемам и алгоритмам с русской нотацией, решают текстовые задачи на компьютере, записывают процедурные ответы для логических задач в форме алгоритмической нотации. Работая с таблицами чисел или таблицами имен, решают достаточно сложные задачи сортировки, приобретают навык работы с графическими и текстовыми редакторами, простейшими системами формирования баз данных.

Компьютерные программные средства и методика их использования

Программа «Азбучный микромир»

Программа имеет встроенный графический редактор-конструктор. Поле экрана разделено на 4 части: для записи названия демонстрируемого объекта (например, «ДОМ»), для записи его имени (например, «Д»), для создания его изображения и для выбора режима работы.

Программа позволяет выполнить ряд упражнений. Например, по указанному значению устанавливать имя объекта,

для указанного имени отыскивать или создавать его значение и т. п.

Приобретение учащимися навыков практической деятельности, связанной с оперированием понятиями «объект», «имя» и «значение», имеет очень важный мировоззренческий смысл.

Во-первых, в жизни люди всегда имеют дело с объектами (естественными предметами труда, животными, другими людьми...). Каждый рассматриваемый объект обычно получает имя (название, кличку...). Это позволяет один объект отличать от другого. По имени можно отыскать нужный объект или даже несколько объектов, ему соответствующих. Имя может иметь несколько значений.

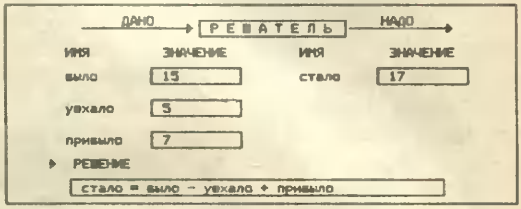
Во-вторых, эти понятия находят все большее отражение в информатике. Являясь атрибутами некоторого языка, они выражают форму представления и передачи информации. Объект — источник информации, имя — носитель информации, значение — смысловое содержание (информация). Это широко используется в современных языках программирования, где в качестве объекта могут выступать реальные величины, в качестве имен — идентификаторы (имена величин), а в качестве значений — числа, тексты и т. д.

И в-третьих, в современном обучении в начальной школе существуют объективные предпосылки овладения учащимися данными понятиями. С I класса, усваивая счет и отношения между числами, школьники овладевают содержательной деятельностью, связанной с этими понятиями. Например, на первых этапах обучения для них в качестве объекта могут выступать реальные предметы или их рисунки, в качестве имен — их названия. В дальнейшем, по мере обогащения учебной деятельности (оперирование с числами, геометрическими фигурами, знаковыми обозначениями), усложняется и процесс информационной обработки: объекты могут иметь лишь мысленное представление, в качестве имен могут выступать различные знаковые и звуковые сочетания, а значения имен могут быть достаточно вариативными.

Целенаправленное формирование понятий «объект», «имя», «значение» через систематическое включение их в учебную деятельность младших школьников, может оказать существенное влияние и на математическое развитие учащихся, усвоение таких математических понятий, как «переменная» и «значение переменной».

Программа «Решатель»

Она представляет простейшую модель электронной таблицы. Имеет поле для записи имен и значений исходных данных и результатов, поле для записи решения задачи в формульном виде. Позволяет по одной и той же формуле производить многократные вычисления с измененными значениями исходных данных. Общий вид поля экрана:



Включение младших школьников в деятельность, связанную с целесообразным использованием компьютеров при решении учебных и практических задач, является важным фактором в формировании основ информационно-компьютерной грамотности: постижению возможностей компьютерной техники, особенностей автоматизации необходимых вычислений, особенностей компьютерного решения задач.

Использование программы «РЕШАТЕЛЬ» предполагает некоторые изменения в методике обучения решению задач. В обычных условиях, как известно, главная цель для учащегося при решении задачи (хотим мы этого или нет) — достичь результата, выраженного числом. Для его получения составляется арифметическое выражение или

по действиям производятся соответствующие расчеты. При этом, как показывает практика, у учащихся порой устанавливается неустойчивая связь: реальные величины задачи — им соответствующие числа. Школьники, например, могут при переходе к оперированию с числами складывать и вычитать разнозначные величины (стоимость и товар, скорость и путь и т. д.), делить и умножать числа так, что в результате получаются трудно объяснимые или вообще необъяснимые величины и отношения.

Поэтому рекомендуется при решении задач не торопиться с переходом от именованных величин к их числовым характеристикам, а более тщательно отрабатывать соотношения различных величин. Как показывает практика, использование программы «РЕШАТЕЛЬ» во многом способствует этому: снимает эмоциональную нагрузку, связанную с ожидаемыми трудностями в вычислениях, позволяет более глубоко вникать в содержательный смысл решаемых задач.

Вместе с тем использование программы «РЕШАТЕЛЬ» позволяет «погружать» учащихся в операционную обстановку работы с компьютером, в которой отрабатываются как операционные навыки, связанные с работой на компьютере, так и навыки его практического использования.

Оформление компьютерного решения задач отличается от обычного тем, что предполагает выбор имен для всех используемых величин и их использование при записи формул. Последовательность анализа задачи при ее компьютерном решении можно представить следующим образом.

1. Выяснить, что дано (известно) в задаче.

2. Дать каждому известному значению имя.

3. Выяснить, что надо найти (неизвестно) в задаче.

4. Установить отношения между всеми величинами в задаче и сделать краткую запись.

5. Записать формулы (у) для решения задачи.

Рассмотрим пример.

1. Формулировка задачи (Математика 1).

У Пети 7 значков, а у Вовы на 3 значка больше. Сколько значков у Вовы?

2. Анализ.

В задаче дано (известно): у Пети 7 значков, у Вовы на 3 значка больше.

В задаче надо найти (неизвестно): сколько значков у Вовы.

Дадим имена известным и неизвестным значениям:

Значки Пети, Больше значков, Значки Вовы.

3. Краткие записи.

Обычная

Значки Пети — 7]
Значки Вовы — ?] больше на 3

Для «РЕШАТЕЛЯ»

Дано	Надо
Значки Пети 7	Значки Вовы?
Больше значков 3	

Пример объяснения краткой записи для «РЕШАТЕЛЯ»:

пишу Дано;

ниже пишу имя «Значки Пети», правее — число 7;

ниже пишу имя «Больше значков», правее — число 3;

правее, на строке Дано, пишу Надо;

ниже пишу имя «Значки Вовы», правее — знак?

4. Записи решения.

Обычная	Для «РЕШАТЕЛЯ»
7+3=10	Значки Вовы=7+3
	или Значки Вовы=Значки Пети+Больше значков

Ответ. У Вовы 10 значков.

Как известно, особый интерес для компьютерного решения представляют задачи, в которых предполагается произвольное задание исходных данных. Рассмотрим пример.

1. Формулировка задачи (Математика 1).

В одной звездочке 3 мальчика, а в другой — ... мальчиков.

Сколько всего мальчиков в этих двух звездочках?

2. Анализ.

В задаче дано (известно): в одной звездочке 3 мальчика, в другой — ...мальчиков.

В задаче надо найти (неизвестно): сколько всего мальчиков.

Дадим имена известным и неизвестным значениям:

Мальчики первой звездочки — Мальчики 1 зв, мальчики второй звездочки — Мальчики 2 зв, всего мальчиков — всего.

3. Краткие записи

Обычная

Первая звездочка — 3
Вторая звездочка — } всего — ?

Для «РЕШАТЕЛЯ»

Дано	Надо
Мальчики 1 зв 3	всего?
Мальчики 2 зв	

Пример объяснения краткой записи для «РЕШАТЕЛЯ»:

пишу Дано;

ниже пишу имя «Мальчики 1 зв», правее — число 3;

ниже пишу имя «Мальчики 2 зв»

правее, на строке Дано, пишу Надо;

ниже пишу имя «всего», правее — знак ?

4. Записи решения.

Обычная	Для «РЕШАТЕЛЯ»
для общего случая	всего = 3 + Мальчики 1 зв
отсутствует	всего = Мальчики 1 зв + Мальчики 2 зв

Программа «РЕШАТЕЛЬ» как модель электронной таблицы позволяет изменять исходные данные и по одной и той же формуле производить новые расчеты. В данном случае можно изменить значение величины с именем «Мальчики 2 зв». Например, при записи значения имени «Мальчики 2 зв» 5 и нажатии клавиши F4 имя «всего» получит значение 8 и т. д. Это позволяет ставить перед учащимися новые задачи, например просчет по записанной формуле нескольких значений. Таким образом уже с 1 класса можно формировать такие важные понятия как, «имя», «переменная», «значения переменной».

Программа «Процедура»

Программа позволяет учащемуся проверять правильность своих процедурных ответов для определенных задач. Общий вид рабочего поля экрана:

```
Задача 1. Толе 5 лет. Разница в возрасте Толи и
          его сестры Оли 2 года. Сколько лет Оле?

Ответ: если Оля старше Толи, то ей 7 лет,
       иначе ей 3 года

Оля старше Толи? да
ей 7 лет
```

Формирование таких важных понятий курса математики, как «больше», «меньше», «равно», в начальной школе не отражается в их использовании в качестве условий. Это связано с тем, что на страницах учебников и в методике не находят места задачи с процедурными ответами, предполагающие в результате решения получить не число или числа (как это имеет место в классе задач с решением — объект), а в форме предписания (алгоритма). С такими задачами учащиеся обычно знакомятся лишь в старших классах, когда решают задачи с параметрами и получают в ответах перечисление различных случаев, возникающих при варьировании этими параметрами. К сожалению, современные тенденции «совершенствования» математического образования привели к тому, что доля таких задач значительно уменьшилась, хотя роль их в формировании математической культуры, на наш взгляд, огромна.

Изучение задач с процедурными ответами в начальной школе, как показывают эксперименты, позволяет обогатить учебную деятельность учащихся. Этот класс задач значительно полнее и глубже отражает реальные ситуации, их решение требует от учащихся владения алгоритмическим стилем мышления, при этом создаются предпосылки целенаправленного формирования статистического и вероятностного мышления.

Для таких задач немаловажное значение имеет выбор языковых структур описания решений. В целом мы склонны считать, что их ассортимент может быть достаточно богатым (как богат ими сам язык). Однако в настоящее время, преследуя пропедевтическую цель, мы ориентируемся на структуры алгоритмического языка современного курса «Основы информатики и вычислительной техники».

Программа «Процедура» предлагает для решения серию задач, воспринимает ответы ученика в текстовой форме, задает вопросы, формулируемые на основе условий, и в зависимости от вводимых ответов на вопрос (форма ответа «да — нет») выдает результат: текст между «то» и «иначе» или после «ина-

че». Воспринимаются также структуры выбора:

если условие 1, то результат 1
если условие 2, то результат 2
. . . . . .
если условие N, то результат N

Предполагается развитие программы до работы со структурами типа «цикл».

Первый тип упражнений «процедуры ветвления» ориентирован на решение задач с процедурными ответами, записываемыми в следующей форме: если ..., то ..., иначе ... .

Рассмотрим примеры решения таких задач.

Задача 1. (Математика 1). Толе 5 лет. Разница в возрасте Толи и его сестры Оли — 2 года. Сколько лет Оле?

С учащимися полезно обсудить различные варианты записи ответа, подчеркивая тем самым богатство выбора записей условий и зависимость от этого записей результатов.

Возможные ответы:

1. Если Оля старше Толи, то ей 7 лет, иначе ей 3 года.
2. Если Оля младше Толи, то ей 3 года, иначе ей 7 лет.
3. Если Толя старше Оли, то ей 3 года, иначе ей 7 лет.
4. Если Толя младше Оли, то ей 7 лет, иначе ей 3 года.

Можно подобрать задачи, в которых анализ ситуации приводит к многозначному выбору. Покажем это на примере.

Задача 2. (Математика 1). У Бори было 4 машины, а у Коли — 3. Коле могут купить еще 2 машины. У кого из ребят будет больше машин?

Ответ: если Коле купят 2 машины, то у Коли будет машин больше, иначе если Коле купят 1 машину, то машин будет поровну, иначе больше машин будет у Бори.

Особый интерес представляют задачи, в которых ответ может быть представлен со сложным условием.

Задача 3. (Математика 2). Сереже дали 28 копеек, и он совершил покупку. Сколько денег у него осталось, если он мог купить мороженое за 22 копейки, сок за 15 копеек, пирожное за 22 копейки?

Вначале сформулируем всевозможные суждения и им соответствующие ответы.

Суждение 1. Сережа купил мороженое. Ответ 1: осталось 6 коп.

Суждение 2. Сережа купил сок. Ответ 2: осталось 13 коп.

Суждение 3. Сережа купил пирожное. Ответ 3: осталось 6 коп.

Используем выделенные суждения в качестве условий при записи общего ответа.

Если Сережа купил мороженое, то у него осталось 6 коп.

Если Сережа купил сок, то у него осталось 13 коп.

Если Сережа купил пирожное, то у него осталось 6 коп.

Можно поступить иначе, проведя другие рассуждения: у Сережи может остаться либо 6 копеек, либо 13 копеек. Поэтому форма ответа может быть несколько иной.

Вариант 1. Если Сережа купил мороженое или пирожное, то у него осталось 6 копеек, иначе у него осталось 13 копеек.

Вариант 2. Если Сережа купил сок, то у него осталось 13 копеек, иначе у него осталось 6 копеек.

Представляют интерес и задачи с ответом вероятностного характера.

Задача 4. На одном участке посеяли 304 ц пшеницы, а на другом — на 29 ц больше. С первого участка обычно собирают в 21 раз больше, а со второго — в 24 раза больше, чем посеяли. В случае засушливой погоды урожай сокращается в три раза. Какой урожай можно ожидать с двух участков?

Ответ: если погода будет не засушливая, то урожай может быть 14 374 ц, иначе урожай может быть 4458 ц.

Рассмотрим примеры задач для I—III классов с процедурными ответами со структурой выбора:

если условие 1, то результат 1
если условие 2, то результат 2
. . . . . .
если условие N, то результат N

Задача 5. (Математика 1). Миша посадил 5 горошин. Несколько из них дали ростки. Сколько горошин не дали ростков?

Ответ:

если дала росток 1, то не дали 4;
если дали ростки 2, то не дали 3;
если дали ростки 3, то не дали 2;

если дали ростки 4, то не дал 1;
если дали ростки 5, то не давших ростков нет!

Задача 6. (Математика 2). Аня купила 8 тетрадей. 4 она отдала Юре и несколько тетрадей взял Дима. Сколько тетрадей осталось у Ани?

Ответ:

если Дима взял 1 тетрадь, то у Ани осталось 3;

если Дима взял 2 тетради, то у Ани осталось 2;

если Дима взял 3 тетради, то у Ани осталась 1;

если Дима взял 4 тетради, то у Ани ничего не осталось.

Задача 7. (Математика 3). Спичечный автомат пускает и упаковывает спички в коробки. Его производительность 30 000 спичек в минуту. Он может настраиваться на упаковку коробок по 50, 60 и 100 спичек. Сколько коробок в минуту делает автомат?

Ответ:

Если автомат настроен на упаковку коробки в 50 спичек, то он выпускает 600 коробок в минуту.

Если автомат настроен на упаковку коробки в 60 спичек, то он выпускает 500 коробок в минуту.

Если автомат настроен на упаковку коробки в 100 спичек, то он выпускает 300 коробок в минуту.

Продолжение следует

В. ТИЩЕНКО

Работа простейшего компьютера

Выводы Д0 — Д7 микропроцессора служат для ввода и вывода восьмиразрядных двоичных чисел. Это может быть информация, подлежащая обработке, или коды операций, указывающие микропроцессору, какую обработку он должен выполнить. Так, например, подавая на выводы Д0 — Д7 код операции 10000000, мы укажем микропроцессору на необходимость сложения чисел, находящихся в регистрах А и В, а если подать код 10010000, то микропроцессор поймет это как приказ вычесть число, находящееся в регистре В, из числа, находящегося в регистре А.

Перечень двоичных кодов операций и описание действий, которые производит по этим приказам микропроцессор, называют системой команд.

Часть системы команд микропроцессора КР580ИК80А приведена в табл. 3.

Помимо рассмотренных выводов А0 — А15 и Д0—Д7 микропроцессор КР580ИК80А имеет 12 управляющих выводов. Коротко расскажем о них.

Название вывода ПМ — сокращение слова *прием*. Сюда микропроцессор выдает управляющий сигнал, указывающий на то, что он находится в режиме приема, т. е. может принять данные на выводы Д0—Д7 по шине данных из памяти или внешнего устройства.

ВД — сокращение слова *выдача*. Сигнал на этом выводе появляется, когда микропроцессор может выдать данные по шине данных в память или внешнее устройство.

ГТ — *готов*. Через ГТ микропроцессор получает сигнал от внешнего устройства, сообщающий о готовности последнего обмениваться данными с микропроцессором. Если микропроцессору нужны данные извне, а сигнала на выводе ГТ нет, микропроцессор останавливает свою работу и переходит в режим ожидания.

ОЖ — *ожидание*. На этом выводе появляется выходной сигнал микропроцессора, извещающий о том, что он ожидает сигнала от внешних устройств об их готовности к обмену данными.

ЗХ — *захват шин*. На вывод ЗХ поступает сигнал от внешних устройств, означающий запрос разрешения на использование (захват) ими шин адреса и данных, подключенных к микропроцессору. Этот захват бывает необходим для прямого доступа внешних устройств к памяти, минуя микропроцессор.

ПЗХ — *подтверждение захвата шин*. На этот вывод микропроцессор выдает сигнал, разрешающий внешним устройствам использовать шины данных и адреса, отключая одновременно свои выводы А0—А15 и Д0—Д7 от этих шин.

ЗПР — *запрос прерывания*. На этот вывод поступает сигнал от внешних устройств, требующий прерывания основной программы, выполняемой микропроцессором, и внеочередного обслуживания более важной и нужной программы внешнего устройства.

РПР — *разрешение прерывания*. Когда микропроцессор подает сигнал на вывод РПР, это означает, что он прервал выполнение своей программы и готов выполнить внеочередную программу внешнего устройства.

С — *синхронизация*. Этот вывод микропроцессор использует для выдачи сигнала о начале очередного такта работы, о том, что он приступил к очередному этапу выполнения команды.

СБР — *сброс*. Через вывод СБР в микропроцессор поступает входной сигнал, сбрасывающий счетчик адреса в начальное, нулевое состояние. Счетчиком адреса называется специальный регистр микропроцессора, в котором находится адрес той команды, которую нужно будет выполнить следующей. Поэтому обнуление счетчика адреса заставляет микропроцессор начать работу с выполнения команды, записанной в ячейке памяти с нулевым адресом.

На выводы Ф1 и Ф2 поступают сигналы запуска из тактового генератора. В результате действия этих сигналов микропроцес-

Окончание. Начало см.: Информатика и образование. 1991. № 1.

Таблица 3

№ п/п	Описание команды	Двоичный код	Условное обозначение
1.	Записать в регистр A число K	0 0 1 1 1 1 1 0 число K	MVI A, K
2.	Записать в регистр B число K	0 0 0 0 0 1 1 0 число K	MVI B, K
3.	Переслать число из регистра A в регистр B	0 1 1 1 1 0 0 0	MOV B, A
4.	Переслать число из регистра B в регистра A	0 1 0 0 0 1 1 1	MOV A, B
5.	Переслать число из регистра A в ячейку памяти, адрес которой записан во втором и третьем байтах команды	0 0 1 1 0 0 1 0 младшие разряды адреса старшие разряды адреса	STA (ADR)
6.	Переслать число в регистр A из ячейки памяти, адрес которой записан во втором и третьем байтах команды	0 0 1 1 1 0 1 0 младшие разряды адреса старшие разряды адреса	LDA (ADR)
7.	Сложить числа, находящиеся в регистрах A и B	1 0 0 0 0 0 0 0	ADD B
8.	Сложить число, находящееся в регистре A, с числом K, записанным во втором байте команды	1 1 0 0 0 1 1 0 число K	ADI K
9.	Вычесть из числа, находящегося в регистре A, число, находящееся в регистре B	1 0 0 1 0 0 0 0	SUB B
10.	Вычесть из числа, находящегося в регистре A, число K, записанное во втором байте команды	1 1 0 1 0 1 1 0 число K	SUI K
11.	Перейти к выполнению команды, адрес которой записан во втором и третьем байтах данной команды	1 1 0 0 0 0 1 1 младшие разряды адреса старшие разряды адреса	JMP (ADR)
12.	Остановить выполнение программы	0 1 1 1 0 1 1 0	HLT

сор выполняет один такт своей работы и переходит в режим ожидания следующих сигналов Ф1 и Ф2.

На все перечисленные выводы управления микропроцессор выдает или получает от внешних устройств управляющие сигналы в виде высокого или низкого электрического напряжения.

Кроме русских сокращений выводы зачастую обозначают английскими.

В заключение коротко расскажем о том, как работает микропроцессор.

Перед началом работы в ячейки памяти компьютера помещают нужную информацию в виде двоичных кодов: программу (последовательность операций) обработки данных и сами эти данные. Микропроцессор, обращаясь к ячейкам памяти, последовательно считывает в свои регистры коды данных и коды операций, а затем выполняет операции. Выполняет, естественно, в том порядке, который задумал и заложил в память компьютера программист.

Модуль вывода

Наиболее простым устройством вывода двоичных чисел, обработанных микропроцессором, служат светодиоды, подключаемые к шине данных и сообщающие своим свечением об уровне напряжения (соответствующем единице или нулю) на каждом проводе (Д0—Д7) шины данных.

Более сложное устройство вывода — дисплей. Это специальный телевизор, на экране которого могут воспроизводиться не только двоичные коды чисел, но и та информа-

ция, которая ими закодирована. Преобразует двоичные числа в текст, рисунки или графики сложная электронная схема.

Существует много конструкций принтеров, печатающих на бумаге цифры, текст и даже рисунки. Простейшие принтеры почти не отличаются от электрических пишущих машинок.

При работе компьютера в качестве устройства управления станками или бытовыми приборами устройством вывода служит исполнительный механизм, преобразующий выходной двоичный код микропроцессора в электрическое напряжение, механическое перемещение, изменение давления газа и т. п., синтезаторы речи.

Программирование

Чтобы написать программу для микропроцессора, необходимо знать те конкретные действия, которые он способен выполнять, т. е. знать его систему (список) команд.

В систему команд микропроцессора КР580ИК80А входит более 200 операций (команд).

В табл. 3 приведена часть этой системы: перечень некоторых команд, необходимых для составления простых программ. В четвертой колонке этой таблицы приведены общепринятые мнемонические обозначения команд. Они используются в языке ассемблера, программировать на котором в нашем простейшем компьютере, увы, нельзя, однако знать полезно.

Команды микропроцессора бывают простыми и сложными. Для выполнения простой команды (например, команда № 3) достаточно сообщить микропроцессору одно восьмиразрядное двоичное число — код этой команды (для команды № 3—01111000), т. е. один байт информации. Более сложные команды (№ 1 или № 2) определяются двумя байтами, а еще более сложные (№ 5) — тремя.

Кроме машинного языка (третья колонка табл. 3) и языка ассемблера (четвертая колонка) существует множество (несколько тысяч!) других языков программирования. Однако микропроцессор понимает только машинный язык своей системы команд, «буквами» которого служат нули и единицы. На каком бы другом языке ни была написана программа, ее перед выполнением нужно переводить на машинный язык. Перевод осуществляется с помощью специальных программ — трансляторов (интерпретаторов и компиляторов), размеры которых достигают десятков тысяч команд.

Есть разные подходы к изучению языков программирования. Опыт работы кружка «Компьютер» Киевской городской станции юных техников показал, что изучение программирования с кружковцами лучше всего начинать с машинного языка, органически связанного с компьютером, собираемым ими. Эта связь обусловлена тем, что нельзя понять устройство компьютера, не зная основ машинного языка, так же как невозможно освоить машинный язык, не понимая основ работы компьютера (особенно его микропроцессора).

Освоив машинный язык, следует переходить к языку ассемблера, и лишь потом, когда в кружке появится современный персональный компьютер, снабженный полной клавиатурой, дисплеем и интерпретатором, можно приступить к освоению Бейсика, Лого, Фортрана или других языков программирования.

Перейдем к практическому освоению машинного языка. Попробуем написать на нем простейшую учебную программу сложения двух чисел — например, сложения двойки и тройки с последующей пересылкой суммы в двенадцатую ячейку памяти для хранения и прочтения.

Прежде всего необходимо наметить общий план решения поставленной задачи в соответствии с возможностями нашего микропроцессора и его системой команд. Такой план может сводиться к тому, что исходные числа (2 и 3) следует записать в регистры микропроцессора, затем сложить в АЛУ, а потом переслать результат сложения в ячейку памяти.

Учитывая, что микропроцессор КР580ИК80А операцию сложения выполняет только тогда, когда одно из слагаемых находится в его регистре А, нужно в качестве первой команды нашей программы использовать команду № 1 (из табл. 3). Она двухбайтовая. Первый байт

00111110 (1)

представляет собой код операции записи числа в регистр А, а второй байт должен содержать записываемое число, в нашем случае 2:

00000010. (2)

Пользуясь командой № 2, можно записать в регистр В второе слагаемое.

Первый байт (код операции):

00000110 (3)

Второй байт (двоичная запись числа 3):

00000011 (4)

Таким образом, после выполнения микропроцессором команд, записанных в четырех байтах (1)—(4), заданные числа будут занесены в его регистры А и В.

Чтобы сложить эти числа, следует воспользоваться командой № 7, код операции которой

10000000 (5).

Эта команда однобайтовая. В процессе ее выполнения числа из А и В будут сложены в АЛУ, а результат сложения будет помещен в регистр А вместо находившегося там ранее числа 3. Чтобы переслать результат из регистра А в ячейку памяти № 12, воспользуемся командой № 5.

Первый ее байт — код операции пересылки числа:

00110010 (6)

Второй и третий байты — младшие и старшие разряды адреса ячейки памяти, в которую нужно переслать число. Так как адрес двенадцатой ячейки выражается в двоичной форме числом

0000000000001100,

то второй байт команды будет иметь вид

00001100, (7)

а третий —

00000000. (8)

Итак, наша программа будет состоять из восьми байтов (1)—(8). Если каждый ее байт поместить в подряд расположенные ячейки памяти, начиная с нулевой (как это указано в табл. 4), то микропроцессор, последовательно обращаясь к этим ячейкам (т. е. считывая из них информацию), выполнит программу и в ячейке с адресом 0000000000001100 (№ 12) окажется двоичное число 00000101 — сумма двойки и тройки.

Таблица 4

Адрес ячейки памяти	Содержимое ячейки памяти
0000000000000000	00111110
0000000000000001	00000010
0000000000000010	00000110
0000000000000011	00000011
0000000000000100	10000000
0000000000000101	00110010
0000000000000110	00001100
0000000000000111	00000000

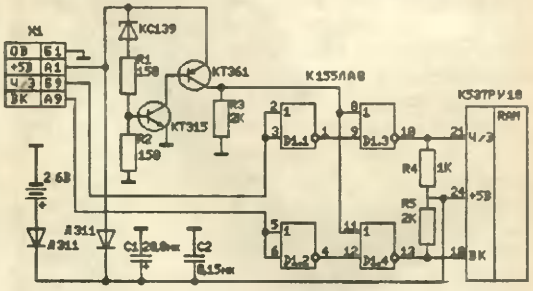
Для наглядности эту программу рекомендуется выполнять в пошаговом режиме. После восьмого нажатия кнопки ШАГ, расположенной в модуле управления, адресные светодиоды автоматически высветят адрес ячейки памяти (№ 12), в которую пересылается результат вычислений, а светодиоды данных покажут само пересылаемое число (5).

С помощью команд табл. 3 можно составлять только простейшие учебные программы. Для написания более сложных, практических полезных программ необходимо знать не только основные команды КР580ИК80А, но и правила их выполнения, а также ознакомиться с опущенными здесь особенностями работы микропроцессора (использование слова состояния, обращение к стековой памяти, выполнение прерываний и др.). Все это довольно сложно и требует постепенного и кропотливого изучения специальной литературы. А пока — два совета, позволяющих получить практический эффект от компьютера уже сейчас.

Даже очень простые программы позволяют включать и выключать электрические устройства (елочные гирлянды, реле, световые и звуковые приборы). Для этого используются свободные от работы адресные выводы А11—А15 микропроцессора. Подключение устройств выполняют через согласующий элемент (например, половину микросхемы К155ТМ2).

Включение устройства происходит, когда микропроцессор переходит к выполнению команды с адресом, имеющим единичный уровень в том разряде А11—А15, к проводу которого подключено устройство.

Малый ток, потребляемый микросхемой памяти К537РУ10, позволяет превратить наше ОЗУ в энергонезависимое путем подключения резервного источника питания в виде двух гальванических элементов. Эту переделку можно выполнить путем отключения трех выводов (21, 24 и 8) микросхемы от контактов разъема общей шины и подключения этих выводов по схеме рис. 2. Теперь



содержимое ячеек памяти не исчезнет при выключении питания, и в ОЗУ можно хранить десятки разнообразных программ, обращаясь к ним при помощи команды № 11. Для этого в начальную (нулевую) ячейку памяти записывают код этой команды (11000011), а в две последующие ячейки — младший и старший полуадресы той ячейки памяти, в которой находится первая команда вызываемой программы.

А. ЕФРЕМОВ, Ю. СЕНИН

Универсальная программа для обучения и контроля знаний по физике

На подготовительном факультете Университета дружбы народов им. П. Лумумбы студенты-иностранцы изучают часть курса элементарной физики, сходного с тем что входит в программу среднего образования: механика, молекулярная физика и электричество. До последнего времени преподавание этих разделов велось традиционными методами лекционно-практических и лабораторных занятий (по существу — уроков), что оказалось не вполне удовлетворительным как с точки зрения результатов обучения — студент не всегда проявляли интерес к повторному прослушиванию (хотя и на другом языке) уже известного им школьного курса,— так и с точки зрения современных технических возможностей. Кроме того, на изучение указанных разделов отведено весьма небольшое количество часов, время же активного обучения оказывалось еще меньше, поскольку почти 10 % его затрачивалось на проведение контрольных работ.

Появление на факультете персональных компьютеров явилось стимулом для интенсификации обучения с помощью программных средств. На факультете была создана творческая группа, которая в краткие сроки смогла решить две задачи: разработать методику единой системы тестирования (и обучения) по физике и составить универсальную компьютерную программу, удовлетворяющую требованиям этой методики (более того, допускающую использование по весьма широкому диапазону учебных дисциплин). Кратко изложим основные положения единой системы тестирования по физике (ЕСТФ).

Ясно сознавая, что решающим звеном в процессе передачи знаний является общение студентов с преподавателем и что личность последнего обеспечивает 90 % успеха, авторы ЕСТФ определили ей вспомогательное назначение — как методическому приему, рассчитанному, главным образом, для самостоятельной работы и контроля. Как уже отмечалось, был также расчет на сокращение непродуктивно затрачиваемого учебного времени и на использование моды на компьютеры, чтобы заинтересовать студентов в изучении предмета с их помощью.

В основу ЕСТФ положен принцип разделения изучаемого материала на логически связанные темы, заданные в рациональной последовательности (которая в основном соответствует изложению и духу учебника «Механика», изданного в УДН). Выбранная последовательность изложения материала, сходная с принятой в классическом курсе элементарной физики, в значительной степени соответствует истории процесса познания законов природы, что, отвечая концепции «от простого — к сложному», придает процессу обучения и усвоения предмета наибольшую эффективность. Темы состоят из нескольких разделов, каждый из которых содержит краткое изложение теории и многовариантный набор задач-тестов (по каждому разделу 20—30 задач). Приведем характерный пример. Тема «Кинематика» из части «Механика» включает разделы «Векторная алгебра» (30 задач), «Кинематические характеристики и виды движения» (30 задач), «Равномерное движение» (20 задач), «Равноускоренное движение» (30 задач), «Движение с ускорением g» (20 задач), «Кинематика движения по окружности» (20 задач). Аналогичным образом разбиты на разделы и другие темы.

На базе ЕСТФ разработана универсальная компьютерная программа «Импульс», предназначенная для обучения и контроля знаний. Это достаточно сложная система, которая состоит из головной программы, осуществляющей общее управление, набора вспомогательных и сервисных программ, текстов упражнений, файлов с упакованными иллюстрациями к задачам и файла прямого доступа, содержащего теоретические тесты по разделам ЕСТФ. Кроме того, имеется специальная автономная программа, служащая для ввода поясняющей информации в файл с прямым доступом. Программа действует в диалоговом режиме, допускает исправления и дополнения введенных ранее текстов, а также создание новых текстов. Для введения иллюстраций к задачам был создан независимый комплекс программ, позволяющих вводить картинки с помощью сканера, собирать их в один файл, дополнять скомпонованный файл картинками к новым задачам и т. п. Пакет программ реализован на языке Турбо-Паскаль 5.5 и может быть использован на компьютерах типа IBM XT/AT и совместимых с ними. Имеется версия для ЭВМ ЕС-1840 и ЕС-1841.

Возможности системы «Импульс» доволь-

но широки. Она предлагает учащемуся:

серию основных задач некоторого уровня трудности по данному разделу;

серию дополнительных, менее трудных, задач при неудачном решении предыдущих;

краткое изложение теории, учитывающей специфику сделанных при решении ошибок;

возможность дополнительного обдумывания правильности каждого ответа.

Кроме того, системы автоматически учитывает и хранит в специальном файле следующую информацию:

фамилию, имя, группу учащегося;

дату работы и затраченное на нее время (фиксируются автоматически);

номер варианта теста;

число правильно и неправильно решенных основных и дополнительных задач;

оценку, полученную учащимся по результатам тестирования.

При всей сложности структуры работа с программой весьма проста и для преподавателя, и для учащегося. После запуска программы учащийся (возможно, с помощью преподавателя) вводит с клавиатуры свои данные — фамилию, имя и группу. Все остальные манипуляции осуществляются простым перемещением курсора. На экране появляется список уроков по данному разделу, и установкой курсора выбирается один из них. Учащийся видит текст первой задачи, иллюстрацию к ней, а также ряд альтернативных ответов. Проанализировав задачу, учащийся выбирает один ответ, после чего получает запрос, уверен ли он в правильности данного решения. Практика показывает, что возможность дополнительного обдумывания хода решения, предоставляемая этим приемом, в большинстве случаев оказывается весьма плодотворной как побуждение к углубленному аналитическому мышлению. За уверенный правильный ответ дается высшая оценка (+4), за уверенный неправильный ответ — наименьшая оценка (—2); в случае неуверенности оценки соответственно равны +2 и 0. Каждый результат имеет свое музыкальное сопровождение, которое при желании может быть отключено.

После правильного решения задачи на экране появляется текст следующей основной задачи (всего в одном уроке четыре задачи, однако это число строго не фиксировано). Если же решение неправильно, то учащемуся предоставляется возможность ознакомиться с кратко изложенной теорией по данному разделу; впрочем, учащийся может отказаться от этого. В любом случае он затем получает текст аналогичной задачи низшего уровня сложности. После любого результата решения этой «штрафной» задачи на экране появляется текст второй задачи основного уровня.

По окончании работы учащийся (или преподаватель) может посмотреть результаты: число правильно и неправильно решенных задач, время работы и окончательную оценку (которая, конечно, может оказаться и отрицательной).

Когда тест в группе завершен, преподаватель собирает именные дискеты всех учащихся и может для удобства последующего анализа успеваемости переписать результирующую информацию на отдельную групповую дискету.

Несколько слов об использованной в системе «Импульс» методике альтернативных ответов. Известно, что в отечественной, да и в зарубежной школе эта методика неоднократно подвергалась критике за то, что не столько способствует развитию аналитического мышления учащихся, сколько вырабатывает у них механизм моторного запоминания числовых комбинаций, соответствующих правильным ответам. Если число вариантов таких комбинаций невелико, то подобная критика, безусловно, справедлива. Однако использование ЭВМ позволяет расширить число вариантов до таких пределов, что работа по запоминанию становится бессмысленной, и учащиеся предпочитают процесс решения процессу механического запоминания; тем более, к решению их стимулирует диалог с компьютером. Есть и другие причины того, что вместо набираемого на клавиатуре ответа (что, кстати, несложно применить в системе «Импульс») мы использовали метод альтернативных ответов: во-первых, отсутствие необходимости набирать ответ на клавиатуре исключает появление дополнительных ошибок при наборе, что существенно при обучении лиц, плохо знающих русский язык; во-вторых (и это главное), альтернативные ответы составлены таким образом, что они учитывают возможные ошибки, допускаемые учащимися. Это позволяет каждому неправильному ответу поставить в соответствие надлежащий блок теоретического материала, что и сделано в предлагаемом автоматизированном варианте ЕСТФ.

В заключение заметим, что предложенная идеология учебно-контролирующей программы, воплощенная в системе «Импульс», может использоваться для создания систем обучения в других областях знаний, например в биологии, в математике, при обучении языкам и т. д. В этом смысле система «Импульс» универсальна. Для создания новой учебно-контролирующей системы необходимо лишь разработать упражнения, тексты теоретических разделов, нарисовать

иллюстрирующие картинки определенного размера и... обратиться по адресу: 117198, Москва, ул. Миклухо-Маклая, 4, Университет дружбы народов им. Патриса Лумумбы, подготовительный факультет, кафедра физики, Ефремову А. П., Сенину Ю. Е. (тел. 433-01-01) или 113105, Москва, 1-й Нагатинский пр., 6, НТК КАППА (тел. 111-03-25, факс 9306430 или 1113116).

По вопросам приобретения самой системы «Импульс» можно обратиться по тем же адресам. Мы обучаем преподавателей работе на системе (цены для школ — не коммерческие!). Не забывайте, что наши дети должны войти в XXI век полноценными членами развивающейся цивилизации, а не варварским племенем, отгороженным от всего мира компьютерным непониманием.

Хотим также сообщить, что обучение советских студентов в Университете дружбы народов, которое с нынешнего года проводится по принятой в странах Запада двухступенчатой системе: бакалавр — магистр, потребует от них определенных навыков в работе с микро-ЭВМ. Так что — звоните, пишите, внедряйте!

Л. УТКО, Е. ПАЗЮРА

Использование графики при разработке обучающихся программ

В соответствии с принципом «лучше один раз увидеть, чем сто раз услышать» доля графической информации в обучающих программах постоянно возрастает. Современные требования к программам, используемым в учебном процессе, диктуют увеличение этой части до 40—90 % от общего объема отображаемой информации [1]. При разработке моделирующих программ важно использовать динамическую графику, т. е. строить на экране монитора постоянно изменяющееся изображение. При этом графические построения должны адекватно отражать соответствующие модели и процессы, так как в противном случае целесообразность применения графики зачастую сводится к нулю.

Как же строить изображение на экране? Предоставляемые для этого возможности делятся на аппаратные и программные. Используемые в народном образовании компьютеры линии INTEL («Ямаха», «Корвет», ЕС-1840) и линии DEC (ДВК, УКНЦ, БК) существенно различаются по своим возможностям. Компьютеры первого типа дают более широкие возможности работы с графической информацией, и соответствующее программное обеспечение (ПО) делает работу с графикой относительно простой. Как следствие — большее количество и лучшее качество разработанных (в том числе и учащимися) педагогических программных средств (ППС).

Сложнее дело обстоит с ПО для компьютеров второго типа. Они отличаются более слабыми аппаратными возможностями работы с графикой и практически отсутствующими сервисными средствами в стандартном ПО. Поэтому дальше мы будем ориентировать на компьютеры линии DEC, хотя предлагаемые методы реализации графических процедур пригодны для любого компьютера.

Рассмотрим аппаратные возможности компьютера на примере ДВК. Работу с графикой на ДВК обеспечивает плата КГД (контроллер графических данных) [2]. КГД позволяет работать с экраном 400×286 точек с помощью трех специальных регистров: регистра управления (адрес 176640), регистра адреса (176642) и регистра данных (176644).

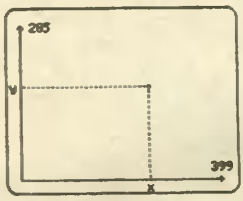
Регистр управления (РУ) используется для управления выводом информации на экран. Бит 15 этого регистра позволяет разрешить (значение 1) или запретить (значение 0) отображение на экране графической информации. Бит 14 позволяет разрешить (значение 0) или запретить (значение 1) отображение на экране символьной информации.

Регистр адреса (РА) используется для задания адреса чтения или записи информации в буфере КГД, из которого эта информация отображается на экран в режиме 1 бит — 1 точка. Соответствие адресов байтов в буфере и точек на экране приведено в таблице.

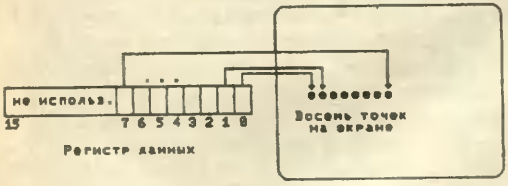
Таблица

Номера строк	Номера точек в строке			
	1—8	9—16	...	393—400
Строка 1	0	1		49
Строка 2	50	51		99
...	Адреса байтов в буфере КГД			
Строка 286	14250	14251		14299

Таким образом при расположении начала координат в левом нижнем углу экрана (рис. 1) адрес байта, содержащего точку с координатами (X, Y), можно вычислить по формуле A=14250—Y×50+[X/8], где квадратные скобки обозначают целую часть числа.



Регистр данных (РД) используется для записи (чтения) данных по адресу, находящемуся в РА. Младшие 8 битов регистра соответствуют 8 точкам на экране (1 — белая точка, 0 — черная). Отображение информации из буфера КГД на экран монитора для каждого байта выполняется в соответствии с рис. 2, т. е. номер бита в байте вычисляется по формуле N={X/8}, где фигурные скобки обозначают остаток от деления.



Таким образом, для отображения светлой точки с координатами X=18, Y=20 необходимо:

1) разрешить отображение графической информации, т. е. в РУ установить бит 15 в 1;

2) вычислить адрес байта в буфере, содержащего данную точку, и полученное значение поместить в РА:

A=14250—20×50+[18/8]=13248;

3) в регистре данных установить бит 2 ({18/8}) в 1, не изменяя значения остальных битов.

Для УКНЦ алгоритм отображения точки будет значительно сложнее. Естественно, что написать программу формирования рисунка и тем более адаптировать ее на другой компьютер — задача не из простых. Распространение графического Бейсика [3, 4] упрощает разработку и адаптацию ППС, но остаются сложности: ограниченные возможности работы с графикой и малый объем оперативной памяти для программы пользователя. Использование языка Паскаль снимает ограничение на объем программы (оверлейные структуры), но работать с графикой приходится на аппаратном уровне, так как стандарт языка для ЭВМ линии DEC не предусматривает работы с графикой.

Описанный выше алгоритм вывода точки на экран на языке Паскаль для ДВК можно реализовать в виде следующей процедуры.

```
PROCEDURE POINT(X,Y:INTEGER);
CONST ADR=14250;
VAR   RA ORIGIN 176642,
      RD ORIGIN 176644,
      RU ORIGIN 176640:INTEGER;
      I:INTEGER;
      M:ARRAY [0..7] OF INTEGER;
BEGIN
  RU:=100000B;
          (* РАЗРЕШИТЬ ВЫВОД ГРАФИКИ *)
  RA:=ADR-Y*50+(X DIV 8);
          (* АДРЕС ТОЧКИ              *)

  (* СФОРМИРОВАТЬ МАССИВ МАСОК *)
  (* В M[I] БИТ I УСТАНОВЛЕН В 1,
                      ОСТАЛЬНЫЕ В 0 *)
  M[0]:=1;
  FOR I:=1 TO 7 DO
  M[I]:=M[I-1]*2;

  (* ОТОБРАЗИТЬ ТОЧКУ (СФОРМИРОВАТЬ
                          В БУФЕРЕ) *)
  RD:=RD OR M[X MOD 8]

END;
```

Такой уровень работы позволяет использовать все возможности компьютера, но, как уже отмечалось, существенно усложняет разработку и адаптацию программ.

Для облегчения разработки программ обычно идут по пути создания библиотек процедур работы с графической информацией. Перечень возможностей, предоставляемых процедурами библиотеки, зависит от ее назначения. Библиотека, используемая при разработке ППС, должна наряду с общепринятыми возможностями (работать с окнами, рисовать точку, линию, ломаную, кривую, окружность, дугу, закрасить или заштриховать контур, выбрать цвет и тип линии) предоставлять удобные средства работы с текстами (вывод как отдельных символов, так и текста в целом, изменение масштаба, наклона, поворота символов, замена таблицы символов для использования национального алфавита и т. п.) и обеспечивать возможность создания динамических рисунков.

В связи с тем что быстродействие используемых в учебном процессе компьютеров не позволяет обычным образом (стирание и перерисовка элементов рисунка) получать динамическое изображение, предлагаются следующие два способа обеспечения динамики. Оба они основаны на том факте, что изображение формируется в буфере монитора (видеопамять) и из него регенерируется на экран с определенной часто-

той. Учитывая это, вместо многократного стирания старого элемента рисунка и прорисовки его заново на новом месте достаточно просто выполнять сдвиг битов в заданном направлении в рамках некоторого окна в видеопамяти. При этом можно организовать перенос выпадающих за рамки окна точек в начало окна, т. е. исчезающие за окном элементы рисунка будут «выплывать» с другой стороны окна. Организованное таким образом движение выполняется достаточно быстро, так как не требует больших вычислений при перерисовке элементов (особенно при закраске и штриховке контуров).

Второй способ заключается в том, что нужные окна из видеопамяти можно запомнить в ОЗУ и затем из ОЗУ пересылать их в видеопамять на любое место в любом порядке.

Для решения задачи адаптации разрабатываемых ППС на другие компьютеры, используемые в учебном процессе, все процедуры библиотеки должны быть разделены на две группы: не связанные непосредственно с выводом на экран, например расчет координат точек окружности, и процедуры непосредственного вывода информации, т. е. вычисление адреса точки в видеопамяти, заполнение регистра адреса, формирование регистра данных в соответствии с заданным цветом и т. п. При такой организации библиотеки заменой аппаратно зависимых процедур на процедуры, поддерживающие работу другого компьютера, решается задача мобильности ППС на уровне исходного языка.

Использование указанных принципов построения и возможностей при создании библиотеки МИГ [5] позволило существенно расширить возможности ППС и облегчить работу по их созданию.

Информацию о приобретении МИГа можно получить по адресу: 220080, Минск, Ленинский просп., 4, БГУ им. В. И. Ленина, ФМП, кафедра ИПМОАП, Фонд алгоритмов и программ учебного назначения. Тел. 26-86-36.

Литература

1. Практическое руководство по написанию сценариев педагогических программных средств / *В. И. Бабич, О. В. Возняк* и др.— Минск: ИВЦ МП БССР, 1987.

2. Плата КГД. Техническое описание ПБА4.135.998 ТО.

3. Программное обеспечение комплекса вычислительного учебного «Электроника МС 0202». Бейсик. Описание языка. VI.00031-01 35 01.

4. Программное обеспечение комплекса вычислительного учебного «Электроника МС 0202». Бейсик. Руководство программиста. VI.00031-01 33 01.

5. Базовый пакет обеспечения машинной графики для ЭВМ линии DEC «МИГ»/*Л. З. Утко, Е. В. Пазюра, Е. П. Андрушкевич, И. И. Писарук*. РФАП АН БССР. Рег. № 2071817.00339.

Ю. ГОРВИЦ, Е. ЗВЕРЕВА, А. ГЛАЗКО
ЦНИИ «Электроника»

Печатаем рисунки в КУВТ-86

В процессе использования компьютера в обучении специалисты выявили потребность детей сохранить результаты своей деятельности за компьютером в виде «печатной продукции». В большей степени это относится к программам типа графических редакторов, развивающих игр конструктивного типа (конструкторов, мозаик, игр типа «Танграм», «Пентамино» и т. п.). С этой проблемой мы столкнулись при исследовании игровой деятельности дошкольников в компьютерно-игровом комплексе (КИК) «Электроник». Предлагаемый способ ее решения может быть использован и в любых учебных заведениях, оборудованных классом КУВТ-86.

Итак, в классе несколько учеников занимаются на своих компьютерах рисованием, конструированием, а печатающее устройство — одно на всех и подключено к компьютеру педагога, т. е. непосредственного доступа с рабочего места ученика (РМУ) к нему нет.

На наш взгляд, наиболее приемлемым вариантом коллективного использования принтера является подход, в основе которого лежат следующие принципы: во-первых, закодированный образ экрана каждого РМУ должен независимо (асинхронно) передаваться на рабочее место педагога (РМП); во-вторых, содержимое экрана должно сохраняться на магнитном диске в виде файла с уникальным именем, чтобы учитель или ученик могли получить твердую копию (распечатку) содержимого экрана в удобное время.

Для обеспечения такого режима должны быть созданы, с одной стороны, средства, встраиваемые в учебную (игровую, диагно-

стическую и т. д.) программу на Бейсике БК-0010.01 и выполняющие пересылку из экранной памяти компьютера требуемой информации в РМП, а с другой стороны — программа дешифровки и печати в искомом виде сформированного на магнитном диске файла.

Предполагается, что для обмена информацией между РМУ и РМП используется стандартная для КУВТ-86 программа сетевого обмена NET3 или ее аналог.

Рассмотрим подробнее разработанные нами программные средства, обеспечивающие решение поставленной задачи.

Подпрограмма на Бейсике для БК (условно назовем ее «посылка экрана», ПЭК) достаточно компактна для включения ее в любую прикладную программу на Бейсике.

Программа печати полученного файла на ДВК написана на Паскале, приведен текст варианта программы, рассчитанного на принтер Robotron-6329.02. Получить исполняемую программу можно либо в готовом виде у разработчиков, либо скомпилировав ее в соответствии с документацией «Система программирования Паскаль».

Подпрограмма ПЭК выполняет передачу по линии связи (в режиме работы с программой сетевого обмена NET3) экранной области ОЗУ с адреса 40 000 по 77 776, при этом на ДВК на диске ДК: создается файл размером 33 блока с именем С*.BIN, где * — любой символ (задаваемый в программе на БК).

Из-за того, что оператор Бейсика PRINT# при передаче преобразует любую информацию в текст, в подпрограмму ПЭК включена более эффективная внутренняя подпрограмма на языке ассемблера (она размещается в операторах DATA), которая записывается в стековую область памяти с адреса 420 по 554.

При обращении к подпрограмме ПЭК из основной программы запрашивается индекс, используемый для формирования имени файла, создаваемого на диске. Запрос индекса осуществляется в строке 22 экрана — это единственное ограничение при получении твердой копии экрана. После ввода индекса эта строка стирается пробелами. Если такой вариант формирования имени файла не подходит, пользователь может сделать строки программы 9000, 9001 и 9255 комментариями, а в основной программе перед обращением GOSUB 9000 внести в символьную переменную один символ, который прикомпонуется к имени файла при передаче (чтобы сделать его уникальным).

Распечатка образа экрана из файла, полученного на диске в результате передачи, осуществляется на ДВК в автономном режиме (без программы сетевого обмена NET3).

Программа печати запускается командой монитора

RU OTO.SAV

(если программа размещена на диске ДК:).

Программа запрашивает индекс файла, содержащего требуемый образ экрана, а затем проверяет, если ли на диске файл с соответствующим именем (например, если введен символ А, то программа ищет файл с именем CA.BIN). Если файл с этим индексом отсутствует на диске ДК, то программа сообщает о его отсутствии и повторно запрашивает индекс файла.

Файл, в каждом слове которого содержится информация о цвете восьми точек экранной области, считывается в ОЗУ, производятся необходимое преобразование и выдача информации на принтер. Печать осуществляется в точечном режиме двойной плотности по 960 позициям в строке.

К сожалению, тем, кто использует цветные мониторы и, соответственно, создает программы в расчете на цветное изображение, придется смириться с получаемым на принтере черно-белым изображением и учесть, что черному цвету на экране соответствует белый цвет на бумаге, красному цвету — черный, а синему и зеленому — серый.

Но и при этом ограничении предлагаемый способ получения твердой копии экрана на принтере, подключенном к центральной машине, все-таки позволяет существенно расширить возможности использования КУВТ-86 в воспитательном процессе в детских садах, в учебном процессе в школах и других учебных заведениях. Используя это средство, можно разрабатывать новые сценарии развивающих, обучающих и контролирующих компьютерных программ для детей дошкольного возраста и школьников.

```
9000 LOCATE 1,22
9001 INPUT "ВВЕДИТЕ ИНДЕКС
                        (ОДИН СИМВОЛ)";U0$
9100 DATA 5568,2,2527,-21908,2527,
       -21920,5571,10,5570,256,2551,58,
         2527,-21920,4098,769,135,5570,
           256,4298,5618,16384,2,5571,4
9110 DATA 2551,28,6275,4290,2565,
        -27520,2527,-21908,24581,32453,
          2527,-21920,4416,2527,-21908,
          135,-27520,2527,-21908,32452,
                                  135,0
9140 RESTORE 9100
9150 FOR U1%=&0420 TO &0554 STEP 2%
9160 READ IP%
9170 POKE U1%,IP%
9200 NEXT
9211 POKE &0400,ASC(U0$)*(2%^8%)+&0103
9220 POKE &0402,&020040
9230 POKE &0404,&020040
```

```
9240 POKE &0406,&041056
9250 POKE &0410,&047111
9255 ? AT(1,22) "
9260 DEF USR2=&0420
9270 A=USR2(U1%)
9300 RETURN
```

```
        PROGRAM ОТОБРАЖЕНИЕ
LABEL 1;
CONST READY =200B;
TYPE
   MASS = ARRAY[1..8193] OF INTEGER;
VAR
   OUTF : TEXT;
   OBR : FILE OF INTEGER;
   SYM : ARRAY[1..9] OF CHAR;
   MA : MASS;
   K,I,J,N,M,L,T : INTEGER;
   CSR ORIGIN 177514B,
   DAT ORIGIN 177516B : INTEGER;
PROCEDURE DRV;
   BEGIN
     WHILE L>=256 DO L:=L-256;
      L:=L+400B;
       WHILE (CSR AND READY)=0 DO;
      DAT:=L;
   WHILE (CSR AND READY)<>0 DO;
   END;
(*НАЧАЛО РАБОТЫ ПРОГРАММЫ ОТОБРАЖЕНИЯ*)
BEGIN
SYM:= DK:C .BIN';
 1:
WRITELN;READLN;
WRITE(CHR(16B),CHR(7),'**ВВЕДИТЕ ИНДЕКС
      ФАЙЛА (ОДИН СИМВОЛ) >',CHR(17B));
READ(SYM[5]);
RESET(OBR,SYM,,N);
   IF N=-1 THEN BEGIN
     WRITELN(CHR(16B),CHR(7),'   **НЕТ
           ФАЙЛА НА ДИСКЕ**',CHR(17B));
     GOTO 1;
    END;
   GET(OBR);GET(OBR);
FOR I:=1 TO 8192 DO BEGIN
      MA[I]:=OBR^;
      GET(OBR);
    END;
    CLOSE(OBR);
REWRITE(OUTF,'LP:');
    BREAK(OUTF);
      WHILE (CSR AND READY)=0 DO;
     DAT:=27;
      WHILE (CSR AND READY)=0 DO;
     DAT:=64;
      WHILE (CSR AND READY)=0 DO;
     DAT:=27;
      WHILE (CSR AND READY)=0 DO;
     DAT:=49;
FOR I:=0 TO 31 DO BEGIN
      WHILE (CSR AND READY)=0 DO;
     DAT:=27;
      WHILE (CSR AND READY)=0 DO;
     DAT:=76;
      WHILE (CSR AND READY)=0 DO;
     DAT:=1;
      WHILE (CSR AND READY)=0 DO;
     DAT:=2;
FOR J:=0 TO 31 DO BEGIN
   K:=J+I*256+1;
     FOR M:=0 TO 15 DO BEGIN
      IF M=0 THEN N:=1 ELSE BEGIN
      IF M<>15 THEN BEGIN N:=1; FOR
                              T:=1 TO M DO
       N:=N*2 END ELSE N:=100000B;END;
     L:=0;
   IF (MA[K] AND N)<>0 THEN L:=L+128;
       IF (MA[K+32] AND N)<>0 THEN
                                 L:=L+64;
        IF (MA[K+64] AND N)<>0 THEN
                                 L:=L+32;
         IF (MA[K+96] AND N)<>0 THEN
                                 L:=L+16;
          IF (MA[K+128] AND N)<>0 THEN
                                  L:=L+8;
         IF (MA[K+160] AND N)<>0 THEN
                                  L:=L+4;
        IF (MA[K+192] AND N)<>0 THEN
                                  L:=L+2;
       IF (MA[K+224] AND N)<>0 THEN
                                  L:=L+1;
     DRV;
   END;END;
     L:=0;
     DRV;
     L:=10;
     DRV;
   END;
      WHILE (CSR AND READY)=0 DO;
     DAT:=27;
          WHILE (CSR AND READY)=0 DO;
         DAT:=64;
      WRITE(CHR(17B));
  END.(*КОНЕЦ ПРОГРАММЫ*)
```


В августе 1991 г. в г. Троицке Московской области в центре информатики «Байтик» планируется проведение международной конференции «Использование новых информационных технологий в образовании».

За справками обращаться по адресу: 142092, Московская обл., г. Троицк, Сиреневый бульвар, д. 11, «Байтик».

Телефон в Москве: 334-03-67.

Организаторы конференции ищут спонсоров!

В. КАНИВЕЦ
Нежинский педагогический институт им. Н. В. Гоголя

Использование ассемблера в среде MSX-BASIC

Пользователи MSX-BASIC нередко сталкиваются с необходимостью программировать в машинных кодах — других средств для работы «на уровне процессора» в интерпретаторе практически нет. В то же время система MSX-BASIC предоставляет необходимый сервис в виде редактора текста и ассемблера (M80). Предлагаемая BASIC-программа (назовем ее BOOT) дает возможность использовать результаты работы M80. Несмотря на простоту, использование BOOT приводит к весьма эффективным результатам.

Продемонстрируем один из них на примере разработки справочной системы для MSX-BASIC. Программа-справочник HELP работает следующим образом. Будучи загруженной в выбранную пользователем область ОЗУ, программа HELP начинает работу по нажатию клавиши ESC. На экран выводится меню — список вопросов, по которым дается справка. Нажимая клавиши с номерами вопросов (в нашем примере «1», «2», «3»), пользователь получает соответствующий текст. Возврат в меню осуществляется нажатием клавиши «0», выход из справочника — «*».

Чтобы из текста на языке ассемблера получить работающую программу, необходимо выполнить следующие действия:

1. В системе MSX-DOS командой

M80 HELP,HELP=HELP.ASM

оттранслировать программу HELP.ASM, чтобы получить файл листинга HELP.PRN. В случае отсутствия ошибок перейти к следующему пункту.

2. В системе MSX-BASIC запустить программу BOOT.BAS. Последняя, запросив имя файла (HELP.PRN) и адрес загрузки (D000H), выбирает шестнадцатеричное кодовое представление программы из листинга и загружает с указанного адреса.

3. После очистки ОЗУ командой NEW запустить программу LHOOK.BAS, предназначенную для записи кода команды JP D000H в ячейки ловушки нажатия клавиши.

4. Очистив ОЗУ, можно проверять работу программы справочника.

К предложенной инструкции сделаем несколько замечаний:

пункты 2 и 3 можно объединить, «дописав» к BOOT.BAS соответствующим образом измененную программу LHOOK.BAS;

после того как правильность HELP.ASM не вызывает сомнений, имеет смысл сохранить на диске область ОЗУ, которую она занимает (командой BSAVE), чтобы в дальнейшем для загрузки использовать более удобную команду BLOAD или сетевую функцию BSEND;

для предотвращения перекрытия областей памяти, занимаемых программами пользователя и программой-справочником, область загрузки последней необходимо зарезервировать оператором CLEAR.

```
        ;Программа-справочник HELP.ASM
        .Z80
        ASEG
        ORG 0D000H
        JP  START
CHGET   EQU 009FH
  ESC   EQU 3AH
  CLS   EQU 00C3H
CHPUT   EQU 00A2H
  RUS:  DB 0H
   M0:  DB '1.(Оператор) IF ',0DH,1FH
        DB '2.(Оператор) GOTO',0DH,1FH
        DB '3.(Оператор) FOR - NEXT',
                                  0DH,1FH
        DB '(ВЫХОД В МЕНЮ) - 0 ',
                                  0DH,1FH
        DB '(КОНЕЦ РАБОТЫ) - * $ ',
                                  0DH,1FH
   M1:  DB '(Справка по оператору)
                              IF$',0DH,1FH
   M2:  DB '(Справка по оператору)
                            GOTO$',0DH,1FH
   M3:  DB '(Справка по оператору)
                            FOR$',0DH,1FH
START:  CP ESC
        RET NZ
        PUSH  AF
        PUSH BC
        PUSH DE
        PUSH HL
        LD HL,M0
L0:     PUSH HL
        LD A,00H
        BIT 0,A
        CALL CLS
        POP HL
   PR1: LD A,(HL)
        CP '$'
        JP Z,L1
        CP '('
        JP Z,UPRU
        CP ')'
        JP Z,DORU
        CP 41H
        JP M,LCH2
        LD A,(RUS)
        CP 0H
        JP Z,LCHP
        LD A,80H
        ADD A,(HL)
        JP LCH2
```

```
LCHP:   LD A,(HL)
LCH2:   CALL CHPUT
        JP LCH3
UPRU:   PUSH HL
        LD HL,RUS
```

```
5  REM ТЕКСТ ПРОГРАММЫ BOOT.BAS
10 PRINT"input listing name"
20 INPUT N$
25 PRINT"input hex load adres"
28 INPUT M$
30 OPEN N$ FOR INPUT AS #1
33 A1=VAL("&H"+M$)
35 IF EOF(1) THEN 70
40 LINE INPUT #1,S$
45 GOSUB 80
60 GOTO 35
70 END
80 IF MID$(S$,11,1)=" " THEN 200
85 IF MID$(S$,1,1)<>" " THEN 200
88 A$=MID$(S$,11,2)
90 PRINT MID$(S$,11,2)
94 POKE A1,VAL("&H"+A$):A1=A1+1
95 IF MID$(S$,14,1)=" " THEN 200
100 A$=MID$(S$,14,2)
110 IF MID$(S$,16,1)=" " THEN 150
120 B$=MID$(S$,16,2)
130 PRINT B$:PRINT A$
135 POKE A1,VAL("&H"+B$):A1=A1+1
138 POKE A1,VAL("&H"+A$):A1=A1+1
140 GOTO 200
150 PRINT A$
155 POKE A1,VAL("&H"+A$):A1=A1+1
160 IF MID$(S$,17,1)=" " THEN 200
170 B$=MID$(S$,17,2)
175 POKE A1,VAL("&H"+B$):A1=A1+1
180 PRINT B$
185 IF MID$(S$,20,1)=" " THEN 200
190 C$=MID$(S$,20,2)
193 POKE A1,VAL("&H"+C$):A1=A1+1
195 PRINT C$
200 RETURN
```

```
10 REM ТЕКСТ ПОДПРОГРАММЫ LHOOK.BAS
                        (ЗАГРУЗКА ЛОВУШКИ)
20 POKE &HFDCC,&HC3
30 POKE &HFDCD,0
40 POKE &HFDCE,&HD0
```

Программа HELP.ASM — иллюстративная. Этим и объясняется условность «справочного материала». Читатели, программирующие на ассемблере, без труда сами доведут ее до «товарного вида», а заинтересовавшимся больше справочной системой, чем процессом ее разработки, автор обязуется выслать все файлы реализованной им программы-справки на присланной дискете.

ДОСКА ОБЪЯВЛЕНИЙ

Хочу переписываться и обмениваться программами для ПК «ZX Spectrum». Хотел бы научиться работать на ассемблере. Имею дисковод.

170033, Тверь, пр. Волоколамский, 16—68. С. Б. Запруднов.

Создающийся клуб по переписке пользователей фирменных ПЭВМ IBM, «Коммодор», «Атари», «Спектрум», «Правец», «Амига» и т. д. приглашает всех желающих для обмена опытом, программами, литературой.

Пишите по адресу: 143400, Красногорск-4, а/я 54, В. В. Московцеву.

Ищу программы для ПЭВМ «Вектор-06Ц» и «COMX-PC1», хотел бы переписываться с владельцами этих компьютеров.

700000, Ташкент, ул. Жуковского, 43, кв. 7. Ш. Т. Даутов.

Готов обмениваться игровыми, системными, обучающими программами для УКНЦ.

398001, Липецк, ул. Советская, 45, кв. 49. А. В. Абрамову.

Хотел бы обмениваться программами и различной информацией с владельцами ПЭВМ «Вектор-06Ц», БК-0010.01, «ZX-Spectrum». Имею солидный банк программ, вышлю каталоги.

610004, Киров, а/я 1199. Н. Е. Таранцев.

Хотел бы переписываться, обмениваться программами с владельцами компьютера «Апогей БК-01».

301650, Тульская обл., г. Узловая, квартал «50 лет Октября», д. 21, кв. 76. И. Н. Калинкин.

Прошу откликнуться тех, кто может помочь с программами для КУВТ «Корвет».

623730, Свердловская обл., г. Реж-2, ул. Чапаева, 25, кв. 1. Д. Н. Посунько.

Хотел бы переписываться с работающими на «Ямахе-2».

614107, Пермь, а/я 4055, М. Смолев.

Хочу переписываться с владельцами «Искры 1030» или других IBM-совместимых ПЭВМ. Имею несколько интересных обучающих и игровых программ.

334510, Керчь, ул. Бульвар пионеров, 2/5, кв. 48, С. Шумляев.

Имею приличный набор игровых и системных программ, хотел бы обмениваться и переписываться с обладателями «Кристы» и совместимых с ней компьютеров («Микроши» и т. п.).

213320, Могилевская обл., г. Быков, Октябрьский пер., 5. С. Л. Ткач.

Автоматизированное рабочее место учителя информатики

В нашей школе силами кружка электронного конструирования создано автоматизированное место учителя информатики на базе КУВТ УКНЦ. Для этого разработаны и смонтированы четыре сервисные электронные системы. Все они надежно работают в интегрированном кабинете физики и информатики, повышая эффективность учебного процесса.

Каждая из систем совершенно не связана с локальной сетью и является хорошим дополнением к ней.

Первая система подробно описана в «ИНФО» № 3 за 1989 г. в статье «Электронный коммутатор видеосигнала для дисплейного класса». Она позволяет «копировать» изображение с монитора любого учащегося на монитор преподавателя в двух режимах: автоматического переключения (при обзорном контроле за работой учащихся) и индивидуальных консультаций (когда изображение с нужного РМУ «фиксируется» на мониторе учителя в течение любого промежутка времени).

Чтобы все учащиеся могли видеть копируемое изображение, на боковой стене класса установлены два демонстрационных монитора — цветной и черно-белый. Они выполнены на базе бытовых телевизоров с экраном 61 см по диагонали. Цветной монитор имеет встроенный кнопочный переключатель и работает в нескольких режимах.

Вторая система позволяет учителю, не покидая своего рабочего места, оперативно производить дистанционный перезапуск любой машины учащегося при сбоях.

Третья система предназначена для проведения индивидуальных консультаций по микрофону тому ученику, изображение с монитора которого в данный момент выведено к преподавателю. Для этого на столах учащихся установлены микродинамики, синхронно переключаемые системой при копировании изображений. Особенно целесообразно применение этой системы в тех случаях, когда РМГ удалено от РМУ (что имеет место в совмещенных кабинетах).

Четвертая система реализует обратную связь с учащимися. Мы назвали ее «Жду консультацию». Если в процессе работы у ученика появляется вопрос, он нажимает кнопку, установленную на столе. Простейшая ячейка памяти запоминает номер его машины, и на пульте преподавателя загорается соответствующий светодиод. Он будет гореть до тех пор, пока не проведена консультация. После этого учитель кнопкой сброса «обнуляет» ячейку памяти.

Основными компонентами перечисленных систем являются микросхемы серии К155, транзисторы КТ315, КТ361, низковольтные реле, светодиоды. Каждую из систем можно повторить в любом дисплейном классе.

Информацию по принципиальным схемам, чертежам печатных плат, особенностям монтажа можно получить по договоренности с автором по адресу: 613401, Кировская обл., Куменский р-н, речная средняя школа.

А. КАРАВАЕВ

В. АРТАМОНОВ

Однопроходной дизассемблер для «Корвета»

Системную программу, восстанавливающую исходный текст на языке ассемблера по загрузочному модулю, принято называть дизассемблером, а процесс восстановления — дизассемблированием. Такая программа часто бывает незаменима при проведении на компьютере различных системных работ, например при изучении или модификации программного обеспечения, исходные тексты которого отсутствуют. К сожалению, среди программ, поставляемых для КУВТ «Корвет», дизассемблер отсутствует. Пришлось восполнять этот недостаток самостоятельно.

Разработанный дизассемблер представляет собой простую однопроходную программу [1] на Бейсике, работающую по дампу памяти. Работа программы начинается с задания диапазона анализируемых адресов (значение адреса не должно превышать 65535) и интервалов пропускаемых адресов. Последнее требует некоторых пояснений. Дело в том, что наиболее сложной задачей в дизассемблировании считается задача разделения данных и команд. В данном случае для решения этой задачи избран наиболее простой путь: с помощью программы DUMP,

также разработанной по причине отсутствия аналога в поставляемом программном обеспечении, перед дизассемблированием делается дамп нужного участка памяти и определяются адреса, по которым содержатся данные. Найденные адреса (в виде интервалов пропускаемых адресов) задаются в начале работы дизассемблера. Если в процессе дизассемблирования текущий адрес попадает в один из интервалов пропускаемых адресов, а таких интервалов может быть задано до 100, то содержимое ячейки памяти по этому адресу интерпретируется только как данное; иначе — как данное и как команда.

Литература

1. *Безруков Н. Н.* Эвристические методы повышения качества дизассемблирования // Программирование. 1988. № 4.

Ю. ПЛЕТНЕВ

Использование функциональных ключей на УКНЦ

Пытаясь запрограммировать функциональные ключи на «Электронике МС 0511», я обнаружил на с. 30 руководства «Работа с внешними устройствами» (00031—01 90 01—ЛУ) неточность.

Для переопределения ключей необходимо выдать на терминал:

⟨НАЧ⟩ — начальную управляющую последовательность (коды 33 45 41 61 33 120 61);

⟨П1⟩, ⟨КЛН⟩, ⟨СТРN⟩ — непосредственно значения ключей;

⟨КОН⟩ — конечную управляющую последовательность (коды 33 57 33 45 41 63).

Ниже предлагается текст программы на Паскале (INS.PAS). Она транслируется обычным образом с получением загрузочного модуля INS.SAV; использует текстовый файл *.HLP и создает файл *.INS (например, PAS.INS из PAS.HLP). Вывод последнего на терминал командой TYPE приводит к перепрограммированию функциональных ключей в соответствии с файлом *.HLP. Программа предусматривает назначение 15 ключей. После ее запуска командой RUN INS в ответ на приглашение INS⟩ необходимо ввести имя соответствующего файла с расширением HLP (например, PAS.HLP).

Файл *.HLP создается с помощью любого текстового редактора. Значение ключа записывается с позиции 22; сообщение auto в позиции 12 вызывает автоматическое выполнение соответствующей команды.

Включение команд TYPE PAS.INS и TYPE PAS.HLP в стартовый файл STARTS.COM приведет к назначению функциональных ключей сразу после загрузки операционной системы. Создание нескольких файлов *.INS позволяет в процессе работы оперативно изменять значения ключей. Таким путем можно, например, облегчить работу при написании в редакторе текстов программ на Паскале.

```
PROGRAM INS(INPUT,OUTPUT);
 VAR I,J,K,LEN:INTEGER;
     DD,CH,CH1:CHAR;
     INF,OUTF:TEXT;
     DAN:ARRAY[1..25,1..80] OF CHAR;
     NAME:ARRAY[1..15] OF CHAR;
PROCEDURE HEX(DD:CHAR;VAR CH,CH1:CHAR);
   VAR I,J,D1:INTEGER;
   BEGIN
     D1:=ORD(DD);
     I:=D1 DIV 16;
     J:=D1 MOD 16;
     IF J>9 THEN J:=J+7;
     CH:=CHR(ORD('0')+I);
     CH1:=CHR(ORD('0')+J);
   END;
PROCEDURE INFORM;
   BEGIN
    WRITELN('
?INS-F-File not found
*INS V02 09.90');
   END;
   BEGIN(*main*)
     WRITE('INS>'); READLN(NAME);
     RESET(INF,NAME,'HLP',LEN);
     IF LEN=-1 THEN INFORM
       ELSE
       BEGIN
         FOR I=1 TO 20 DO
         FOR J=1 TO 80 DO DAN[I,J]:='_';
         I:=1;
         WHILE NOT EOF(INF) DO
           BEGIN
           J:=1;
           WHILE NOT EOLN(INF) DO
             BEGIN
               READ(INF,DAN[I,J]);
               J:=J+1;
             END;
           READLN(INF);
           I:=I+1;
           END;
         CLOSE(INF);
         LEN:=2;
           REWRITE(OUTF,NAME,'INS',LEN);
WRITE(OUTF,CHR(33B),CHR(45B),
            CHR(41B),CHR(61B),CHR(33B),
           CHR(120B),CHR(61B),CHR(73B),
                     CHR(61B),CHR(174B));
         FOR I=1 TO 15 DO
           BEGIN
```

```
            IF I<10 THEN WRITE(OUTF,
                  CHR(ORD('0')+I)) ELSE
                        WRITE(OUTF,
               '1',CHR(ORD('0')+I-10));
            WRITE(OUTF,'/');
            J:=22;
            WHILE DAN[I+3,J]#'_' DO
              BEGIN
                HEX(DAN[I+3,J],CH,CH1);

                WRITE(OUTF,CH,CH1);
                J:=J+1;
              END;
            IF DAN[I+3,12]='a' THEN
                        WRITE(OUTF,'0D');
            WRITE(OUTF,';');
          END;
      WRITE(OUTF,CHR(33B),CHR(57B),
            CHR(33B),CHR(45B),CHR(41B),
                                CHR(63B));
        CLOSE(OUTF);
      END;
END.
```

```
      Keys functions:
Number      Status        Value
======      ======  ====================
K1           auto   DIR/FU/BL/VOL MZ0:
K2           auto   DIR/FU/BL/VOL MZ1:
K3           auto   DEASS
K4           auto   R PASCAL
K5                  MACRO
K6           auto   TY SY:PAS.HLP
K7           auto   ASS TT LP
K8           auto   ASS MZ: DK
K9           auto   ASS MZ1: DK
K10          auto   ,SY:PASCAL,SY:GRPAS
K11                 PRINT
K12                 R TESTMZ
K13                 LINK
K14          auto   @SY:NED
K15          auto   R NPL
==========================================
   *V02 09.90
```

Справки по данной тематике можно получить по адресу: 644005, Омск, ул. Добролюбова, 15, индустриально-педагогический техникум, *Ю. Д. Плетневу*. Тел. 41-04-17.

Знакомьтесь: «Немига»

В летописях это название впервые встречается в 1097 г., когда в том месте, где река Немига впадает в Свислочь, в кровопролитном сражении сыны земли русской отстояли несколько холмов и основали город Менеск, который впоследствии стал называться Минском. Некогда полноводная Немига сейчас закована в бетонную трубу и только в центре Минска, в Троицком предместье, появляется на свет слабым ручейком.

Недалеко от этого места появился еще один «ручеек», постепенно набирающий силу. КУВТ «Немига» не просто еще один компьютерный класс, созданный специально для школ. Основу КУВТа составляет микро-ЭВМ, разработанная для цифровых осциллографов и анализаторов спектра. Таким образом, впервые учебная и производственная техника соединились в одном лице. Сконструировали «Немигу» в Минском радиотехническом институте на кафедре ЭВМ под руководством доктора технических наук А. А. Петровского, а выпускает его Минское производственное объединение «Калибр».

Вычислительный комплекс «Немига» включает в себя рабочее место преподавателя — ПК588 и до 15 рабочих мест учащихся — ПК588-01, объединенных в локальную вычислительную сеть. Основой всех рабочих мест является системный блок СБ588, который включает модули процессора, ОЗУ, локальной сети с таймером, а также клавиатуру и дисплей МС6105. РМП включает еще один блок, содержащий накопители на гибких магнитных дисках с контроллером и контроллер печатающего устройства. Благодаря этому любое РМУ может быть преобразовано в РМП путем подключения блока накопителей и изменения номера рабочего места в локальной сети, на что уходит не более 2—3 мин.

Элементной базой КУВТ «Немига» является микропроцессорный комплект БИС К588, что и определило архитектуру комплекса, совместимую с известной американской мини-ЭВМ PDP-11. Другое следствие применения этого комплекта — чрезвычайно низкое энергопотребление: 500 Вт на класс из 13 рабочих мест.

Системный блок содержит 16-разрядный процессор (соответствует LSI-11/2 или «Электроника-60») с быстродействием 500 тыс. регистровых операций в секунду. Команды расширенной арифметики выполняются с помощью аппаратного умножителя 16×16 с быстродействием 100 тыс. операций в секунду.

Оперативная память объемом 128К байт выполнена на 16 микросхемах типа 565РУ5 и распределена между системной областью, ОЗУ экрана и виртуальным диском. Бейсик-компилятор расположен в ПЗУ объемом 32К байт.

Локальная сеть — двухпроводная линия со скоростью передачи 600К бод — выполнена в соответствии со стандартом MIL/STD-1553b. Программное обеспечение позволяет рассматривать виртуальные диски, экраны или ОЗУ РМУ как устройства, подключенные к РМП, и, следовательно, копировать их содержимое на РМП практически мгновенно и незаметно для учащегося. Виртуальные диски всех РМУ можно программно объединить в один виртуальный диск объемом до 416К байт для РМП. Учащийся также может послать сообщение на РМП, чтобы преподаватель обратил на него внимание.

В системный блок встроено звукоформирующее устройство, позволяющее воспроизводить мелодии в диапазоне 7 октав с 4 уровнями громкости для каждого звука.

В качестве базовой операционной системы в РМП используется адаптированная ОС RT-11. С точки зрения преподавателя КУВТ выглядит как ПЭВМ, к которой в качестве периферийных устройств подключены РМУ. С этими устройствами можно выполнять такие же действия, как, например, с накопителями на магнитных дисках. Можно скопировать на любое РМУ файл, можно прочитать с любого РМУ файл, можно скопировать на РМП состояние любого РМУ.

В то же время для учащегося РМУ представляется инструментом, возможности которого определяются программным обеспечением, переданным сюда преподавателем. Так, один учащийся может изучать иностранный язык с помощью обучающей программы, другой — моделировать физические процессы, работая с Бейсиком, третий — сочинять музыку, используя встроенный формирователь звука.

Но все-таки зачем создан еще один КУВТ? Ведь и локальная сеть, и музыка, и графика — не новинки.

Во-первых, низкая потребляемая мощность предопределяет высокую надежность и практически бесшумную ввиду отсутствия вентиляторов работу. Высокую надежность также обеспечивают архитектура комплекса, позволяющая быстро преобразовывать один тип рабочего места в другой, и конструкция, позволяющая легко заменять платы в модуле.

Во-вторых, в архитектуру комплекса заложена структурная расширяемость при сохранении совместимости с архитектурой PDP-11. Так, например, ОЗУ системного блока без изменения конструкции можно увеличить до 512К байт.

В-третьих, разрабатывали и изготовляют КУВТ приборостроители. Они обеспечили удобную возможность комплектации «Немиги» измерительными и исполнительными приборами. Одинаковые корпуса модулей и дополнительных приборов делают комплекс удобным в обращении, единое программное обеспечение закладывает в школе необходимые в будущем производственные навыки.

Заложены в КУВТ и идеи «на перспективу». В частности, концепция архитектуры позволяет расширять сферу действия КУВТ, например объединив несколько классов в систему через файл-сервер.

Уже имеющееся программное обеспечение позволяет использовать «Немигу» не только на уроках информатики, но и на уроках иностранного языка, математики, физики, электротехники, химии.

Немалую помощь в освоении КУВТ оказывает пользователям Минское ПО «Калибр», систематически проводя семинары на заводе-изготовителе.

Разработчики программных средств также не отстают. Кроме постоянного участия в семинарах они не отказываются консультировать всех желающих. Нелишне заметить, что системные программные средства распространяются бесплатно.

Пожелаем же доброго пути новому участнику сложного процесса компьютеризации страны.

А. АРХАНГЕЛЬСКИЙ

Справки о семинарах на заводе-изготовителе
можно получить по адресу:

220815, Минск, ул. Фабрициуса, 8. Тел. 25-34-08, 25-48-96.

Консультации по программным средствам
можно получить по адресу:

220600, Минск, ул. П. Бровки, 6.
Тел. 39-89-66, 39-89-78 (А. А. Петровский, В. А. Мухаметов).

ЧТО? МОЖЕТ? ЭВМ

Что покупают?

В общем объеме продаж программного обеспечения в Японии игровые программы занимают наибольшую долю: 30 %. На втором месте — учебные программы (20 %), затем идут обслуживающие (17 %), деловые (13 %) и научные программы (10 %). Программы обработки текстов, хоть и являются обслуживающими, выделены в особую графу — их продажа достаточно велика: 6 %. Оставшаяся доля рынка обозначена словом «прочие».

Мышиные новости

Фирма Microgram Computers вывела «бесхвостую» мышь: для передачи сигнала к компьютеру в ней используется не кабель, а инфракрасное излучение (аналогично системе дистанционного управления, например в телевизорах).

Изделие Logitech Mouse Series 9 фирмы Logitech подключается к компьютеру традиционным кабелем, но зато его разрешающая способность (чувствительность к перемещениям) регулируется в диапазоне 50—19200 точек на дюйм. Это не расчет на Левшу, способного сдвинуть мышь на 0,0013 мм, а забота о тех, кто работает за экранами с разрешением более чем 1000×1000 точек.

Как работают у них?

В 1987 г. 24 % объявлений о найме в ФРГ касались специалистов в области компьютеров, в 1989 г. доля таких объявлений увеличилась до 31 %. Здесь имеется в виду спрос на специалистов по наладке и пуску промышленных установок, сооружению и обслуживанию банков данных и компьютерных сетей, системных аналитиков, программистов, операторов и администраторов информационных банков.

Почти в половине объявлений о найме речь идет о так называемых пограничных профессиях (в основном это конторские служащие, умеющие работать на ПЭВМ).

Наибольшее число рабочих мест с использованием компьютеров — на высокоавтоматизированных промышленных предприятиях; на втором месте — сфера обслуживания: туристские и рекламные агентства, медицина, адвокатура, транспортные предприятия. Фирмы, разрабатывающие программное обеспечение и консультирующие предпринимателей, в 1989 г. откатились на третье место.

Рост спроса на «компьютеризованных» тружеников для государственной службы превышает показатели частного бизнеса: бизнесмены раньше поняли выгоды информатизации и уже удовлетворили в основном свои потребности.

А в целом около 43 % имеющихся сейчас в ФРГ рабочих мест требуют тех или иных знаний в области информатики. 21 % из них связан с работой на больших ЭВМ, 61 % — на ЭВМ среднего класса, 18 % — на ПЭВМ. Наиболее часто используемые программные продукты — MS Word (31 %), dBASE (22 %), Lotus 1-2-3 (15 %), MS Windows (9,5 %). Из языков программирования вне конкуренции Си (27 %) и малоизвестный в СССР, несмотря на мафусаилов возраст, Кобол (25 %). Набирает популярность язык PRG (10 %). Кстати, если вы хотите работать программистом, от вас потребуют знания («без словаря»!) минимум двух языков программирования, одной операционной системы, одной СУБД.

В США ПЭВМ оказали на рынок рабочей силы еще более яркое воздействие. Уже более чем в 100 американских компаниях служащие работают на дому, используя для общения с начальством и сослуживцами, а также для передачи результатов своих трудов компьютеры, подключенные к сетям или снабженные модемами. Число таких служащих превышает 3 млн. человек.

Не глядите свысока!

К ПЭВМ «Ямаха», не говоря уж о каком-нибудь игровом «Спектурме», относятся порой без должного уважения. Всего лишь восьмиразрядный процессор Z80, созданный чуть не десять лет назад,— это лишь игрушка. То ли дело IBM PS/2 на процессоре i386!

Конечно, i386 хорошо, но и Z80 неплохо — нужно только знать, где и как его использовать. Создатели рабочей станции (мощный компьютер для профессиональной работы) Minstrel 386SX не пренебрегли продукцией фирмы Intel, но не забыли и скромный восьмиразрядный процессор. Minstrel 386SX выделяется своим сверхоперативным контроллером твердого диска, позволяющим использовать диски со сравнительно большим временем доступа без ухудшения производительности компьютера. Достигнуто это благодаря использованию кэш-памяти, управляемой Z80. Контроллер получился более дешевым, чем традиционный, на базе процессора 68000 фирмы Motorolla, и ускоряет доступ к диску для большинства программных средств в 10 раз.

Компьютер в осциллографе

О телевидении высокой четкости (ТВЧ) многие, и не только в нашей стране, знают только понаслышке, а японская фирма Sony Tektronix уже продает специальные осциллографы для ремонта такой аппаратуры. Портативный цифровой прибор позволяет зафиксировать в своей памяти любую часть видеосигнала и подробно, с точностью до пиксела, рассмотреть его.

Телевидение высокой четкости весьма молодо и не имеет пока единого стандарта, поэтому фирма, создавая компьютеризированный осциллограф, вложила в него способность распознавать кодировку по крайней мере трех систем ТВЧ. Новый прибор не смущает и обилие способов кодировки цветного видеосигнала — он распознает и адаптируется к любому из трех основных его видов: NTSC, PAL и SECAM.

Ю. ЗАЛЬЦМАН

Архитектура БК

Архитектура БК

Центральный процессор

Теперь пришло время более подробно рассмотреть главную часть ЭВМ — ЦП. Как уже говорилось, это универсальное логическое устройство для операций с числами. Но это не все. Процессор имеет свою собственную память, так называемые *регистры общего назначения* (РОН). Этих регистров восемь, и они, как и ячейки памяти и системные регистры, имеют по 16 разрядов; туда могут быть записаны или оттуда прочитаны числа. Но в остальном регистры ЦП существенно отличаются от ячеек памяти. Прежде всего они не входят в общее адресное пространство ЭВМ, и обращения к ним — особые, с применением специальных команд. Кроме того, операции с числами в регистрах ЦП выполняются намного быстрее, чем с числами в памяти. И наконец, целый ряд команд ЭВМ может быть выполнен только с использованием регистров и никак иначе. Таким образом, это как бы отдельная специфическая память ЭВМ — «сверхоперативная память», имеющая большие преимущества перед ОЗУ. Жаль только, что регистров этих так мало!

Название «регистры общего назначения» не совсем точно. Действительно, шесть из них вполне могут быть так названы, они применяются для любых целей, их «имена» — R0, R1, R2, R3, R4, R5. Но два других регистра применяются для специальных целей, их использование наравне с остальными крайне ограничено или исключено вовсе. Что это за регистры?

R6, или, как его обозначают на языке ассемблера, SP — stack pointer, *указатель стека*. Мы уже говорили, что *стек* — это специально выделенная область памяти. R6, или SP, всегда содержит адрес ее начала или, как говорят, *вершины стека*. Пользуясь этим регистром, можно очень удобно записывать в стек или извлекать из него информацию, причем при этом адрес вершины автоматически может меняться, все время указывая ту ячейку памяти, с которой в данный момент мы работаем. Но главное назначение стека — не обслуживать нужды программиста, хотя и это немаловажно, а временно сохранять необходимые самой ЭВМ данные, например, при переходе к подпрограмме или при обработке *прерывания*.

R7, или, на языка ассемблера, PC — program counter, *программный счетчик*. Мы уже говорили, что он всегда указывает адрес очередной команды и тем самым позволяет процессору исполнять программу в правильной последовательности, а изменяя его содержимое, можно изменять и естественный порядок исполнения команд программы. По сути смысл его тот же, что и SP,— он указывает определенный адрес памяти, но в отличие от стека это не память данных, а память программ.

Есть в нашей ЭВМ и еще один регистр, совсем уже специфический. Он носит сокращенное название PS (от слов processor status), иначе его называют *словом состояния процессора* (ССП). Важность этого регистра трудно переоценить — только с его помощью ЭВМ может выполнять, например, такие необходимые действия, как проверку условий и условные переходы, а также делать многое другое, влияющее на вычислительный процесс. Рассмотрим его поэтому подробнее.

В БК-0010 используется только младший байт PS. Все его разряды доступны по записи и чтению, хотя запись в отдельные разряды возможна только с помощью специальных команд или приемов, а чтение из некоторых разрядов происходит вовсе не обязательно в том виде, к которому мы привыкли,— ЦП читает эти разряды и использует их сам, с помощью специальных команд.

Что же это за разряды? Вначале идут четыре *флага*, или *разряда, условий*. Они принимают то или иное значение в зависимости от результата исполнения процессором очередной команды. «Исходное» состоя-

Продолжение. Начало см.: Информатика и образование. 1990. № 4—6; 1991. № 1.

ние этих разрядов, если указанные ниже условия не выполняются,— нули.

00 (разряд C) — *перенос*. Устанавливается в 1, если в результате исполнения команды возник перенос за пределы слова (или байта при работе с байтом). Тут, очевидно, надо оговорить, что перенос может быть только единицы. Есть ли перенос нуля или вовсе нет переноса — это все равно, ноль есть ноль, он на результат не влияет. Возможно, правильнее поэтому было бы сказать, что C-разряд принимает значение разряда переноса.

01 (разряд V) — *переполнение*. Устанавливается в 1, если в результате исполнения команды имело место переполнение, т. е. перенос в знаковый разряд слова (или байта при работе с байтом).

02 (разряд Z) — *ноль*. Устанавливается в 1, если результатом исполнения команды является нулевое содержимое слова (или байта при работе с байтом).

03 (разряд N) — *отрицательность*. Устанавливается в 1, если результатом исполнения команды является отрицательное число в слове (или в байте при работе с байтом).

Используя содержимое этих разрядов, можно организовать ветвление программы, т. е. переходы по условию или, что то же самое, по результатам исполнения команды. Помимо того что эти разряды устанавливаются в 0 или 1 автоматически по результатам исполнения каждой команды, есть еще специальные команды для их установки.

Следующий разряд — 04, или T-разряд. Если его содержимое равно 1, то программа останавливается каждый раз после выполнения очередной команды: происходит так называемое *прерывание по T-разряду*. Этот особый режим процессора широко используется в специальных программах-отладчиках, позволяющих исполнять программы по одной команде с целью их отладки. Записать в T-разряд 1 не так просто, для этого существует специальный прием, с которым мы в свое время познакомимся.

И наконец, последние три разряда — 05, 06, 07 — так называемые *разряды приоритета процессора*. Что это такое?

В процессе работы ЭВМ ЦП преимущественно обрабатывает информацию, относящуюся к основной задаче (скажем, вычисление таблицы синусов). Но с ЭВМ работают и внешние устройства, например клавиатура, которой также нужно уделять внимание. Организовать работу с клавиатурой можно двумя способами.

По первому ЦП после выполнения определенного числа команд прерывает решение задачи и, обратившись к клавиатуре, опрашивает ее. Если ни одна клавиша не была нажата, продолжаются вычисления, затем снова происходит опрос клавиатуры и т. д. Этот способ нерационален — независимо от того, нажата или нет клавиша, ЦП вынужден периодически «отвлекаться» и тратить время на опрос клавиатуры. А если учесть, что на нажатие клавиши ЭВМ должна реагировать мгновенно (чтобы не создавать неудобств в работе), то понятно, что клавиши должны опрашиваться никак не реже, чем 100 или даже 1000 раз в секунду.

Второй способ куда лучше: ЦП решает задачу, а если будет нажата клавиша, то он получит так называемый *запрос на прерывание*. Это специальный сигнал, говорящий процессору, что с какого-то из внешних устройств поступила порция информации и оно ждет, чтобы ЦП отреагировал. ЦП прервет решение задачи, *обработает прерывание* и продолжит вычисления.

Но есть ведь такие задачи, когда ЦП никак не может «отвлечься», например работа с магнитофоном. Лента не будет стоять и ждать, пока ЦП обработает прерывание... Вот здесь и пригодится приоритет.

Если приоритет, установленный на данный момент для ЦП (как говорят, *приоритет текущей задачи*) ниже, чем установленный *приоритет запроса на прерывание*, то ЦП прервет работу и обработает прерывание. Если же приоритет текущей задачи равен или выше приоритета запроса, то ЦП сначала закончит вычисления, а уже потом обработает прерывание. Сразу поясним, что приоритет запросов на прерывание от внешних устройств (у БК таких устройств, по сути, может быть всего два — клавиатура и внешний таймер) установлен для данной конфигурации ЭВМ раз и навсегда. А вот приоритет ЦП может меняться; он определяется содержимым указанных трех разрядов PS.

Сколько может быть задано различных уровней приоритета процессора? Очевидно, восемь, ведь в три двоичных разряда можно записать восемь различных комбинаций. Минимальный приоритет, когда прервать работу процессора может любое устройство — 000, максимальный, когда прервать работу процессора нельзя ничем, кроме клавиши **СТОП** (это — «внеприоритетное» внешнее устройство),— 111.

Продолжение следует.

Операторы управления программой

Эта группа операторов предназначена для организации ветвления, подпрограмм и циклов, т. е. для управления ходом вычислительного процесса путем изменения естественного порядка выполнения операторов.

25. BR MET — ветвление безусловное. Вызывает переход (передачу управления) по адресу MET независимо от битов условий. Метка, по которой выполняется переход, может быть как обычной, так и локальной.

Во всех командах ветвления адрес передачи управления определяется суммой текущего значения счетчика команд PC и младшего байта команды, в который при трансляции заносится смещение — разность # MET — PC (в дополнительном коде, деленная на 2). При исполнении смещение умножается на 2 с учетом знака и, таким образом, обеспечивает передачу управления на расстояние до 256*д* байт от текущего адреса команды ветвления «вперед» или «назад».

Все команды ветвления, кроме BR, являются условными, т. е. передача управления по адресу MET осуществляется, если биты условий, установившиеся после выполнения предшествующей команды в разрядах 00—03 ССП, удовлетворяют условиям оператора ветвления; в противном случае управление передается команде, следующей за командой ветвления. Сами по себе операторы ветвления биты условий не меняют, поэтому можно располагать их друг за другом для последовательной проверки нескольких условий. Удобно описать операторы ветвления в таблице.

№ п/п	Форма записи команды	Ветвление, если результат операции:	Биты условий
26	BNE MET	не равен нулю	Z=0
27	BEQ MET	равен нулю	Z=1
28	BPL MET	плюс	N=0
29	BMI MET	минус	N=1
30	BVC MET	нет переполнения	V=0
31	BVS MET	есть переполнение	V=1
32	BCC MET	нет переноса	C=0
33	BCS MET	есть перенос	C=1
		С учетом знака	
34	BGT MET	больше нуля	N=V и Z=0
35	BLT MET	меньше нуля	N+V=1
36	BGE MET	больше или равен нулю	N=V
37	BLE MET	меньше или равен нулю	N+V=1 или Z=1
		Без учета знака	
38	BHI MET	больше нуля	C=0 и Z=0
39	BLO MET	меньше нуля	C=1
40	BHIS MET	больше или равен нулю	C=0
41	BLOS MET	меньше или равен нулю	C=1 или Z=1

Первые восемь операторов в особых пояснениях не нуждаются — они проверяют отдельные биты условий. Операторы BLO и BHIS тоже проверяют отдельные биты и полностью соответствуют операторам BCS и BCC соответственно. Эти дополнительные наименования операторов введены исключительно для удобств программистов. А вот различие в действии операторов переходов с учетом и без учета знака нуждается в пояснении.

Как известно, числа со знаком изображаются в системе команд нашей ЭВМ в дополнительном коде, т. е. старший бит числа является *знаковым* — ноль, если число положительное, и единица, если отрицательное. Диапазон абсолютных величин чисел со знаком, естественно, меньше из-за уменьшения разрядности на 1. Поэтому используется и второе представление — без знака, только положительных чисел, но с большей разрядностью. В связи с этим при сравнении чисел с разным представлением возникают несуразицы. Например, любое число в ряду чисел без знака больше нуля, среди же имеющих знак больше нуля только положительные числа, а отрицательные — меньше. Это утверждение кажется банальным, но при использовании операторов ветвления форму представления чисел обязательно надо учитывать. Как правило, при логических операциях с числами (сравнение кодов, сортировка и т. п.) удобнее рассматривать их как числа без знака, а при арифметических вычислениях — как имеющие знак. Вообще же правильный выбор команд ветвления — довольно сложная задача, и некоторые авторы рекомендуют сводить сравнение чисел к таким случаям, когда становится возможным применение лишь двух операторов — BNE и BEQ, всегда действующих однозначно. Относясь к таким рекомендациям сочувственно, нужно все-таки отдавать себе отчет, что далеко не все задачи сводимы к частным вариантам, а цена за такое преобразование — потеря памяти и быстродействия. Поэтому следует применять все операторы ветвления, без вычетов, но тогда, когда это необходимо. А чтобы не делать ошибок, нужно ясно представлять, какой результат возможен при той или иной операции, при тех или иных исходных данных, а не применять операторы бездумно, по шаблону.

В заключение приведем пример простейшей программы с использованием операторов ветвления:

```
; Программа мигания экрана
0:      MOV     #177777,R0      Все единицы
1:      MOV     #40000,R1       Начало экрана
2:      MOV     R0,(R1)+        Запись оче-
        ;                       редного слова
        ;                       в ОЗУ
        CMP     #100000,R1      Конец экрана?
        BNE     2               Нет-продолжать
        TST     R0              R0=0?
        BEQ     0               Да - записать
        ;                       единицы
        CLR     R0              Иначе обнулить
        BR      1               R0 и войти в
        ;                       цикл очистки
        ;                       экрана
END
```

42. JMP N — безусловный переход. Эта команда, в отличие от безусловного ветвления, передает управление по любому адресу, а не только на 256*д* байт, и допускает задание адреса перехода любым способом (есть, конечно, и исключения — например, невозможна передача управления на регистр (JMP RN), а метки в составе адреса перехода допустимы только обычные).

43. JSR RN, N — переход к подпрограмме. По данной команде процессор прерывает обработку текущей последовательности операторов, записывает в стек содержимое указанного регистра RN (автоматически выполняя команду MOV RN,—(SP)), заносит в регистр содержимое PC (иначе говоря, адрес следующей команды программы), а затем заносит в PC адрес подпрограммы N (способ его задания может быть любым — с теми же ограничениями, что и для оператора JMP), вследствие чего следующей исполняется первая команда подпрограммы. Обычно в конце подпрограммы должна стоять *команда возврата* — RTS RN, где RN — тот же самый регистр, что в операторе JSR. По этой команде процессор переписывает содержимое RN в PC, т. е. восстанавливает адрес следующей за переходом к подпрограмме команды основной программы, затем восстанавливает из стека содержимое RN (автоматически выполняя команду MOV (SP)+,RN), и выполнение основной программы продолжается. Из одной подпрограммы можно обратиться к другой, из другой — к третьей и т. д.; важно лишь, чтобы используемый в качестве буфера адреса возврата регистр (это может быть как один и тот же регистр для всех подпрограмм, так и разные) во время работы подпрограммы не менял свое содержимое (если, конечно, это не делается со специальными целями), иначе правильный возврат из подпрограммы станет невозможен. Число последовательных обращений из одной подпрограммы в другую (так называемый *уровень вложенности*) ограничено допустимой глубиной стека и на БК-0010 может достигать 262*д*.

Важный частный случай применения данного оператора — использование в качестве RN счетчика команд PC. Если проанализировать работу операторов JSR и RTS, то окажется, что по команде JSR PC,N в стеке сохраняется, а по команде RTS PC восстанавливается непосредственно адрес возврата из подпрограммы! Этот способ обращения к подпрограммам очень удобен тем, что позволяет не занимать под адрес возврата ни одного операционного регистра, ведь их всего шесть. Понятно, что данный способ обращения к подпрограммам используется чаще других и для него в ассемблере МИКРО.10К даже предусмотрена специальная форма записи: JSR RC,N—CALL N, вместо RTS PC—RET.

Наличие такой формы отнюдь не запрещает стандартного обращения, просто такое короче и удобнее.

Выходов из подпрограммы (как и входов) может быть несколько. Всегда нужно помнить, что при входе в подпрограмму указатель стека SP уменьшается на 2, и при выходе из нее, если он осуществляется не оператором RTS (RET), а иначе (например, BR, JMP и т. п.), стек необходимо восстанавливать, иначе он рано или поздно будет исчерпан и ЭВМ перестанет работать. В подпрограмме же со стеком следует работать осторожно и заботиться, чтобы при выходе из нее адрес возврата находился на вершине стека.

44. RTS RN — возврат из подпрограммы. Подробно рассмотрен выше. В МИКРО.10К вместо RTS PC может использоваться оператор RET.

45. MARK X — пометить стек. Эта команда применяется крайне редко, может быть потому, что нигде она не описана достаточно понятно. Попробуем восполнить этот пробел.

Бывают случаи, когда при обращении к подпрограмме нужно передать ей ряд *параметров*, попросту говоря, чисел. Эти числа при разных обращениях к одной и той же подпрограмме могут быть разными и их может быть разное количество. В таких случаях довольно удобно перед обращением заносить параметры в стек, извлечь же их в подпрограмме можно, например, с помощью индексной адресации или иным путем. Но так как параметров может быть разное количество, то каждый раз по выходе из подпрограммы стек надо восстанавливать. Команда MARK делает это автоматически. Как ее применять?

Сначала выберем один из регистров RN под буфер адреса возврата и запишем его содержимое в стек. Затем занесем в стек X параметров. После этого запишем в стек саму команду в форме MARK X, где X — восьмеричное число параметров (от 0 до 77). Нако-

нец, запишем адрес вершины стека в RN и обратимся к подпрограмме через другой регистр. Если в конце подпрограммы стоит команда RTS RN, то стек и регистры автоматически будут восстановлены. Не вдаваясь в дальнейшие подробности, просто приведем пример. Пусть регистром возврата выбран R5, а обращение к подпрограмме будет через регистр PC. Занесем, например, в стек три параметра:

```
        MOV    R5,-(SP)      Сохранить R5
        MOV    #101,-(SP)    Занести
        MOV    #102,-(SP)    параметры
        MOV    #103,-(SP)    1, 2, 3
        MOV    MR3,-(SP)     Занести код
        ;                    команды
        ;                    "MARK 3"
        MOV    SP,R5         Занести адрес
        ;                    возврата
        CALL   SBR           Перейти к под-
        ;                    программе SBR
        ;.....               Дальнейший
        ;                    текст программы
SBR:    ;.....               Текст под-
        ;                    программы SBR
        RTS    R5            Выход из под-
        ;                    программы с
        ;                    восстановле-
        ;                    нием стека
MR3:    MARK   3             Код команды
                             "MARK 3"
```

Читатель, программирующий на ассемблере, может, конечно, предложить массу других способов как передачи параметров подпрограмме, так и восстановления стека без команды MARK. Но поскольку данная команда существует, разобрать ее было необходимо хотя бы для того, чтобы избежать вопросов. Эту команду можно также использовать для модификации регистра SP, для чего просто впишите в текст программы MARK X; после ее выполнения содержимое SP возрастет на 2X+2.

46. SOB RN,MET — вычитание единицы и ветвление. Этот оператор может быть использован для организации циклов. При его исполнении происходит сначала вычитание единицы из содержимого регистра RN, а затем, если результат не ноль, переход к метке MET; если же содержимое RN стало равно нулю, то программа выполняется дальше. Метка MET может быть как локальной, так и обычной и должна быть расположена до оператора SOB, т. е. передача управления возможна только «назад» и не более, чем на 64$_{10}$ слова. Это объясняется тем, что в операторе SOB для хранения смещения используются только младшие 6 битов, а при исполнении перехода смещение (без знака) умножается на 2 и вычитается из PC. Циклы могут быть и вложенными, но число вложений, если не принимать специальных мер, не может превышать числа операционных регистров, которые используются для хранения переменных циклов. Помимо организации вычислений этот оператор часто используется для создания временных задержек при работе программ, причем «пустой» цикл SOB при исходном содержимом регистра, равном нулю, выполняется примерно за 0,4 с. Если требуются бóльшие задержки, следует прибегнуть к вложенным циклам или ввести в цикл «пустые» команды, например CMP R4, R4. Приведем пример использования данного оператора: уже знакомую программу мигания экрана, но решенную иными средствами.

```
; Программа мигания экрана
Ø:      MOV    #177777,RØ    Все единицы
1:      MOV    #40000,R1     Начало экрана
        MOV    #20000,R2     Цикл из 20000
        ;                    повторов
2:      MOV    RØ,(R1)+      Запись очеред-
        ;                    ного слова в
        ;                    ОЗУ
        SOB    R2,2          Если не конец -
        ;                    продолжать
        BEQ    Ø             Если Ø - запи-
        ;                    сать все
        ;                    единицы
        CLR    RØ            Иначе обнулить
        BR     1             RØ и войти в
        ;                    цикл очистки
        ;                    экрана
END
```

Сравните эту программу с приведенной ранее (где цикл был организован с помощью оператора CMP). Какая из них короче и какая работает быстрее? Обратите внимание, что оператор SOB не меняет биты условий, поэтому нам удалось избавиться от проверки содержимого R0 оператором TST.

При работе с циклами нужно быть внимательным, чтобы не передать управление на оператор, задающий начальное значение переменной цикла, иначе цикл станет бесконечным. С этой же целью нужно заботиться о сохранении в цикле содержимого RN.

Операторы прерывания программы

Эта группа подробно описывается в разделе, посвященном прерываниям, и здесь мы лишь кратко перечислим соответствующие команды.

47. EMT X — командное прерывание по вектору 30. Предназначено преимущественно для обслуживания системных программ. В младшем байте команды передается дополнительная информация X (номер EMT, восьмеричное число от 0 до 377). На БК-0010 предусмотрены следующие прерывания:

EMT 4 — инициализация драйвера клавиатуры;

EMT 6 — ввод кода символа с клавиатуры;

EMT 10 — ввод строки символов с клавиатуры;

EMT 12 — установка ключей клавиатуры;

EMT 14 — инициализация всех драйверов;
EMT 16 — передача кодов драйверу дисплея;
EMT 20 — вывод строки символов на экран;
EMT 22 — запись символа в служебную строку;
EMT 24 — установка курсора по координатам;
EMT 26 — чтение координат курсора;
EMT 30 — формирование точки по координатам;
EMT 32 — формирование вектора;
EMT 34 — чтение слова состояния дисплея;
EMT 36 — работа с магнитофоном;
EMT 40—50 — работа с ТЛГ-каналом;
EMT 52—110 — резервные входы EMT.

Используя набор EMT, можно выполнять самые разнообразные действия: вводить и выводить символы и графику, читать с магнитофона файлы и выводить массивы на МЛ и т. п. Это как бы уже имеющийся на БК-0010 пакет подпрограмм на все случаи жизни. Оператор EMT активно используется самим БК-0010, поэтому менять вектор 30 (или добавлять новые EMT) нужно с осторожностью.

48. TRAP X — командное прерывание по вектору 34. X — номер TRAP, восьмеричное число от 0 до 377. Предназначено в основном для программ пользования. Пользователь может написать программу TRAP-диспетчера и создать свой пакет программ обработки прерываний TRAP, аналогичный EMT-пакету, но решающий другие задачи. Воспользуемся случаем и приведем пример простейшей программы TRAP-диспетчера. Предположим, мы хотим использовать прерывание TRAP для такой банальной цели, как вывод на экран символов и исполнение команд, код которых соответствует номеру TRAP. Вспомним, как выполняется командное прерывание. Обнаружив в программе команду прерывания TRAP, процессор выполняет действия, которые можно условно изобразить следующей программой:

```
MFPS   -(SP)        ; Сохранить
;                   ; текущее ССП
MOV    PC,-(SP)     ; Сохранить адрес
;                   ; следующей
;                   ; команды
MTPS   @#36         ; Записать новое
;                   ; ССП
MOV    @#34,PC      ; Записать адрес
;                   ; программы
;                   ; обработки
;                   ; прерывания
```

Конечно, такой программы нам писать не нужно, процессор делает это сам, программа обработки прерываний заложена в него на аппаратном уровне (если точнее — на так называемом микропрограммном, но для нас это безразлично). Итак, процессор перешел к программе обработки прерывания. Что она будет собой представлять для нашего случая? Раз мы хотим получить вывод символов, код которых соответствует номеру TRAP, нам, видимо, следует выделить этот номер. Вспомним, что коды TRAP лежат в диапазоне 104400—104777, т. е. номер TRAP — это просто младший байт команды. А как «выудить» командный код? В стеке у нас сохранен адрес следующей команды. Уменьшив его на 2, получим адрес текущей, т. е. TRAP! А дальше уже просто: косвенное обращение по этому адресу даст код TRAP; получив его, используем его младший байт как код символа — вот и все! Но прежде чем все это делать, мы должны записать вектор 34, соответствующий адресу нашей программы обработки прерывания. Пусть программа обработки TRAP помечена меткой TRP. Запишем:

```
MOV    #TRP,@#34    Записать новый
;                   вектор 34
```

Если бы мы хотели на фоне исполнения TRAP запретить все прерывания от внешних устройств, мы могли бы задать также нужный приоритет процессора во втором слове вектора, например:

```
MOV    #340,@#36    Задать макси-
;                   мальный
;                   приоритет
```

но в данном случае это не принципиально. А теперь напишем саму программу обработки прерывания.

```
; Программа обработки прерывания TRAP
TRP:  MOV    R0,-(SR)      Сохранить R0
      MOV    2(SR),R0      Извлечь адрес
      ;                    следующей
      ;                    команды
      MOV    -(R0),R0      Извлечь текущую
      ;                    команду (TRAP)
      EMT    16            Вывести на
      ;                    экран символ,
      ;                    код которого
      ;                    в младшем
      ;                    байте R0
      MOV    (SP)+,R0      Восстановить R0
      RTI                  Выход из
      ;                    прерывания
```

Как видим, программа получилась очень простой; это диктуется целью, которую мы себе поставили. Если теперь написать, например, последовательность команд

```
TRAP  101
TRAP  102
TRAP  103
```

то на экран будут выведены три символа: ABC. Программа их выдачи на экран занимает всего три слова, тогда как при «классическом» способе вывода (занести код симво-

ла в регистр R0 и выполнить EMT 16) такая программа заняла бы девять слов! Между тем сама программа обработки TRAP занимает у нас семь слов. Если обращаться к ней придется достаточно часто, мы получим существенную экономию памяти. Если задача сохранения и последующего восстановления R0 не ставится, то программа обработки TRAP будет еще короче.

49. IOT — командное прерывание по вектору 20. Предназначена для обслуживания устройств ввода-вывода в операционных системах, в системе БК-0010 не используется. Выполняется как EMT или TRAP с той разницей, что младший байт команды не содержит дополнительной информации (номера). Может быть использована для обращения к какой-либо одной программе.

50. BPT — командное прерывание по вектору 14. Используется в отладчиках, подробно рассмотрена в разделе о прерываниях. Работает так же, как IOT.

51. RTI — возврат из прерывания. Используется для выхода из программы обработки прерываний (любых, кроме отладочных). Восстанавливает стек, ССП и адрес очередной команды. При обработке оператора RTI процессор автоматически выполняет (на микропрограммном уровне) последовательность команд

```
MOV    (SP)+,PC     Восстановить
;                   адрес очеред-
;                   ной команды
;                   прерванной
;                   программы
MTPS   (SP)+        Восстановить
;                   ССП
```

52. RTT — возврат из прерывания. Используется для выхода из отладочного прерывания, когда установлен Т-разряд ССП. Может применяться и для выхода из других прерываний. Отличие RTT от RTI лишь в том, что прерывание по Т-разряду при отработке RTT не наступает, а выполняется следующая команда, и лишь после нее происходит прерывание по Т-разряду.

53. HALT — останов. В сущности, эта команда есть не что иное, как командное прерывание по вектору 4, и дальнейшие действия ЭВМ всецело зависят от того, как построена программа обработки прерывания по этому вектору. Команда (обычно неправильно относимая к командам управления машиной) может быть использована не только для останова, но и для любых целей аналогично прочим командам прерываний, но нужно помнить, что, изменяя вектор 4 и программу обработки прерывания по этому вектору, мы меняем и порядок работы клавиши **СТОП.**

Операторы управления машиной

54. WAIT — ожидание. Приостанавливает выполнение текущей программы до прерывания от внешних устройств (клавиатуры, таймера, клавиши СТОП). Может быть использован для приостанова программы и ожидания инициативы пользователя, но нужно помнить, что прерывания возникают при этом в соответствии с приоритетом внешних устройств и процессора, так что для прерывания, например, от клавиатуры приоритет процессора должен быть менее 4.

55. RESET — сброс внешних устройств. Оператор вызывает выдачу по командной магистрали общей шины сигнала INIT, который сбрасывает все устройства ЭВМ (порт, системные регистры) в исходное состояние. Особенность выполнения данного оператора на БК-0010 такова, что после него необходимо инициализировать драйвер клавиатуры командой EMT 4.

Прочие операторы

56. NOP — нет операции. Используется при отладке, если нужно временно исключить какую-то команду, или для создания небольшой задержки (около 4 мкс). Не делает ничего. Код этой команды — 240.

К данной группе относятся также команды изменения признаков, т. е. разрядов 00—03 ССП. Используются они редко.

57. CLN — очистка N.
58. CLZ — очистка Z.
59. CLV — очистка V.
60. CLC — очистка C.
61. CCC — очистка всех разрядов.
62. SEN — установка N.
63. SEZ — установка Z.
64. SEV — установка V.
65. SEC — установка C.
66. SCC — установка всех разрядов.

Продолжение следует.

А. ДИКОВ, А. КАЛАШНИКОВ, А. КУЛАКОВ

TermOS

Операционная мини-система TermOS с автоматическим управлением накопителем на магнитной ленте разработана с целью максимального облегчения работы пользователей БК-0010(-01) с НМЛ (бытовым магнитофоном). Свое название система получила от «TapE Recorder Machine OS», что буквально можно перевести как «магнитофонная операционная система».

Основным отличием TermOS от других известных программ автоматического управления НМЛ является то, что сама управляющая программа и каталоги кассет загружаются в дополнительное ОЗУ, использующее область адресов 140000—157777 (при подключении такого ОЗУ к БК-0010-01 ПЗУ с Бейсиком автоматически отключается), и остаются там резидентно, т. е. могут быть «испорчены» лишь преднамеренно выключением питания или занесением в эту область новой информации.

TermOS может работать либо с одним каталогом, либо с двумя одновременно, запоминая показания счетчиков лент при переходе от одной кассеты к другой. Благодаря этому для считывания нового файла нет нужды перематывать ленту в начало. А если учесть полностью автоматизированное управление магнитофоном (перемотку влево, вправо, запись и воспроизведение включает непосредственно БК), то становится правомерным сравнение БК с «большими» ЭВМ конца 60-х гг.

Выбор необходимых функций системы осуществляется через меню. Система меню является иерархической и многоуровневой. Ее суть заключается в том, что в верхней строке экрана высвечивается главное меню, каждая функция которого может иметь свое меню (низшего уровня), помещающееся во второй строке. Если у выбранной функции меню низшего уровня нет, то во второй строке помещается объяснение действия выбранной функции. В меню низшего уровня может быть свое меню еще более низкого уровня и т. д. Таким образом, пользователю нет необходимости запоминать все команды системы — они находятся на экране; остается лишь выбрать необходимое, передвигаясь по меню вправо — влево и «в глубину».

Основные функции системы:

загрузка программ по адресу, указанному в оглавлении файла, либо по адресу, указанному пользователем;

загрузка и запуск программ;

подмотка ленты к указанному файлу;

фиктивное чтение имен файлов;

копирование группы файлов в указанной последовательности либо в полностью автоматическом режиме с использованием второго магнитофона (управляемого только по питанию), либо на одном магнитофоне с выдачей подсказок о смене кассет;

распечатка файлов на экране или принтере в символьном виде;

то же для указанного фрагмента ОЗУ;

управление режимами печати принтера (включение и отключение подчеркивания, печати выделенным, черновым, уплотненным шрифтом, шрифтом двойной ширины и высоты, печать графической информации с экрана и т. д.);

переход от каталога одной кассеты к каталогу другой;

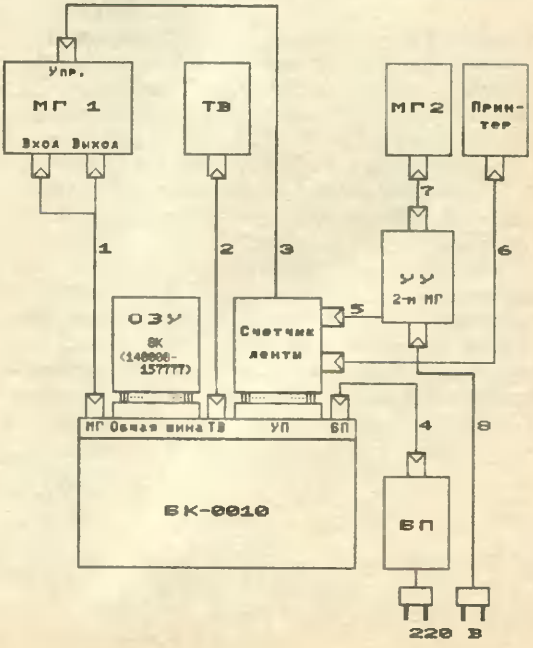
удаление имен файлов из каталога (но не с МЛ);

восстановление удаленных имен файлов в каталоге;

запись обновленного каталога на магнитную ленту;

инициализация системы;

выход в монитор, МСТД, Фокал (возможен выход с последующим дублированием



всей появляющейся на экране информации на принтер и управлением принтером с клавиатуры либо из выполняемой программы).

При любых сбоях счетчика ленты (даже внесенных преднамеренно) управляющая программа замечает ошибку и производит программную коррекцию счетчика, не перематывая ленту в начало.

Если указанную программу считать с ленты не удалось, TermOS делает вторую попытку и, если она окажется неудачной, выдает об этом сообщение, а в каталоге делает отметку, подчеркивая дефектный файл.

Примененная система меню позволяет легко изменить конфигурацию системы — включить дополнительные функции, исключить ненужные. С этим может справиться пользователь, имеющий опыт работы с ассемблером и листинг программы TermOS.

Возможна зашивка программы TermOS в ПЗУ с адреса 120000, т. е. запуск ОС при включении машины, что делает систему еще более удобной (это не исключит возможность работы с Фокал-программами).

TermOS предназначена для работы с комплексом, включающим:

устройство расширения ОЗУ;

специальным образом доработанный магнитофон с электронно-логическим управлением, используемый в качестве НМЛ;

устройство управлением вторым магнитофоном при копировании файлов.

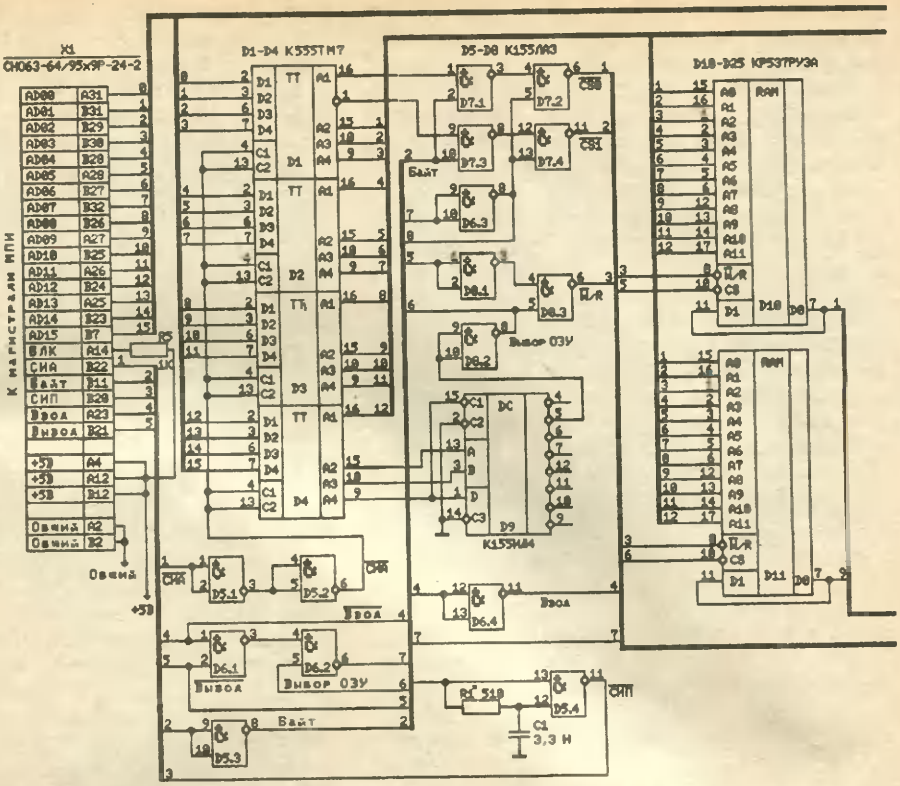
Блок-схема комплекса приведена на рисунке.

Заинтересовавшиеся могут писать по адресу: 123497, Москва, ул. Рогова, 14, корп. 1, кв. 85, *Кулакову Александру Тихоновичу.*

(Продолжение следует.)

В следующих номерах журнала будет рассказано об устройстве расширения ОЗУ и доработке магнитофона.

Энергонезависимое ОЗУ



Энергонезависимое ОЗУ (ЭОЗУ) предназначено для расширения объема ОЗУ пользователя компьютера БК-0010.01. Подключается к магистрали МПИ вместо блока МСТД, при этом БИС ПЗУ языка Бейсик автоматически блокируются.

ЭОЗУ выполнено на микросхемах статической памяти типа КР537РУ3А. Объем памяти — 8К байт, этого достаточно для размещения практически любой системной программы.

ЭОЗУ может работать в любом месте адресного пространства компьютера, но целесообразно размещать его с адреса 120000_8 или 140000_8.

Программы, помещенные в ЭОЗУ, можно подвергать изменению неограниченное число раз. Для сохранения информации при отключении или сбоях питания предусмотрен буферный источник питания

Основные технические характеристики:

напряжение батареи питания — 4 В;

ток, потребляемый от источника +5В компьютера,— 150 мА;

ток, потребляемый от батареи в режиме хранения информации,— 100 мкА.

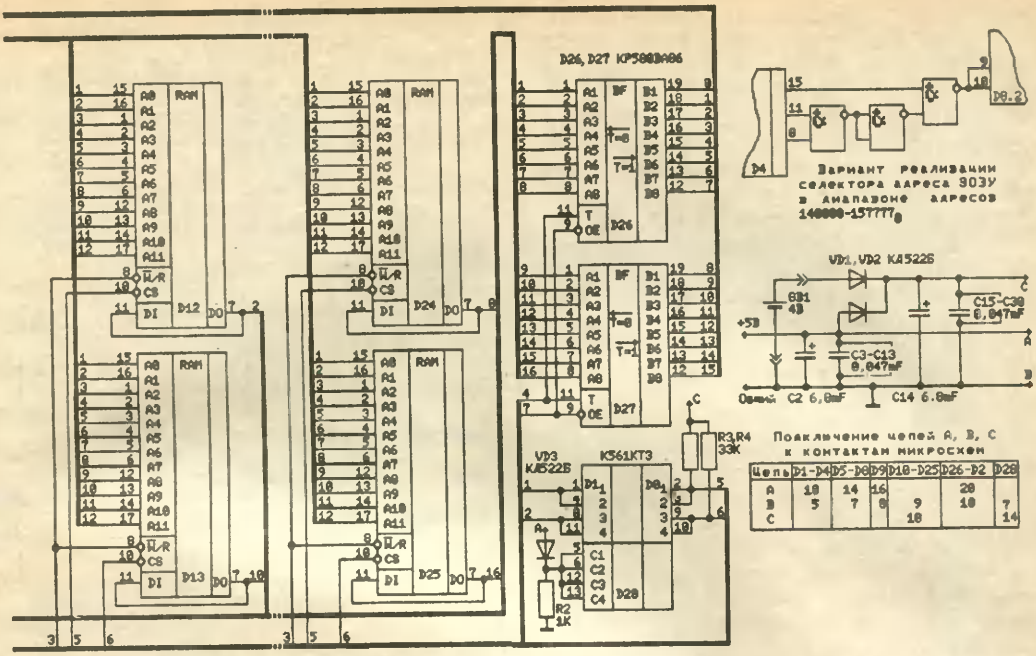
Регистр адреса собран на микросхемах D1 — D4 (К5555ТМ7), их выходы образуют шину адреса ЭОЗУ. Адрес обращения к ОЗУ фиксируется в регистре сигналом СИА. Три старших разряда шины адреса использованы для адресации ЭОЗУ, они подведены к селектору адреса, выполненному на микросхеме D9 (К155ИД4). С помощью перемычек может быть выбрана любая область адресного пространства компьютера с интервалом 20000_8.

Шинные формирователи D26 — D27 (КР580ВА86) образуют шину данных. Направление передачи данных определяется сигналом ВВОД магистрали. Шина данных находится в активном состоянии только в момент непосредственной передачи данных (активен либо сигнал ВВОД, либо сигнал ВЫВОД).

Непосредственно ОЗУ выполнено на 16 микросхемах D10 — D25 (КР537РУ3А). На схеме условно показана только часть микросхем.

Сигнал СИП вырабатывается на выходе элемента D5.4 при активном сигнале ВВОД или ВЫВОД.

Микросхемы D7, D8 вырабатывают сигнал записи-считывания W/R и сигналы выборки кристалла CS0, CS1 для старшего и младшего байтов ОЗУ при выполнении операций чтения-записи байта.



Ключ D28 (К561КТ3) совместно с устройством подключения буферного источника питания GB1 к микросхемам памяти обеспечивают блокировку сигналов выборки кристаллов при сбоях и отключении питания компьютера; при этом микросхемы памяти и ключ оказываются под напряжением GB1, достаточным для сохранения информации.

Изготовление ЭОЗУ может быть выполнено любым доступным пользователю способом. При исправных деталях и правильном монтаже ЭОЗУ не требует наладки.

А. БАРСУКОВ

Просмотрщик памяти

Хороший отладчик кодовых программ — полезнейшая вещь: он позволяет просмотреть и изменить содержимое ОЗУ, произвести пошаговое выполнение программы, установить контрольные точки и т. п. Подобных программ для БК-0010 существует немало. Стоит ли добавлять к ним еще одну, вдобавок имеющую довольно скромные возможности? Решайте сами. Предлагаемая ниже программа может только просматривать и изменять содержимое ОЗУ, а также искать нужное слово или байт, но зато занимает 1600_8 байтов (OTL9, например, занимает 11000_8). К тому же ее можно разместить в той части ОЗУ, которая используется для хранения служебной строки экрана, и, таким образом, не ущемить прикладную программу ни на байт.

При работе программы для каждого слова памяти на экране индицируются его адрес, значение, разбивка на байты в числовом и символьном видах. Предусмотрены следующие режимы работы (в скобках — включающие их клавиши и команды):

следующее слово **(ВНИЗ)**;
предыдущее слово **(ВВЕРХ)**;
установить адрес **(А** ⟨адрес⟩ **ВВОД)**;
изменить слово **(ВВОД** ⟨значение⟩ **ВВОД)**;
изменить байт **(ВС** ⟨значение⟩ **ВВОД)**;
задать модель для поиска **(М** ⟨модель⟩ **ВВОД)**;
искать слово **(I)**;
искать байт **(Q)**;
очистить экран **(СБР)**;
окончить работу **(КТ)**.

Если программа вас заинтересовала — введите ее в БК, дайте команду RUN. Экран очистится. В служебной строке разместится сама программа, а на экране появится содержимое нулевого слова. Проверьте правильность исполнения описанных команд

(для проверки изменения содержимого ОЗУ используйте свободную область с адреса 20000). После окончания проверки нажмите клавишу **КТ** и вернитесь в транслятор Бейсика командой S120170. Убедитесь, что вы не повредили важных областей памяти. Для этого повторно запустите программу и снова вернитесь в Бейсик.

Теперь осталось записать на МЛ кодовый вариант программы. Для этого удалите строку 100 и запустите программу командой RUN; в служебной строке сформируется кодовый вариант программы. Запишите его командой

```
BSAVE "<имя>",&040000,&041600
```

Программа является абсолютно перемещаемой. Ее можно загрузить в любое место памяти с адреса 1000 до адреса 40000. Желаемый адрес загрузки можно задать в команде BLOAD или указать его при создании программы, изменив значение в строке 10 (и в команде BSAVE соответственно). Загрузка программы в ОЗУ пользователя несколько увеличивает скорость вывода информации на экран.

Несколько замечаний об использовании программы в экранном ОЗУ. Скорость вывода можно увеличить, периодически очищая экран. Внимательно вводите числовые значения. Нажатие любой клавиши, выводящей информацию в служебную строку, может повредить программу. Перед каждой новой загрузкой программы не забывайте устанавливать значение регистра рулонного сдвига двукратным нажатием клавиш **АР2+СБР**.

При вводе чисел исправления не допускаются. В случае ошибки ввода нажмите клавишу **СТОП** и повторите ввод. Попытки изменить ПЗУ блокируются.

```
1 ? CHR$(140),CHR$(140)
10 A%=&040000
20 D%=&01600
30 N%=A%
40 B%=A%+D%
50 READ S%
60 POKE N%,S%
70 N%=N%+2
80 IF N%<B% THEN 50
90 DEF USR=A%
100 A=USR(A)
110 END
120 DATA &010767,&01376,&05037,&0220,
                              &012706,&01000
130 DATA &012700,&0233,&0105737,&040,
                              &01401,&0104016
140 DATA &0105737,&056,&01402,&05300,
                              &0104016,&05003
150 DATA &052737,&0100,&0177660,
                              &010700,&062700,&01274
160 DATA &010037,&020,&012700,&014,
                              &0104016,&010700
170 DATA &062700,&020,&010037,4,&0403,
                                        &012700
180 DATA &014,&0104016,&04767,&0536,
                                        &032737,&0100
190 DATA &0177716,&01374,&032737,
                              &0200,&0177660,&01414
200 DATA &013767,&0177662,&01232,
                              &042767,&0177640,&01224
210 DATA &012700,7,&0104016,&016700,
                              &01212,&0402
220 DATA &016700,&01204,&0120027,3,
                              &01013,&012716
230 DATA &0100442,&011637,4,&042737,
                              &0100,&0177660
240 DATA &012700,&0236,&0104016,
                              &0207,&0120027,&033
250 DATA &01004,&04767,&01020,
                              &05723,&0723,&0120027
260 DATA &032,&01006,&05703,&01716,
                              &05743,&04767
270 DATA &0736,&0712,&0120027,&012,
                              &01006,&0112700
280 DATA &076,&0104016,4,&010513,
                              &0764,&0120027
290 DATA &023,&01012,&04167,&0254,
                              &0140742,&0152312
300 DATA &072,&0110513,&012701,1,
                              &074103,&0747
310 DATA &0120027,&055,&01004,
                              &012701,1,&074103
320 DATA &0655,&0120027,&014,&01002,
                              &0104016,&0650
330 DATA &0120027,&0101,&01010,
                              &04167,&0172,&0142341
340 DATA &0142722,&020323,&072,
                              &010503,&0632,&0120027
350 DATA &0115,&01011,&04167,&0144,
                              &0147755,&0142704
360 DATA &0154314,&072,&010567,
                              &0724,&0616,&0120027
370 DATA &0111,&01017,&04167,&0124,
                              &0147760,&0151711
380 DATA &027313,&040,&042767,
                              &0100000,&0124,&042767
390 DATA &0100000,&0120,&04767,
                              &0102,&0427,&0120027
400 DATA &0121,&01261,&04167,&060,
                              &0147760,&0151711
410 DATA &020313,&0140702,&0152312,
                              &027301,&040,&052767
420 DATA &0100000,&052,&052767,
                              &0100000,&046,&05203
430 DATA &04767,&026,&042703,1,
                                        &0167,&0177306
440 DATA &05002,&0104020,4,&0201,
                                        &05002,&0104020
450 DATA &0201,&010305,&020327,
                              &0177500,&0103005,&05723
460 DATA &021367,&0540,&01371,
                              &0207,&010503,&012700
470 DATA 7,&0104016,&0104016,
                              &0104016,&0207,&04437
480 DATA &0110346,&012700,&0232,
                              &0104016,&05000,&010305
490 DATA &04767,&0154,&012700,&040,
                              &0104016,&0104016
500 DATA &012700,&010,&011305,
                              &04767,&0132,&012700
510 DATA &040,&0104016,&0104016,
                              &012700,&020,&011305
520 DATA &04767,&0126,&012700,&056,
                                        &0104016,&012700
530 DATA &024,&011305,&0305,&04767,
                                        &0104,&012700
540 DATA &040.&0104016.&0104016.
```

```
                    &0104026,&012701,&031
550 DATA &0104024,&011300,&04767,
                    &0146,&0300,&04767
560 DATA &0140,&0104026,&012701,
                    &035,&0104024,&012700
570 DATA &0231,&0104016,&012700,
                    &0232,&0104016,&04437
580 DATA &0110362,&0207,&012767,6,
                    &040,&012767
590 DATA 1,&036,&0406,&012767,3,&022
600 DATA &012767,2,&020,&04437,
                    &0110346,&0104026
610 DATA &010001,&0104024,&012703,
                    3,&012702,2
620 DATA &05000,&06305,&06100,
                    &077203,&052700,&060
630 DATA &0104016,&012702,3,
                    &077312,&04437,&0110362
640 DATA &0207,&010046,&042700,
                    &0177400,&020027,&040
650 DATA &02406,&020027,&0200,
                    &0101405,&020027,&0240
660 DATA &0103002,&012700,&040,
                    &0104016,&012600,&0207
670 DATA &026727.&0152.&040000.
                    &02407,&0104026,&05702
680 DATA &01004,&012700,&024,
                    &0104016,&0207,&012700
690 DATA &032,&0104016,&0207,
                    &026727,&0114,&040000
700 DATA &02423,&0104026,&020227,
                    &026,&02417,&05002
710 DATA &0104024,&012700,&0232,
                    &0104016,&012700,&023
720 DATA &0104016,&012702,&026,
                    &0104024,&012700,&0232
730 DATA &0104016,&0207,&012700,
                    &033,&0104016,&0207
740 DATA &042737,&0100,&0177660,
                    &04737,&0100472,&052737
750 DATA &0100,&0177660,2,0,0,0,0,
                    0,0,0,0,0,0,0,0,0,0,0,0
760 DATA 0,0,0,0,0,0,0,0,0,0,0,0,0,0,
                    0,0,0,0,0,0,0,0,0,0,0,0
770 DATA 0,0,0,0,0,0,0,0,0,0,0,0,0,0,
                    0,0,0,0,0,0,0,0,0,0,0,0
```

А. ИНГОРЬ

Удобный аналог команды MERGE

Для объединения текстов программ, написанных на языке Бейсик-БК, служит заявленная, но не реализованная в вильнюсской версии языка 1987 г. команда MERGE. Вместе с тем имеется ряд программ в машинных кодах, восполняющих этот недостаток, однако для их работы требуется запись текстов объединяемых программ в крайне неэкономном формате команды SAVE, а подавляющее большинство пользователей БК предпочитают ей команду CSAVE, позволяющую хранить на дефицитной магнитной ленте программы в наиболее компактном виде.

В связи с этим практическая значимость имеющихся на сегодня программ-аналогов команды MERGE невысока и требуется поиск новых подходов к решению этой проблемы.

Предлагаемая программа CMERGE позволяет объединять в ОЗУ БК программы, записанные в формате команды CSAVE. Программа перемещаема и может загружаться с любого адреса; например, можно воспользоваться приводимой ниже процедурой и загрузить ее в область буфера команды SAVE, что является наиболее удобным с точки зрения экономии адресного пространства в Бейсик-системе БК. Загруженную программу можно записать на МЛ командой BSAVE и пользоваться ею по мере надобности, вызывая командой BLOAD «CMERGE», &O37400,Р. Всякий раз при этом программа CMERGE будет запрашивать имя добавляемой программы из пяти символов и по команде **ВВОД** объединять по логике команды MERGE нужный текст с текстом программы в ОЗУ БК.

Для многократного вызова программы CMERGE удобно определить ее как функцию пользователя командой DEFUSR=&O37400 и вызывать на исполнение командой A==USR(A) с параметром любого типа. Перед вызовом программы CMERGE следует произвести процедуру «сборки мусора», поскольку в результате редактирования или применения программы CMERGE строки текста программы в памяти БК могут расположиться не вплотную друг к другу, а при объединении нескольких больших фрагментов это может привести к потере текста. Сжать текст в памяти БК и убрать «мусор» можно командой RUN, не обращая внимания на ошибки ее исполнения.

Для объяснения принципа программы CMERGE рассмотрим размещение текстов программ в памяти БК. Текст с ленты записывается сразу же вслед за текстом в памяти БК, причем на место сшивки указывает содержимое слова &O2002, а длина загруженного текста хранится в слове &O266. Строки программ в обоих текстах хранятся без номеров и не всегда в порядке их воз-

растания. Списки номеров и адресов соответствующих строк хранятся отдельно: для исходного текста список идет с адреса, записанного в &O2026, в сторону уменьшения, а для подгружаемого текста список идет сразу же вслед за текстом. Адреса строк в списке соответствуют загрузке строк с адреса &O3052, причем второе слово от конца загруженного текста указывает на конец списка. Для каждой строки текста в списке выделены три слова, важны два из них: слово со старшим адресом содержит номер, а следующее за ним — адрес строки.

Зная эту структуру данных, можно сформировать новый список строк, имитируя работу оператора MERGE и взяв в качестве буфера экранную область памяти БК. По завершении формирования списка он копируется программой CMERGE из буфера на место списка строк исходной программы.

Процедура загрузки программы CMERGE:

```
10 DATA &012701,&0320,&012721,&03,.....
.....
.....
```

```
100 FOR I%=0 TO 74
110 READ A%
120 POKE &037400+I%%2,A%
130 NEXT I%
140 DEF USR=&037400
```

Текст программы:

12701	320	12721	3	13721
2002	5021	12703	100724	4737
100460	12702	5005	12700	334
4737	137670	4737	100566	105700
1064	13705	264	63705	266
10502	14500	160005	12700	2024
13704	2002	162704	3052	10537
2002	34242	13701	2026	5741
12703	52000	20205	3423	20110
3410	21211	2406	1413	11143
14143	14143	5741	764	11243
14243	60413	14243	5742	756
162701	6	767	20110	3360
12701	52000	13702	2026	14142
20103	3375	10210	137	120234

А. ГАРМАШОВ

Оснащаем компьютеры УКНЦ («Электроника МС0511») жесткими дисками типа «Винчестер», кассетами ППЗУ с Бейсиком.

Оказываем всем предприятиям и организациям, заинтересованным в закупках разнообразных импортных товаров народного потребления за рубли, посреднические услуги: помощь в заключении контрактов с инофирмами и совместными предприятиями. Списки предлагаемых к закупке товаров высылается по вашей письменной заявке.

Приглашаем ваше предприятие стать полноправным участником вновь создающихся предприятий в свободной экономической зоне г. Зеленограда. Кооператив «Электрон» и предприятие «Восток Лтд» берут на себя регистрацию предприятия, обеспечивают офисом, телефоном, менеджерами, бухгалтером.

Гарантируем прибыльную работу создаваемых предприятий. Все, что требуется от вашего предприятия — это заключить с «Восток Лтд» договор, определяющий долю участников в уставном капитале и процент, идущий на оплату наших услуг.

Высылаем наложенным платежом:

- комплект рабочей документации для изготовления малогабаритного недорогого станка для изготовления стенных шлакоблоков;
- книги и брошюры по программному и аппаратному обеспечению ПЭВМ «Синклер ZX Спектрум»;
- комплекты документации по созданию обществ с ограниченной ответственностью, акционерных обществ, малых предприятий;
- справочник адресов и телефонов зарубежных фирм, аккредитованных в Москве.

Оказываем всем предприятиям услуги по маркетингу наиболее эффективным способом: разошлем рекламу вашей продукции по 40000 адресов предприятий и учреждений.

Приглашаем к сотрудничеству дилеров для помощи в реализации продукции и услуг кооператива «Электрон» и предприятия «Восток Лтд» с выплатой процентов от реализации.

Почтовый адрес для всех справок, запросов каталогов и заявок: **103489, Москва, Зеленоград, корпус 705, кооператив «Электрон».**

По желанию заказчиков каталоги, программы, компьютеры, документация высылаются по почте.

Телефон для справок: 536-12-81 с 12 до 18 часов.

Молодежный компьютерный центр «КОНТАКТ»

предлагает пользователям БК-0011

Пакет из 25 учебных и игровых программ.

Эффективное средство для перевода Бейсик-программ с БК-0010.01, позволяющее:

исправить недостатки Бейсика БК-0010, ограничивающие применение операторов LINE и DRAW;

автоматически масштабировать переносимое графическое изображение.

Эмулятор, дающий возможность без переделки использовать большинство существующих на БК-0010 кодовых программ с сохранением звуковых эффектов.

Наш адрес:
614010, Пермь, а/я 4694.

И. БЕЛАЯ, А. ДУБИЛЬТ, О. ЕГОРЫШЕВА, И. ЮРИНА

Экспертная система определения «белых пятен» в знаниях ученика

Одним из направлений развития программного обеспечения для обучения являются экспертные системы (ЭС), которые являются, по сути, моделью эксперта очень высокой квалификации в конкретной предметной области. Существует много различных применений ЭС, а именно: составление прогноза, постановка диагноза, задачи управления, планирования, обучения и др.

Рассмотрим построение и использование ЭС для учебных целей. Функции преподавателя условно можно разделить на изложение нового материала, осуществление контроля знаний учащихся, проведение консультаций. Это сложная задача, которая решается только на основе богатейшего опыта преподавателя, его профессиональной квалификации. Для учителя, который пока не обладает таким уровнем квалификации, весьма полезным помощником может стать ЭС.

В данной статье на конкретном примере показано использование экспертных систем для выявления «моментов незнания», повлекших за собой некачественное усвоение материала в определенной предметной области. Конечно, преподавателю было бы удобнее иметь оболочку ЭС, которую он, как педагог — непрограммист, мог бы наполнять конкретным содержанием, но на данном этапе развития ЭС для подобного класса задач построение такой оболочки крайне затруднено.

В качестве предметной области нами был выбран курс тригонометрии в объеме школьной программы. Этот раздел математической дисциплины сравнительно небольшой, достаточно замкнутый, связи между понятиями в нем уже установлены и легко прослеживаются. Есть эксперты, определяющие границы предметной области, типовые задачи и ключевую информацию для построения алгоритма решения задач. По названию предметной области разработанная ЭС получила название «Тригонометрия». Описанный подход может быть распространен и на другие предметы.

Представление знаний в ЭС «Тригонометрия»

Представление знаний — это соглашение о том, как описывать конкретную предметную область. Способ представления знаний в ЭС зависит от решаемой задачи и ее специфики. В существующих ЭС для представления знаний чаще всего используются следующие формализмы: логические исчисления, фреймы, семантические сети и продукционные системы.

Для описания предметной области «Тригонометрия» в качестве основных понятий выбраны следующие:

угол;

прямые тригонометрические функции;

обратные тригонометрические функции;

элементарные тригонометрические уравнения;

типы тригонометрических уравнений.

Основные понятия курса объединены в иерархическую структуру, изображенную на схеме 1. В основании структуры лежит узел, содержащий понятие угла — простейшее понятие курса; ниже располагаются узлы, соответствующие прямым и обратным тригонометрическим функциям, элементарным тригонометрическим уравнениям и, наконец, более сложным основным типам тригонометрических уравнений.

Будем считать, что учащийся для того, чтобы овладеть понятием, располагающимся в узле более низкого уровня, должен овладеть всеми понятиями, располагающимися в узлах более высокого уровня. Такая структура представляет собой ориентированный граф, узлы которого соответствуют основным понятиям курса «Тригонометрия». Для решения задачи определения «моментов незнания» учащегося в узлах графа должна помещаться такая информация, которая позволяла бы определять степень усвоения понятия конкретного узла, а также информация о том, как нужно обрабатывать данный узел и куда переходить после его обработки. Весь граф делится на две части: верхнюю —

от корневой вершины до узлов, определяющих элементарные тригонометрические уравнения, и нижнюю — от узлов, определяющих элементарные тригонометрические уравнения, и ниже.

Методики опроса, объяснения материала, содержащегося в узлах нижней и верхней частях графа, существенно различны, поэтому все узлы графа по способу их описания и обработки при построении алгоритма решения задачи можно разделить на два типа. Каждый узел графа представляет собой фреймоподобные конструкции, в слотах которых содержится вся необходимая информация.

Для представления знаний в разрабатываемой экспертной системе выбран фреймовый формализм, причем фреймы, описывающие отдельные информационные фрагменты, соответствующие основным понятиям курса, образуют иерархическую структуру — ориентированный граф с двумя типами узлов.

Фрейм, содержащий информацию об узле первого типа (схема 2), содержит следующие слоты:

NCL — номер фрейма (числовой параметр);

Type — тип фрейма (числовой параметр);

Text — вопросы и указания пользователю (текст);

Options — варианты ответов (текст);

Help — помощь ученику, выдаваемая в случае неправильного ответа (текстовая информация);

Answer — список номеров правильных ответов (числовые параметры);

Subtext — список дополнительных текстов (промежуточные выкладки, оценка сделанного шага);

Subclass — список номеров фреймов, на которые возможен переход в случае правильного решения (числовые параметры);

NCLO — номер фрейма, на который возможен переход в случае неправильного решения (числовой параметр).

НОМЕР
ТИП
ТЕКСТ ВОПРОСОВ
СПИСОК ВОЗМОЖНЫХ ОТВЕТОВ
ПОМОЩЬ
СПИСОК НОМЕРОВ ПРАВИЛЬНЫХ ОТВЕТОВ
СПИСОК ДОПОЛНИТЕЛЬНЫХ ТЕКСТОВ
СПИСОК НОМЕРОВ ФРЕЙМОВ ПЕРЕХОДА ПО ПРАВИЛЬНОМУ ОТВЕТУ
НОМЕР ФРЕЙМА ПЕРЕХОДА ПО НЕПРАВИЛЬНОМУ ОТВЕТУ

Фрейм первого типа

НОМЕР
ТИП
ТЕКСТ, ЗАДАЮЩИЙ ПОСЛЕДОВАТЕЛЬНОСТЬ ДЕЙСТВИЙ
СПИСОК КОНТРОЛЬНЫХ ВОПРОСОВ
СПИСОК ВОЗМОЖНЫХ ОТВЕТОВ
СПИСОК НОМЕРОВ ПРАВИЛЬНЫХ ОТВЕТОВ
НОМЕР ФРЕЙМА ПЕРЕХОДА

Фрейм второго типа

Фрейм второго типа (схема 3) включает в себя следующие слоты:

NCL — номер, под которым данный фрейм обозначен в других фреймах графа (числовой параметр);

Type — тип фрейма (числовой параметр);

Text — текст, задающий последовательность действий на данном шаге;

Texte — список контрольных вопросов (текстовая информация);

Options — варианты ответов (текст);

Answer — список номеров правильных ответов (числовые параметры);

NCLO — номер фрейма, на который переходим в любом случае (числовой параметр).

Алгоритм обработки фреймов

В начал работы программы всегда происходит обращение к фрейму первого типа.

Рассмотрим пример обработки фрейма первого типа (схема 4). Каждому фрейму соответствуют свой номер (N), управляющий и поясняющий текст, меню вариантов правильных и неправильных ответов, список правильных ответов, список номеров фреймов, на которые возможен переход при правильном ответе, номер фрейма перехода при неправильном ответе.

Ученику задается вопрос и высвечивается меню ответов (правильных и неправильных). Если ученик выбрал правильный ответ, то он переходит на фрейм следующего уровня, который также является фреймом первого типа. Промежуточные этапы решения выводятся на экран дисплея (преобразования, связанные с использованием основных тригонометрических формул, приведение подобных членов и т. д.).

«Правильным направлением» назовем путь нахождения решения исходного уравнения. В случае безошибочного прохождения до конца по всем узлам данной ветви «правильного направления» ученику выдается текст-поощрение.

Мы рассмотрели ситуацию, когда возможно несколько правильных ответов. Существуют ситуации, когда возможен только один правильный ответ. В этом случае и список правильных ответов, и список узлов, по которым происходит переход, состоят из одного элемента.

В случае выбора ошибочного ответа на некотором этапе информация об ошибке запоминается. Ученику выдается текст помощи в принятии решения на данном шаге, и происходит возврат к рассматриваемому фрейму. При повторном выборе неправильного ответа возможны действия двух типов.

1. Анализ ошибки, допущенной учеником, говорит о том, что незнание относится не к курсу тригонометрии (например, ошибка в

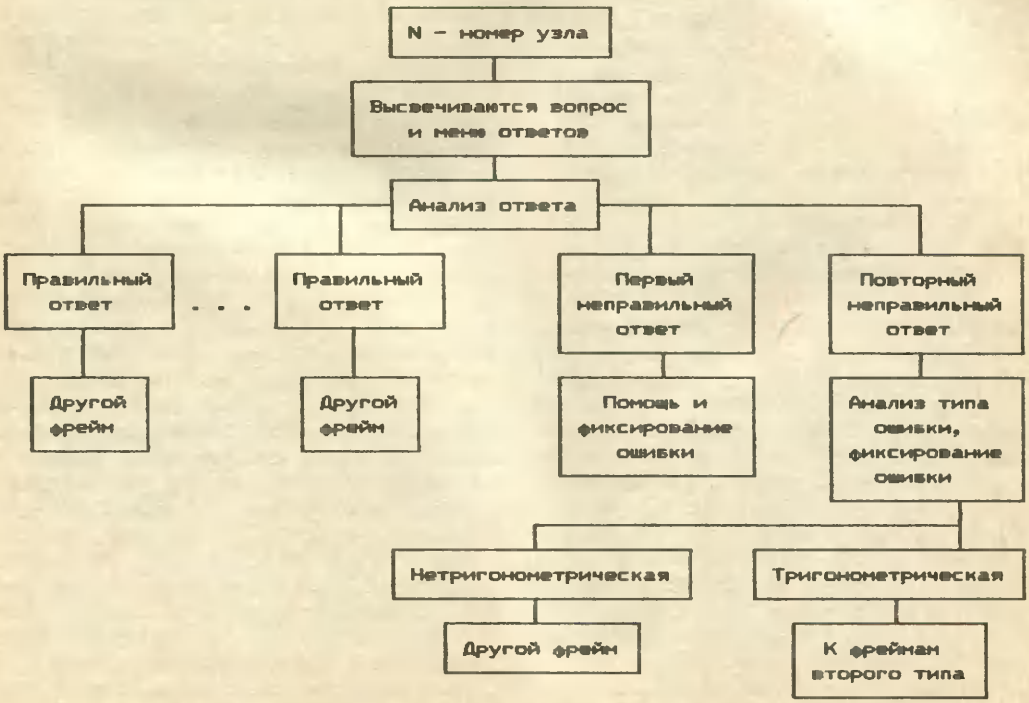


Схема обработки фрейма первого типа

нахождении корней алгебраического уравнения). Следующим действием работы программы является переход к узлу более низкого уровня в правильном направлении.

2. Анализ повторной ошибки, допущенной учеником, говорит о том, что незнание относится к одному из разделов тригонометрии. В этом случае работа программы по уточнению конкретного понятия, которое ученик не усвоил, происходит по ветке графа, состоящей только из фреймов второго типа. Запомним, что ошибка сделана повторно, и будем шаг за шагом спускаться по ветке из фреймов второго типа для уточнения этого незнания.

Концом этой ветви является фрейм, содержащий информацию о понятии угла. В процессе прохождения по данной ветви информация об ошибках, сделанных на каждом шаге, запоминается.

Рассмотрим обработку фреймов второго типа (схема 5).

Схема обработки фрейма второго типа

Каждому фрейму соответствуют номер (N), перечень вопросов, меню возможных ответов, список номеров правильных ответов, номер узла более низкого уровня, на который происходит безусловный переход. Ученику по очереди задаются вопросы из списка вопросов, который соответствует фрейму с данным номером. На экране дисплея высвечивается меню ответов и предлагается выбор. При неправильном ответе на поставленный вопрос фиксируется ошибка. После того как все вопросы из списка будут заданы, происходит безусловный переход на фрейм более низкого уровня.

На последнем шаге работы программы производится анализ сделанных ошибок и выдаются рекомендации по устранению «белых пятен» в знаниях ученика по курсу «Тригонометрия».

Рассмотрим последовательность обработки фреймов на конкретном примере.

Пусть фрейм первого типа имеет вид:

1
1
НАЙТИ РЕШЕНИЕ УРАВНЕНИЯ cos2x+sinx=0
1. sin2x=2sinx*cosx 2. cos2x=(cosx)^2-(sinx)^2 3. cos2x=1-2*(sinx)^2 4. cos2x=2*(cosx)^2-1 5. Замена sinx=Y 6. Обратная замена Y=sinx 7. (sinx)^2+(cosx)^2=1 8. 1-(sinx)^2=(cosx)^2 9. 1-(cosx)^2=(sinx)^2 10. Найти корни квадратного уравнения
ПРИМЕНИТЕ ФОРМУЛУ ДВОЙНОГО УГЛА
[2, 3, 4]
[" "]
[21, 3, 23]
13

После активизации данного фрейма на экране дисплея ученику предлагается решить уравнение, хранящееся в слоте 3, и высвечивается меню ответов из слота 4. Например, ученик при решении предложенного уравнения воспользовался формулой 3, на экране появляется преобразованное уравнение с учетом этой формулы: $1-2\sin^2 x+\sin x$. Информация по преобразованию исходного уравнения хранится в слоте 7. Этот ответ ученика является правильным, поэтому переходим к другому фрейму первого типа с номером, хранящимся в слоте 8, который обрабатывается аналогичным образом. В случае неправильного ответа, данного учеником, происходит фиксирование ошибки и выдается «помощь», информация о которой хранится в слоте 5. После этого ученику предлагается тот же самый вопрос, на который он не смог дать правильный ответ. Если повторный ответ оказался правильным, то осуществляется переход к другому фрейму первого типа.

Если повторный ответ оказался неправильным, то фиксируется повторная ошибка и

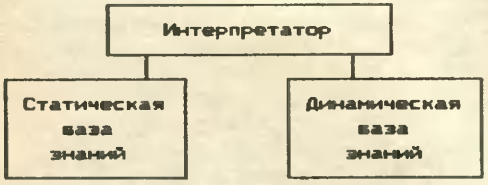
осуществляется переход к фрейму второго типа, который в данном случае имеет номер 13 и содержит информацию о понятии — обратные тригонометрические функции:

11
2
ОТВЕТИТЬ НА ВОПРОСЫ
1. arcsin1=? 2. arcsin(-0.5)=?
1. π/2+2πn 2. ((-1)^2(n+1)*π+πn 3. π/2 4. -π/6 5. 0 6. 270 7. π
[3, 4]
14

После активизации фрейма с номером 13, который является фреймом второго типа, на экране дисплея появляется информация, хранящаяся в слотах 2 и 4, причем вопросы задаются по очереди. В случае правильного ответа на оба вопроса осуществляется переход к фрейму с номером 14. В случае неправильного ответа хотя бы на один из вопросов фиксируется ошибка и осуществляется переход к фрейму с номером 14.

Реализация ЭС

На схеме 6 представлена логическая структура системы:



Для данной системы разработан естественный для нее язык представления знаний. Интерпретатор этого языка на каждом цикле своей работы, исходя из текущего состояния динамической базы знаний, выбирает соответствующий фрагмент статической базы знаний, обрабатывает его в соответствии с алгоритмом и модифицирует динамическую базу знаний.

Интерпретатор языка представления знаний реализован на языке Turbo — prolog.

Динамическая часть базы знаний

Для работы Пролог-программы необходимо описать динамическую часть базы знаний. Под динамической базой знаний понимается рабочее поле, куда принимаются текущие сведения, необходимые для решения задачи, изменяющиеся в процессе решения.

Для организации работы динамической части базы знаний определены следующие предикаты:

```
next_class (integer)
mem_inf (integer)
end
```

Интерпретатор представляет собой управляющую программу, которая осуществляет поиск решения.

Язык представления знаний в синтаксисе turbo — prologa выглядит следующим образом.

Статическая часть базы знаний

В статической части базы знаний хранятся неизменяемые данные о предметной области. Для хранения информации, содержащейся в слотах фрейма I типа, определены следующие предикаты:

1. «Класс»:

class (N, Type' Text).

2. «Меню»:

class_menu (n, Options, Help, Answer, Subtext, Subclass, NO).

Для хранения информации, содержащейся в слотах фрейма II типа, определены следующие предикаты:

1. «Класс-пример»:

class (N, Type, Text);

2. «Переход»:

class_ex_cont (N, NO);

3. «Меню-пример»:

class_menu_example (N, Text, Options, Answer).

Разработанная экспертная система определения «белых пятен» в знаниях ученика («Тригонометрия») может служить примером применения новых информационных технологий в образовании.

В. ЕСЬКОВ
учитель СШ № 1 г. Луганска

От слов к делу

В физико-математической СШ № 1 г. Луганска работает творческая группа, цель которой — разработка программного обеспечения для курса математики средней школы. В настоящее время закончена работа по созданию компьютерного курса «Алгебра и начала анализа — 10», который предназначен для проведения уроков в школе, ПТУ и техникуме. Содержание соответствует действующей программе по математике для общеобразовательной школы.

Курс содержит 4 пакета и включает 33 учебные программы и 2 сервисные, служащие для организации работы с пакетами и изменений (увеличение) размера символов на экране монитора. Программы написаны для «Ямахи» на Бейсике, версия MSX 2.1. Пакет занимает 658.134К байта. Программы независимы друг от друга, что позволяет работать с ними в условиях КУВТ с одним дисководом. Ниже приводится перечень программ.

Тригонометрические функции. Основные тригонометрические формулы

1. Радианное измерение углов.
2. Определение тригонометрических функций.
3. Знаки тригонометрических функций. Основное тригонометрическое тождество.
4. Тригонометрические функции суммы и разности углов.
5. Тригонометрические функции удвоенного аргумента.
6. Формулы приведения.
7. Тригонометрические функции половинного угла.
8. Сумма и разность тригонометрических функций.

Свойства тригонометрических функций и их графики

1. Четность и нечетность тригонометрических функций.
2. Периодичность тригонометрических функций.
3. Свойства и график функции $y=\sin x$
4. Свойства и график функции $y=\cos x$.
5. Свойства и график функции $y=\operatorname{tg} x$
6. Свойства и график функции $y=\operatorname{ctg} x$

Тригонометрические уравнения и неравенства

1. Обратные тригонометрические функции.
2. Простейшие тригонометрические уравнения.
3. Тригонометрические уравнения, сводимые к алгебраическим и решаемые разложением на множители.
4. Однородные тригонометрические уравнения I и II степени.
5. Неоднородные тригонометрические уравнения I и II степени.
6. Тригонометрические уравнения, решаемые понижением степени и решаемые с использованием формул разложения произведения тригонометрических функций в сумму.
7. Простейшие тригонометрические неравенства.
8. Более сложные тригонометрические неравенства.

Производная и ее применение

1. Задачи, приводящие к понятию производной. Определение производной. Нахождение производной по определению.
2. Теоремы о производных. Нахождение производных с их использованием.
3. Геометрический смысл производной. Уравнение касательной.
4. Механический смысл производной. Скорость и ускорение.
5. Решение неравенств методом интервалов.
6. Производные тригонометрических функций.
7. Исследований функций на монотонность.
8. Исследование функций на экстремумы.
9. Построение графиков функций. Часть 1.
10. Построение графиков функций. Часть 2.
11. Гармонические колебания.

По дидактическим целям предлагаемые уроки с применением компьютеров являются уроками-практикумами, т. е. уроками выработки практических умений и навыков для решения основных типов задач. На этих уроках изучаются особенности, реализуется принцип вариации несущественного в условии задачи, решаются более сложные задачи, основанные на типовых.

Программы имеют сходную структуру, изображенную на схеме. Это позволяет учащимся уверенно с ними работать.

Каждая программа начинает работу с наглядного образного представления теоретического материала, для чего используются динамические возможности компьютерной графики. Учебная информация предъявляется в темпе, рассчитанном на «среднего» ученика. Предполагается, что учащиеся не ведут записей, а лишь знакомятся с информацией на экране монитора. Если предлагаемый темп слишком низок, то ученик может его увеличить путем нажатия произвольной клавиши ПЭВМ. Если темп высок, то не нужно расстраиваться — любой фрагмент материала можно просмотреть сколько угодно раз при выполнении тренировочных заданий с помощью подсказки.

После демонстрации теоретического материала рассматривается его применение для решения типовых задач. Приводятся образцы решений, которые сопровождаются ссылками на соответствующие положения теории и могут служить образцами оформления; темп регулируется учеником.

В программе пересылки предусмотрена возможность, по желанию учителя, обяза-

тельного рассмотрения теоретического материала учащимися, а также работа с программами лишь в качестве тренажеров. Вместе с тем, даже в этой ситуации ученикам предоставлена возможность изучения теоретического материала. Если же ученик повторно работает с программой, то вопрос изучать или не изучать теоретический материал решается учеником самостоятельно.

Далее перед учащимся ставится вопрос о том, хочет ли он потренироваться в решении типовых задач. Если ответ — да, то ему предлагаются тренировочные задания, при решении которых он может пользоваться подсказками. Они не подсчитываются и на оценку не влияют. Если учащийся допускает при выполнении тренировочных заданий более пяти ошибок, то ему будет предложено рассмотреть теоретический материал заново. Если же ученик уверен в своих силах, то он может отказаться от выполнения тренировочных заданий и сразу выйти в режим контрольных заданий.

В контрольных заданиях учащимся предлагается решить несколько примеров, большая часть которых по трудности не превосходит примеры из обязательных по программе. Задания выбираются случайным образом из 6561 варианта. При решении этих примеров можно также воспользоваться подсказками, но теперь их использование влечет снижение оценки. Учащимся следует учитывать, что записываемые ответы должны быть упрощены и преобразованы к наиболее простому виду, дроби — к неправильным, без иррациональности в знаменателе. Форма записи ответов — общепринятая. Если при их записи используются специальные символы ($\sqrt{}$, π, ∞ и т. д.), то их набор оговаривается в программе. Для набора ответов не нужно знание особенностей клавиатуры, надо только нажать клавишу, на которой есть нужный символ. Редактирование ответа производится с помощью клавиши BS — удаление символа слева от курсора; набор ответа завершается нажатием клавиши BK.

После ввода последнего ответа производится их сравнение с верными, вывод тех и других на экран, вывод количества подсказок и окончательная оценка за выполнение контрольных заданий.

Если учащийся работал с программой несколько раз, то на экран выводятся все оценки, получаемые за урок.

Программы использовались при проведении уроков математики в СШ № 1 г. Луганска, а также на семинарских занятиях на курсах повышения квалификации учителей информатики и математики при Луганском институте усовершенствования учителей, рассмотрены на заседании кафедры геометрии и методики преподавания математики Луганского государственного педагогического института им. Т. Г. Шевченко и получили одобрение. Пакет «Производная и ее применение» представлялся на I Всесоюзный конкурс профессиональных, учебных и шахматных программ, награжден Почетной грамотой ЦК ВЛКСМ.

Использование программ дает возможность интересно и содержательно проводить уроки математики в классе с компьютерами и при этом помочь в решении ряда проблем, стоящих перед современной школой, а именно: объективно реализовать идеи индивидуального, дифференцированного подхода в процессе обучения;

оказать помощь учителю в организации действенного систематического контроля, разгрузив при этом учителя от рутинной работы по контролю за уровнем знаний, умений и навыков учащихся;

создать психологическую обстановку в процессе обучения, исключающую всякие конфликты в звене ученик — учитель, необъективную оценку деятельности учащихся;

активизировать познавательную деятельность, выйти на более высокий уровень восприятия и усвоения изученного материала.

Всем, кто уже сейчас хотел бы использовать компьютеры при преподавании математики, и тем, кто интересуется этой проблемой, предлагаем обращаться по адресу:

348031, УССР, г. Луганск, ул. Луганской правды, 151, средняя школа № 1, Еськову В. Л.

Надеемся найти в вашем лице единомышленников и обещаем учитывать ваши предложения по разработке и созданию новых программных средств.

И. ЗЕЛЬЦЕР

Простые числа на уроках информатики

В данной статье предлагаются алгоритмы, которые можно составить с учащимися в качестве вспомогательных перед разбором алгоритма «Простые числа» (ИНФО. 1987. № 5. С. 71), а также ряд задач, используемых на уроках или кружковых занятиях в X и XI классах.

Рассмотрим задачу поиска простых чисел в диапазоне от 1 до N. Ее решение было указано александрийским математиком Эратосфеном (около 200 г. до н. э.). Метод получения простых чисел известен как «решето Эратосфена». Показав учащимся на примере идею «решета», можно предложить им составить алгоритм (X класс) и программу (XI класс) для получения таблицы простых чисел. Алгоритм содержит два цикла: проверка каждого числа от 1 до N «на простоту»; «удаление» чисел, кратных найденному простому.

Далее сформируем таблицу P, в которой на первом месте стоит 1, на втором — 2, ..., на n-м — n. Простые числа в этой таблице останутся неизменны, а числа, кратные простым, будем «удалять», заменяя их единицей. В результате получим таблицу P, в которой все числа, отличные от 1, простые.

Детализируем алгоритм:

```
Цикл 1
 пока просмотреть все числа таблицы P
   нц
      если число i таблицы P не равно 1
        то цикл 2
      все
   кц

Цикл 2
 пока пробежать от 2*i до N в таблице P
                                с шагом i
   нц
      заменить число на 1
   кц
```

В итоге получаем алгоритм:

```
алг Простые числа (цел таб P[1:N],
                                   цел N)
   арг P,N
   рез P
нач цел i,j
   i:=1
    пока i<=N
      нц
        если P[i]<>1
          то j:=i+i
            пока j<=N
              нц
                P[j]:=1
                j:=j+i
            кц
        все
        i:=i+1
      кц
кон
```

В XI классе его можно записать так:

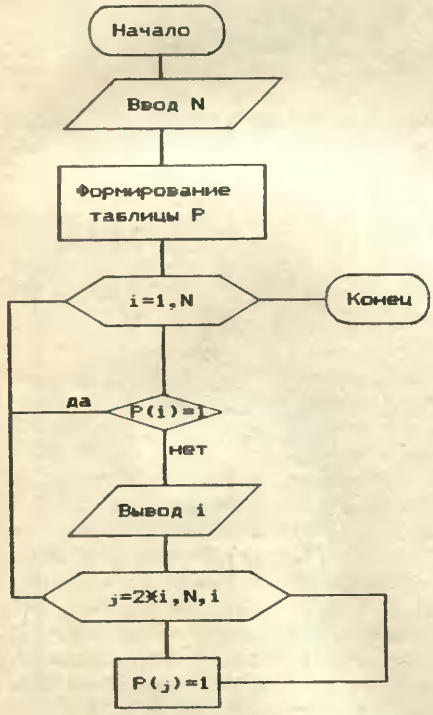
```
алг Простые числа (цел таб P[1:N],
                                   цел N)
   арг P,N
   рез P
нач цел i,j
   для i от 1 до N
      нц
        если P[i]<>1
          то
            для j от 2*i до N шаг i
              нц
                P[j]:=1
              кц
        все
      кц
кон
```

При реализации данных алгоритмов в Е-практикуме не следует забывать, что количество аргументов не должно превосходить 11. В программе существуют ограничения длины интервала, зависящие от конкретной ЭВМ. Результаты лучше всего получить в таблице исполнения алгоритмов (X класс). Составляя программу на Бейсике, можно организовать вывод результатов на экран дисплея, принтер или ГМД.

Рассматриваемый интервал от 1 до 613 можно увеличить, используя в программе признак целого числа и заменив в строках 30, 50, 90, 120 P на P%. Результаты работы программы выводятся на экран дисплея в 5 колонок, но лучше организовать их вывод на принтер или ГМД. Эту задачу можно дать на факультативных занятиях в целях углубления знаний языка программирования и ознакомления с организацией работы различных устройств ЭВМ.

```
10 REM ТАБЛИЦА ПРОСТЫХ ЧИСЕЛ
20 REM ФОРМИРОВАНИЕ ТАБЛИЦЫ P
25 PRINT "До какого натурального числа
                Вы хотите получить таблицу
                   простых чисел";\INPUT N
30 DIM P(6130)
40 FOR I=1 TO N
50 P(I)=I
60 NEXT I
70 REM ЦИКЛ 1
80 FOR I=1 TO N
```

```
90 IF P(I)=1 GOTO 140
100 PRINT I,
105 REM ЦИКЛ 2
110 FOR J=2*I TO N STEP I
120 P(J)=1
130 NEXT J
140 NEXT I
150 END
```



Следующая задача формулируется так: найти все простые числа в интервале от N1 до N2 (где $N1 \geqslant 2$). Точность результатов зависит от возможностей представления чисел в конкретной ЭВМ, например для ДВК-2М $N2 < 1\,000\,000$. Воспользуемся тем, что если число a является составным, то оно имеет делитель среди простых чисел, не превосходящих $\sqrt{a}$. Как и в предыдущей задаче, выделим два цикла: просмотр чисел от N1 до N2; проверка каждого «на простоту».

```
Цикл 1
 пока просмотреть все числа i
                              от N1 до N2
   нц
     таблицу P использовать,
                 начиная с первого числа
     Цикл 2
           формирование таблицы
                        простых чисел P1
   кц
```

```
Цикл 2
 пока число из таблицы P меньше
                              или равно Ji
   нц
     если i кратно числу из таблицы P
       то выйти из цикла 2 и перейти
                    к проверке следующего i
     все
     взять следующее число из таблицы P
   кц
```

Введем блок данных — таблицу P — простых чисел от 1 до 1000. Эти числа можно получить, воспользовавшись результатами предыдущей задачи.

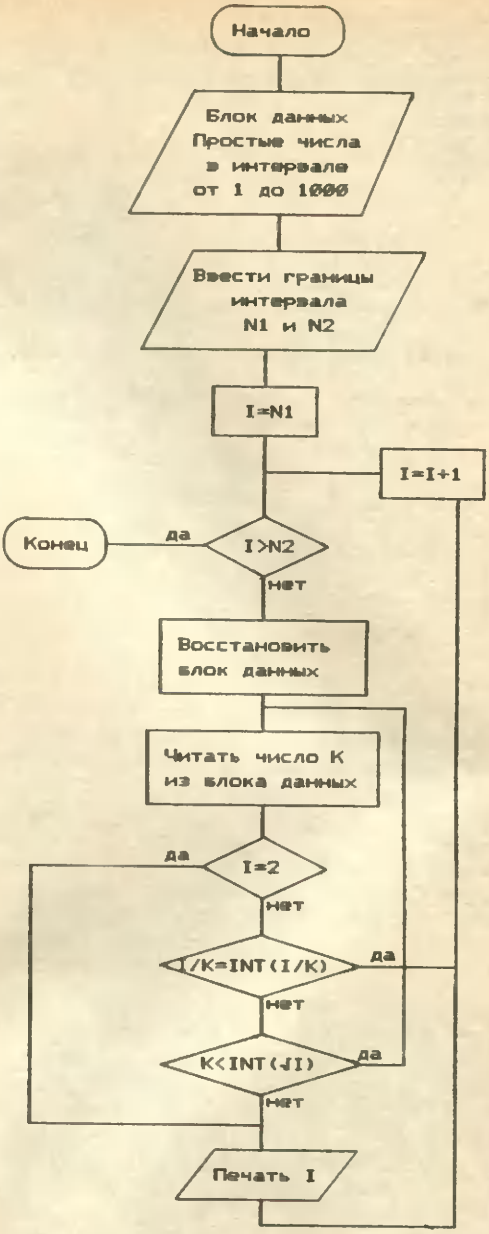
В алгоритме I — управляющая переменная цикла 1, J — индекс очередного простого числа в таблице P1, T — индекс в таблице P, A1 — признак простого (A1=0) или составного (A1=1) числа.


```
алг Простые числа в интервале
                          (цел таб P[1:168],
          цел таб P1[1:78498], цел N1,N2)
   арг P,N1,N2
   рез P1
нач цел i,j,t,a1, вещ a
   i:=N1; j:=1
   пока i<=N2
     нц
       t:=1; a:=SQRT(i); a1:=0
       пока P[t]<=a и a1=0
         нц
           если i/P[t]=INT(i/P[t])
             то a1:=1
           все
           t:=t+1
         кц
       если a1=0
         то P1[j]:=i; j:=j+1
       все
       i:=i+1
     кц
кон
```

Рассмотрим блок-схему и программу на Бейсике.

```
10 DATA 2,3,5,7,11,13,17,19,23,29
20 DATA 31,37,41,43,47,53,59,61,67,71
30 DATA 73,79,83,89,97,101,103,
                                107,109,113
40 DATA 127,131,137,139,149,151,157,
                                163,167,173
50 DATA 179,181,191,193,197,199,211,
                                223,227,229
60 DATA 233,239,241,251,257,263,269,
                                271,277,281
70 DATA 283,293,307,311,313,317,331,
                                337,347,349
72 DATA 353,359,367,373,379,383,389,
                                397,401,409
73 DATA 419,421,431,433,439,443,449,
                                457,461,463
74 DATA 467,479,487,491,499,503,509,
                                521,523,541
```

```
75 DATA 547,557,563,569,571,577,587,
                              593,599,601
76 DATA 607,613,617,619,631,641,643,
                              647,653,659
77 DATA 661,673,677,683,691,701,709,
                              719,727,733
78 DATA 739,743,751,757,761,769,773,
                              787,797,809
79 DATA 811,821,823,827,829,839,853,
                              857,859,863
80 DATA 877,881,883,887,907,911,919,
                              929,937,941
81 DATA 947,953,967,971,977,983,991,
                              997,1009,1013
95 PRINT CHR$(27);CHR$(72);
                         CHR$(27);CHR$(74)
96 PRINT CHR$(27);CHR$(89);
                         CHR$(40);CHR$(32)
100 PRINT "ЕСЛИ ВАС ИНТЕРЕСУЮТ
     ПРОСТЫЕ ЧИСЛА В ЛЮБОМ ИНТЕРВАЛЕ"
105 PRINT "ОТ 2 ДО 1000000, ТО
          РАБОТАЙТЕ С ЭТОЙ ПРОГРАММОЙ"
110 PRINT "УКАЖИТЕ НИЖНЮЮ ГРАНИЦУ
                    ИНТЕРВАЛА - ";\INPUT N1
111 IF N1<2 THEN PRINT "ВЫ
                    НЕВНИМАТЕЛЬНЫ!"\GOTO 110
115 PRINT "УКАЖИТЕ ВЕРХНЮЮ ГРАНИЦУ
                    ИНТЕРВАЛА -";\INPUT N2
116 IF N2>1.0E+06 THEN PRINT "ВЫ
                    НЕВНИМАТЕЛЬНЫ!"\GOTO 115
120 FOR I=N1 TO N2
130 RESTORE
140 READ K
150 IF I/K=INT(I/K) GOTO 180
160 IF K<INT(SQR(I)) GOTO 140
170 PRINT I,
180 NEXT I
185 PRINT
190 PRINT "РАД БЫЛ ПОМОЧЬ. ДО СВИДАНИЯ!"
200 END
```



Здесь в строках 10—81 — блок данных;

95 — очищение экрана;

96 — организация вывода текста (строки 100—110) посередине экрана;

111, 116 — проверка вводимых границ интервала;

120—180 — реализация цикла 1;

140—160 — реализация цикла 2;

170 — вывод простых чисел на экран дисплея.

Поскольку представленные программы неидеальны, то задача улучшения их, возможно, также будет интересна школьникам.

Замечания по поводу одной задачи

С алгоритмом решения уравнения $ax^2+bx+c=0$ по формуле

$$x_{1,2}=\frac{-b\pm\sqrt{b^2-4ac}}{2a}$$

знакомы многие читатели.

При составлении алгоритмов следует выбирать вариант, позволяющий производить минимум вычислительных операций (при работе вручную и с микрокалькулятором) и экономно использовать ресурсы при работе на ПМК с его «бедной» памятью.

С этих позиций искать корни квадратного уравнения рациональнее по формуле

$$x_{1,2} = -\frac{b}{2a} \pm \sqrt{\left(\frac{b}{2a}\right)^2 - \frac{c}{a}},$$

и алгоритм имеет вид:

```
алг КВУР (вещ a,в,c,×1,×2, лит y)
    арг (a,в,c)
    рез (×1,×2,y)
нач в:=-в/(2×a); a:=в^2-c/a
    если a<0
        то y:="нет решения"
        иначе y:="есть решение";
              a:=√a; ×1:=в+a; ×2:=в-a
    все
кон
```

Алгоритм наглядно показывает, как сложное для учащихся понятие "переменная" приобретает совершенно реальный смысл — это определенный участок памяти для записи информации. Имя переменной — символ для обозначения этого участка, позволяющий отличить его от других. Если на уроках алгебры a, b, c — коэффициенты уравнения и так до конца вычислений ими и останутся, то в информатике дело обстоит по-другому. Например, переменная a вначале — коэффициент при x^2, потом изменяется (т. е. в эту ячейку будет записано) на значение целого выражения $\left(\frac{b}{2a}\right)^2 - \frac{c}{a}$, а далее и оно поменяется на $\left(\frac{b}{2a}\right)^2 = \frac{c}{a}$.

Понимание учащимися такого рода особенностей — существенный этап на пути к приобретению навыков алгоритмического мышления.

А. АБЕЛЬСКИЙ

Проблемное обучение на уроке информатики

Одним из способов активизации познавательной деятельности учащихся на уроке является создание проблемных ситуаций, в особенности таких, когда поставленная проблема интересна и личностно значима для учащихся. При проведении уроков информатики можно с успехом использовать этот методический прием.

Известно, что проблемное обучение заключается в создании перед учащимися проблемных ситуаций, осознании, формулировании и разрешении проблемы в процессе совместной деятельности учащихся и учителя [1]. Опишу, каким же образом реализовывались эти этапы на уроке информатики при изучении темы «Алгоритмы».

Для того чтобы привлечь внимание учащихся к понятию алгоритма, объяснить его сущность и создать устойчивую мотивацию для дальнейшего изучения темы, я объявлял, что, зная «секретный» алгоритм, могу отгадать возраст любого находящегося передо мной человека. Более того, уверял учащихся, что сейчас же могу отгадать возраст их родителей, друзей и т. д.

Данное сообщение, как правило, вызывало у учащихся противоречивую реакцию: от недоверия до заинтересованности. Звучали реплики: «Да не может этого быть!», «Что это за алгоритм?», «Давайте проверим!». Эмоциональная обстановка накалялась быстро и познавательный интерес к поставленной проблеме был налицо.

Затем каждому желающему принять участие в эксперименте предлагалось совершить следующие действия:

1. Задумать любое число.
2. Умножить его на 9.
3. Прибавить к полученному результату возраст папы, мамы, брата и т. д.
4. Полученный результат сказать преподавателю. (Вариант самого простого алгоритма [2].)

В данной ситуации меня интересовало только число, полученное в результате выполнения алгоритма, после чего через несколько секунд учащийся получал, как правило, правильный ответ относительно возраста задуманного им человека. Иногда я старался придать этому эксперименту характер психологического опыта, но, «раскрывая» глаза учащимся на суть эксперимента, подчеркивал, что в нем нет никакой мистики, а «правит балом его величество» алгоритм отгадывания. Конечно, это производило впечатление, и после нескольких таких «сеансов» ребят больше всего интересовали вопросы: «Как происходит отгадывание? Что такое и каков алгоритм отгадывания?» Налицо был интерес к разрешению проблемной ситуации и огромное желание учащихся узнать или отгадать «секретный» алгоритм. Таким образом, раскрывая для себя и потом применяя этот несложный алгоритм на практике, учащиеся осознанно «впитывали» понятие алгоритма, убеждаясь, что действительно «некоторая последовательность действий» может приводить к весьма интересным результатам. Конечно, не всегда данный алгоритм отгадывания открывался самими учащимися, чаще всего это происходило совместно с преподавателем. Но следующий «секретный» алгоритм показывался в классе, а отгадывание его задавалось на дом. Он был проще, но не менее эффектен и привлекателен, а главное — учащиеся были в со-

стоянии составлять алгоритм отгадывания самостоятельно, что с удовольствием и делали.

Отмечалось, что после описанного урока интерес к предмету значительно повышался, чему способствовало использование в дальнейшем других математических фокусов и головоломок, описанных в [2].

Впоследствии меня очень заинтересовала реализация вышеописанного эксперимента на ПЭВМ. В результате получалась программа, в которой роль преподавателя исполняет ПЭВМ, ведущая дружеский диалог с пользователем и с успехом отгадывающая его возраст. Надеюсь, что данная программа пригодится преподавателям ОИВТ, имеющим в кабинете КУВТ-86. Желающие ее иметь могут обратиться к автору по московскому телефону: 362-72-11. Возможен обмен другими обучающими и тестовыми программами.

Литература

1. *Кудрявцев Т. В.* Психология профессионального обучения и воспитания. М.: МЭИ, 1985. 106 с.

2. *Гарднер М.* Математические чудеса и тайны. М.: Наука, 1982. 128 с.

С. ЖАРИКОВ

С. ЛАНДО, М. НЕСТЕРОВА

Математические компьютерные проекты

Преподавание информатики в школах, оснащенных компьютерами, и в компьютерных центрах пробудило у многих старшеклассников интерес к программированию. В то же время преподаватели зачастую испытывают недостаток в задачах для практического решения. Особенно это касается задач для сильных, развитых школьников с ярко выраженными интересами.

Цель настоящей статьи — предложить список таких задач. Основу его составили темы, над которыми работали участники летней Международной детской компьютерной школы в г. Переславле-Залесском в период с 1986 по 1989 г. Школу проводит Институт программных систем АН СССР.

Работа над предлагаемыми задачами велась по методу проектов. Каждая из них — некоторый законченный этюд из той или иной области знаний. Отличительная особенность метода проектов — развитие задачи в процессе ее решения и разнообразие методов, применяемых при исследовании рассматриваемого вопроса. Обычно над проектом работает несколько человек, общаясь между собой для генерации новых идей. Однако и один исполнитель может достичь хороших результатов.

Важная сторона использования проектов — разнообразие их тематики. Один и тот же программистский материал может быть освоен учащимися на самых разных задачах, соответствующих их индивидуальным способностям. Другая сторона индивидуализации — самоадаптация уровня сложности проекта к уровню знаний учащегося. Поэтапность проекта позволяет начиная с элементарных вещей постепенно переходить (или не переходить) к более сложным или к созданию более содержательных программ, развивая сделанное ранее. Особенно разительно эта сторона метода проектов проявляется в программировании игровых алгоритмов: простейший (и весьма полезный) алгоритм может записываться в две строчки, а построение эффективных алгоритмов не уступает по сложности любым другим задачам программирования.

При работе по проекту очень важны действия преподавателя. Его задачи: стимулировать и поддерживать интерес школьников, направлять деятельность, своевременно задавая вопросы и помогая в преодолении технических трудностей, организовывать обсуждение, стимулировать генерацию идей и вопросов школьникам, помогать в работе над литературой и в составлении отчета. В силу этого преподаватель обязан «владеть задачей» — видеть пути ее развития, связи с другими областями знаний, многообразие подходов к ее решению.

Проекты — хорошая база для начала обучения школьников работе с научной (точнее, научно-популярной) литературой. Исследуемые в описываемых проектах вопросы не только представляют учебный интерес, но и активно изучаются (в более сложных вариантах) современной наукой. Это относится к фракталам, перколяции, бильярдам, самообучающимся алгоритмам. Они хорошо описаны в научных и научно-популярных статьях, в том числе и в легко доступных. Мы стараемся указать при описании проекта список литературы, способной принести пользу при работе над ним.

Отчет представляет собой заключительную стадию работы над проектом. Он должен включать в себя описание постановки задачи, исследуемые вопросы и полученные в ходе работы ответы на них, тексты программ и образцы диалога с ними, список использованной литературы. Полезно включить в отчет и вопросы, оставшиеся неизученными, пояснив причины этого. Составление отчета на компьютере в процессе работы может принести дополнительную пользу.

Здесь приводятся не все проекты, разрабатывавшиеся в летней школе. Некоторые из них слишком громоздки, другие требуют специального оборудования или программного обеспечения, третьи трудно четко сформулировать. Хотя описания проектов даны весьма сжато, мы надеемся, что они позволяют ухватить суть дела. По каждому проекту указано, кем он был привнесен в школу. Большинство проектов выполнимо на любом языке

программирования на любом из персональных компьютеров. Специальные требования будут оговариваться особо. В случае возникновения у читателей вопросов по тем или иным проектам авторы с удовольствием ответят на них или свяжут читателя с автором проекта.

I. Двусторонние игры

Компьютер предоставляет широкие возможности для реализации игровых программ. Мы здесь ограничимся играми двух партнеров, в которых игроки ходят поочередно (типа шахмат, шашек и т. д.). Многие из таких игр вполне поддаются реализации сильными школьниками. Подобные программы интересны возможностью для школьников опробовать различные алгоритмы игры, сравнить их между собой, организовать турнир с участием программ и людей.

Игровой монитор (Л. В. Воробьева, С. К. Ландо). Многие функции в игре двух партнеров с поочередными ходами являются общими для всех игр. Их целесообразно объединить в игровом мониторе. Монитор должен обеспечивать следующие возможности:

добавление новых игр;

добавление новых алгоритмов;

выбор игры;

игра двух партнеров, каждый из которых может быть человеком или машинным алгоритмом;

ведение протокола;

обеспечение возможности устройства турниров (например, многократной игры двух алгоритмов между собой);

контроль времени.

Задача была упрощена ввиду жестко ограниченного времени в лагере. Монитор был написан преподавателем. Однако, на наш взгляд, это подходящая задача для 2—3 школьников на 6—8 месяцев.

Монитор был реализован на Бейсике для ЭВМ IBM PC, но для него подходит любой другой язык и любая другая машина.

Монитор должен быть насыщен конкретными играми. Впрочем, каждая игра может быть реализована и вне связи с монитором.

При реализации конкретных игр программа, реализующая игру, должна изображать игровое поле, позволять вводить очередной ход, отображать сделанный ход на игровом поле, проверять допустимость хода, сообщать результат игры.

Содержательные игры, допускающие простую реализацию, встречаются не так уж часто. Приведем примеры реализованных игр.

Определители (см.: Квант. 1978. № 8) (Л. В. Воробьева, С. К. Ландо). Игровое поле имеет вид квадрата 3×3. Двое по очереди ставят в свободную клетку квадрата одно из чисел от 1 до 9 (за игру первый игрок делает пять ходов, второй — четыре). Число, однажды поставленное, ставить второй раз нельзя. Если определитель полученной матрицы положителен, то выиграл первый игрок, в противном случае — второй.

Квадратобоязнь (см.: Гарднер М. Математические головоломки и развлечения. М.: Мир, 1971, с. 462) (Л. В. Воробьева, С. К. Ландо). Игровое поле — квадрат 7×7 (начинать можно и с квадрата 3×3). Игроки по очереди ставят в свободные клетки поля красные (первый) и белые (второй) фишки. Проигрывает тот, после хода которого четыре фишки его цвета оказались в вершинах некоторого квадрата (стороны которого необязательно параллельны сторонам игрового поля).

Кооперация (А. Л. Тоом). Эта игра интересна тем, что в ней выигрыш одного игрока не обязательно означает проигрыш другого.

Два игрока делают ходы одновременно. Каждый ход состоит в том, что игрок называет цифру 1 или 2. Если оба игрока назвали 1, то они выбрали различные действия, и назвавший 1 получает 3 очка, а назвавший 2 — теряет 3 очка. По сути дела каждая партия состоит из одного хода каждого из соперников, и интересно учитывать выигрыш после большого числа партий.

Алгоритмы для игровых программ. Эта часть игровых программ наиболее содержательна с математической (программистской) точки зрения. Алгоритмы могут быть очень просты (например, делать совершенно случайный ход; даже это происходит впечатление «разумного»). Вообще, введение случайности в любой алгоритм разнообразит его «поведение» и не позволяет противнику выигрывать за счет единожды найденной удачной комбинации ходов. Конечно, на этот счет имеется развитая математическая теория (см.: Адельсон-Вельский Г. М., Арлазаров В. Л., Донской М. В. Программирование игр. М., Наука, 1978), но и простое экспериментирование без теоретической подоплеки может оказаться весьма полезным.

Естественно, любой алгоритм может использовать историю игры — предыдущие ходы и партии.

II. Математический эксперимент

Перколяция (просачивание) (П. А. Кучмент). Квадрат 51×51 разбит вертикальными и горизонтальными отрезками на 2601 одинаковый квадратик 1×1. На пересечении отрезков находятся узлы. Удаление узла — это удаление всех входящих в него отрезков

длины 1. Некоторые узлы (в количестве N штук), выбранные случайным образом, удалены.

1. Может ли «вода протечь» с верхней стороны квадрата на нижнюю, т. е. существует ли связный путь из единичных отрезков, не лежащих на сторонах квадрата, один конец которого лежит на верхней стороне, другой — на нижней?

2. При каких N существование протекания более вероятно, чем его отсутствие? Если при данном N эти события равновероятны, т. е. половина решеток с N удаленными узлами дает протекание, а половина — его отсутствие, то число $N/2601$ называется коэффициентом перколяции.

3. Ясно, что значение стороны квадрата 51 выбрано случайно. Попробуйте поменять это значение и посмотреть, что случится с N.

4. Заменим квадрат на прямоугольник, одна сторона которого фиксирована, а другая меняется. Как зависит коэффициент перколяции от длины второй стороны?

Наличие графического дисплея облегчает выполнение этого проекта.

Электрическая перколяция (В. Фишман). Близкий проект был реализован на кафедре физики в школе-89.

В квадрате 10×10, расчерченном на клетки, каждое ребро с вероятностью P становится ребром с сопротивлением r и с вероятностью $Q=1-P$ — ребром с сопротивлением $R=10r$.

1. Вычислите сопротивление между противоположными углами квадрата. Для этого нужно решить большую систему линейных уравнений.

2. Вычисленное в предыдущем пункте сопротивление может быть маленьким (если есть «протекание тока» по ребрам с сопротивлением r) или большим (в противном случае). Найдите пороговое значение вероятности P, при котором маленькое и большое значение сопротивления встречаются в половине случаев.

3. Попробуйте поменять число сопротивлений, заменить квадрат прямоугольником (или другой фигурой), изменить отношение сопротивлений R и r.

Перколяция короля и ладьи на шахматной доске (Д. А. Троценко). В сборнике Г. Штейнгауза «Сто задач» приведена как нерешенная следующая задача. Некоторые клетки шахматной доски покрашены в красный, остальные — в синий цвета. Доказать, что либо ладья может перейти с первой горизонтали на восьмую по синим клеткам, либо король с вертикали a на вертикаль h — по красным. Преподаватель школы-89 Д. А. Троценко доказал это утверждение.

1. На красную доску брошено несколько синих клеток. Найти путь ладьи снизу вверх по синим клеткам или путь короля слева направо по красным.

2. Найти пороговое значение перколяции: при каком числе синих клеток вероятность «протекания» ладьи равна 1/2.

3. Исследовать доски с другими размерами (в том числе прямоугольные).

Аналоговое извлечение корня (Ю. Р. Романовский). Период математического маятника длины L равен

$$T=2\pi\sqrt{L/g},$$

где g — гравитационная постоянная. Поэтому квадратный корень из числа L/g можно вычислять по формуле

$$\sqrt{L/g}=\frac{T}{2\pi}.$$

С другой стороны, движение математического маятника описывается дифференциальным уравнением

$$\begin{cases}\dot{X}_1=X_2,\\ \dot{X}_2=-\dfrac{g}{L}X_1.\end{cases}$$

Находя решение этого уравнения методом Эйлера, мы тем самым найдем период, а значит и $\sqrt{L/g}$. Такой способ вычисления квадратного корня отнюдь не экзотика, а вполне реальный метод. Сравните эффективность этого метода с другими алгоритмами вычисления квадратного корня.

Решение квадратного уравнения (С. К. Ландо). Рассмотрим квадратное уравнение

$$AX^2+BX+C=0.$$

Перепишем его в виде

$$X=-\frac{BX+C}{AX}.$$

Положим теперь

$$X_{n+1}=-\frac{BX_n+C}{AX}.$$

Выбрав произвольное значение $X_0\neq0$, получаем последовательность X_0, X_1, ...

1. Проверьте, что эта последовательность сходится к корню уравнения.

2. К какому из двух корней сходится эта последовательность? Как это зависит от выбора значений X_0?

3. Что происходит с последовательностью, если у исходного уравнения нет вещественных корней?

4. Как ведет себя последовательность, если мы начнем ее с комплексного числа?

5. Сравните этот способ приближенного решения квадратного уравнения с другими, известными вам.

Эллиптический бильярд (см.: Гарднер М. Математические досуги. М., 1972, гл. 2; Квант. 1989. № 3) (С. К. Ландо, Г. Б. Шабат).

Точечный шарик равномерно и прямолинейно движется внутри эллипса

$$\frac{x^2}{A^2}+\frac{y^2}{B^2}=1,$$

отражаясь от него по обычному закону отражения. Это означает, что равны угол падения и угол отражения, измеряемые относительно касательной к эллипсу в точке падения. Для удобства наблюдения след шарика должен оставаться на экране. Наблюдая движение шарика, исследовать:

1) как будет выглядеть траектория шарика, пущенного из фокуса эллипса;

2) как выглядит траектория, пересекающая отрезок между двумя фокусами; не пересекающая этого отрезка;

3) по теореме Биркгофа в этом биллиарде существуют замкнутые траектории из любого числа звеньев. Попробуйте отыскать траектории из двух (это совсем просто), трех, четырех, пяти звеньев.

Программа должна позволять задавать начальное положение и направление движения шарика, а также полуоси эллипса (или одну из этих полуосей). Для выполнения этого проекта требуется наличие графического дисплея.

3К+1 (П. А. Кучмент). Очередной член последовательности, состоящей из натуральных чисел, определяется правилом

$$A_n=\begin{cases}A_{n-1}/2, & \text{если } A_{n-1} \text{ четно},\\ 3A_{n-1}+1, & \text{если } A_{n-1} \text{ нечетно}.\end{cases}$$

Начальное значение A_0 выбирается произвольно.

Верно ли, что всякая последовательность начиная с некоторого момента имеет вид 1, 4, 2, 1, 4, 2, 1, 4, 2, ...? Предусмотрите возможность работы с длинными значениями, превосходящими возможности машинного представления.

Фрактальные множества, или фракталы,— это множества сложной структуры на плоскости или в пространстве. Они характеризуются большой изрезанностью. Типичные примеры таких множеств — морозные узоры на стекле и береговая линия. Как ни удивительно, такие сложные объекты легко моделируются на компьютере, позволяя получать изображения, не уступающие по красоте естественным. Для работы с фракталами требуется графический дисплей, наиболее красивые рисунки получаются на цветном дисплее.

Множества Жюлиа и Мандельброта (см.: В мире науки. 1988. № 1; 1989. № 4) (М. В. Нестерова, И. Г. Щербак).

Множество Мандельброта — обитатель комплексной плоскости, т. е. обычной плоскости, каждая точка которой характеризуется двумя координатными значениями. Каждая точка комплексной области представляется числом вида $a+bi$, где $i^2=-1$.

Основная формула, порождающая множества Мандельброта и родственные ему множества Жюлиа:

$$Z_{n+1}=Z_n^2+C,$$

где Z_k и C — комплексные числа. Число Z_n возводится в квадрат и C прибавляется к результату согласно правилам умножения и сложения комплексных чисел. Формула как бы оживает, если начать последовательно вычислять значения сумм, подставляя в формулу каждый раз значение Z_n, полученное на предыдущем шаге. Получающаяся в результате этого итерационного процесса последовательность комплексных чисел образует причудливый узор на комплексной плоскости. Если представить полученную последовательность как блуждания исходной точки, то некоторые комплексные числа уходят в бесконечность, другие навсегда ограничены в своем движении определенной областью, имеющей сложную форму. Эта область имеет фрактальные стены.

Выберем число Z_0 равным нулю и будем брать различные значения C из определенной области комплексной плоскости. Если точке удается бежать, то окрашиваем ее в белый цвет, в противном случае — в черный. Стены области, из которой не удается бежать, принимают форму множества Мандельброта. Если окрашивать точки в зависимости от скорости убегания, то изображение получится еще красивее.

В случае другого выбора значения $Z_0\neq 0$ мы получим деформированную версию множества Мандельброта.

Если следовать противоположному правилу, когда значение C фиксировано, а Z_0 играет роль исходной точки, получающееся в результате итерационного процесса множество уже отличается от множества Мандельброта. Его граница называется множеством Жюлиа. Множество Жюлиа не единственно, на самом деле имеется целое семейство таких множеств: для каждого фиксированного значения C в формуле мы получаем свое множество Жюлиа.

Теперь можно приступить к написанию программ и исследованию с их помощью получающихся множеств. Естественно, что итерационный процесс нельзя продолжать до бесконечности, поэтому достаточно принять граничное число шагов равным 100 и ввести следующий критерий: если за 100 шагов мо-

дуль итерируемой переменной достигает 2, то ей суждено убежать в бесконечность, в противном случае она остается внутри области.

Можно выбирать различные правила для окраски точек, это зависит от количества цветов на экране вашего компьютера. Чем больше цветов вы используете, тем причудливее и интереснее становится изображение, хотя само множество Мандельброта остается неизменным.

Вопросы для исследования.

1. Каким образом зависит связность и несвязность соответствующего множества Жюлиа от принадлежности начальной точки множеству Мандельброта?

2. Как выглядят деформированные множества Мандельброта?

3. Какие правила раскраски точек дают наиболее интересные изображения?

Поскольку каждое изображение содержит лишь конечную выборку точек в какой-то области множества Мандельброта, оно никогда не сможет отобразить всех деталей реальной картины. Однако, сужая диапазоны изменения C, можно повысить степень «увеличения». В действительности множество Мандельброта имеет массу сложных деталей на всех уровнях увеличения, что определяется его фрактальной структурой. Поэтому желательно, чтобы в программе предусматривалась возможность изменения области варьирования C для рассматривания более крупных планов множеств.

Случайные фрактальные скопления (В мире науки. 1989. № 2; Успехи физических наук. 1989. № 2) (С. К. Ландо). В начальный момент на экране изображены окружность и точка в центре этой окружности. Точка является зародышем — первым элементом интересующего нас множества. Шаг процедуры состоит в следующем. Случайно выбранная точка на окружности начинает случайное блуждание: она с одинаковой вероятностью 1/4 делает шаги вверх, вниз, влево или вправо. Если одна из четырех соседних с ней точек оказывается принадлежащей к нашему множеству, то блуждающая точка «прилипает» (добавляется к множеству), а с окружности отправляется в путь следующая точка. Если же точка в своих блужданиях вышла за пределы окружности, она считается «потерянной» и с окружности запускается следующая точка.

Через некоторое время вы получите множество, похожее на кучу железных опилок на магните.

1. Попробуйте применить различную раскраску точек в зависимости от того, на котором шаге произошло «прилипание».

2. Поменяйте правило случайного блуждания. Например, разрешите переходы по диагонали. Или отправляйте точку с треугольника, с квадрата. Или замените квадратную решетку шестиугольной.

3. Радиус окружности 50 (точек) — хорошая база для начала исследования. Попробуйте и другие его значения.

Самый интересный с математической точки зрения вопрос — это вычисление размерности такого фрактала. Такое множество более плотно, чем отрезок (размерность которого равна 1), но разреженнее круга (размерность которого равна 2). Для исследования этого вопроса надо поработать с литературой.

Выполнение данного проекта требует наличия графического дисплея.

Светофор (А. Г. Кулаков). На окружности расположены N точек, каждая окрашена в один из трех цветов: красный, желтый или зеленый. Шаг процедуры состоит в одновременном изменении цветов всех точек. На каждом шаге цвет каждой точки меняется по следующему правилу:

1) если две точки, соседние с данной, имеют одинаковый цвет, то точка перекрашивается в этот цвет;

2) если две точки, соседние с данной, имеют разные цвета, то точка перекрашивается в третий цвет.

Ясно, что если в начальный момент все точки окрашены в один цвет, то перекрашивания не происходит.

1. Существуют ли другие конфигурации, в которых ничего не меняется? При каких N это возможно?

2. Исследуйте поведение различных конфигураций при разных N. При каких N существуют циклы длины 2 (т. е. конфигурация повторяет саму себя через каждые два шага)? Циклы длины 3, 4, 5?

3. Перечислите все длины циклов при заданном N.

4. Попробуйте обобщить задачу на

а) большее число цветов;

б) на другие расположения точек, например точки на торе.

Описанное преобразование хорошо выражается в терминах линейной алгебры над полем из трех элементов.

Этот проект можно выполнить и на алфавитно-цифровом дисплее, представляя каждый цвет своим символом.

Свердловские задачи

Условие данной задачи, посвященной языку PROLAN/F, см.: ИНФО. 1990. № 2.

Решения

Поскольку в языке нет «переборных» циклов, действия должны реализовываться с помощью рекурсий. Кроме того, заметим, что при логических x и y запись «x — y» эквивалентна «и(x, y) — ДА, и(x, не(y) — НЕТ».

```
а) стррав?(x,y)=( пусто?(x) - пусто?(y),
                  пусто?(y) - НЕТ,
симрав?(консим(x),консим(y)) -
          стррав?(начстр(x),начстр(y)),
                  ИНАЧЕ - НЕТ ).

б) хвост?(x,y)=( пусто?(x) - ДА,
                  пусто?(y) - НЕТ,
симрав?(консим(x),консим(y)) -
            хвост(начстр(x),начстр(y)),
                  ИНАЧЕ - НЕТ ),
```


(Является ли строка x подстрокой строки y, завершающей последнюю.)

```
стрпринадл?(x,y)=( хвост?(x,y) - ДА,
          стрпринадл?(x,начстр(y)) - ДА,
                  ИНАЧЕ - НЕТ ).

в) удалконстр(x,y)=
            ( или(пусто?(x),пусто?(y) - y,
симрав?(консим(x),консим(y)) -
        удалконстр(начстр(x),начстр(y)),
                              ИНАЧЕ - y ),
```

(Строка, получающаяся удалением у строки y последних ее символов, образующих строку x).

```
припперев(x,y)=( пусто?(y) - x,
                 ИНАЧЕ -
припперев(припконсим(x,консим(y)),
                          начстр(y))),
```

(Строка — результат приписывания к строке x строки, полученной перестановкой символов строки y в обратном порядке.)

```
перевор(x)=( ДА - припперев('',x),
             ИНАЧЕ - '' ),

г) удалначстр(x,y)=( ДА -
     удалконстр(перевор(x),перевор(y)),
             ИНАЧЕ - '' ),
```

(Строка, получаемая удалением у строки y подстроки из начальных символов, образующих строку x).

```
припстр(x,y)=( ДА -
               припперев(x,перевор(y)),
               ИНАЧЕ - '' ),
```

(Строка, получаемая конкатенацией строк x и y.)

```
начало_с(x,y)=( стрпринадл?(x,y) -
                начало_с(x,начстр(y)),
              ИНАЧЕ - y),
```

(Состоящая из начальных символов строки y строка наименьшей длины из содержащих подстроку x.)

```
начало_без(x,y)=(ДА -
          удалконстр(начало_с(x,y),y),
                ИНАЧЕ - '' ),
```

(Подстрока наибольшей длины из начальных символов y, не содержащая ни одного символа вхождения x.)

```
конец_без(x,y)=( ДА -
        удалначстр(начало_без(x,y),y),
                ИНАЧЕ - '' ),
```

(То же, что и для предыдущей функции, только из конечных символов y.)

```
подстан_лев(x,y,z)=
        ( стрприн?(y,z) -
          припстр(начало_без(y,z),
           припстр(x,конец_без(y,z))),
                ИНАЧЕ - z ),

подстан_всех(x,y,z)=
        ( стрприн?(y,z) -
          припстр(начало_без(y,z),
            припстр(x,подстан_всех
                (x,y,конец_без(y,z))),
              ИНАЧЕ - z ).
```

Доказательства

Доказательства того, что приведенные записи дают решения поставленных задач, целесообразно проводить методом математической индукции. Приведем пример доказательства (с опущенными подробностями) для решения пункта а.

Рассмотрим сначала случай, когда самая короткая из сравниваемых строк имеет нулевую длину (пустая). Такие строки будут равны лишь в том случае, если они обе пустые. Из приведенного нами решения — программы для «стррав?» — следует, что оно удовлетворяет этому условию.

Предположим теперь, что наше решение является верным для случая, когда самая короткая из сравниваемых строк имеет длину $N>0$.

Рассмотрим пару строк x и y, где кратчайшая имеет длину $N+1$. Из определения равенства строк следует, что для равенства непустых строк необходимо и достаточно, чтобы были равны их последние символы и были равны подстроки, остающиеся при отбрасывании последних символов. Поскольку для упомянутых подстрок кратчайшая имеет длину N и поэтому в силу индуктивного предположения функция стррав(начстр(x),

начстр(y)) дает верный ответ, то приведенная программа дает для строк, наикратчайшая из которых имеет длину N+1, верный ответ.

Таким образом, доказано, что программа дает верное решение для аргументов-строк, кратчайшая из которых имеет любую длину N⩾0, т. е. для любых аргументов-строк.

Аналогично можно доказать правильность решений, приведенных для других пунктов,— обязательно проделайте это. Ведь решение любой задачи (по математике, информатике, программированию или чему угодно другому), не снабженное доказательством или хотя бы убедительным обоснованием,— не решение, а всего лишь гипотеза. Которая может быть и ошибочной...

Что же это за странный язык?

PROLAN/F не похож по своей идеологии на привычные языки — Паскаль, Рапиру, Бейсик, «Ершол» и т. п. Действительно, перечисленные языки относятся к обширному семейству алгоритмических (еще говорят: процедурных, операторных, фон-неймановских и даже командно-директивных) языков. Хотя это семейство и обширно, но есть и другие концепции построения языков информатики. Например, можно выделить еще функциональные и логические подходы. Так вот, PROLAN/F относится к функциональным языкам, наиболее известными представителями которых являются разработанные в начале 60-х гг. языки LISP и REFAL, широко используемые при решении определенного класса задач (например, машинного перевода текстов с русского языка на английский). Аналогичные элементы функционального подхода включены и в язык логического программирования PROLOG.

А теперь непростое задание. Разработайте для какой-либо ЭВМ работающий транслятор с языка PROLAN/F, позволяющий выполнять написанные на нем программы. Напоминаю, что система должна воспринимать как новые определения функций, так и запросы вида

```
подстан_всех('курица','собака',
                    'У попа была собака')
```

на которые должен последовать вывод системой на экран значения соответствующей функции; в данном примере на экран выведется:

```
'У попа была курица'
```

Для повышения компактности программ имеет смысл ввести в систему возможность использования макрообозначений «[» и «]»:

«[» — эквивалент «(ДА —»;

«]» — эквивалент «,ИНАЧЕ —"».

Тогда будет можно писать, например,

```
или(x,y)=[ не( и( не(x), не(y) ) ) ],
начало_без(x,y)=
        -[ удалконстр(начало_с(x,y),y) ]
```


Если вы хотите сделать систему не просто для тренировки, а чтобы ею могли пользоваться как системой программирования, не обойдите вниманием то, насколько удобно будет вводить тексты программ, обеспечьте возможность хранения во внешней памяти библиотеки накопленных определений функций. Хорошо, если система будет обеспечивать некоторые режимы отладки программ.

Заинтересовавшиеся описанным языком могут приобрести систему программирования на базе PROLAN/F. Наша система содержит экранный редактор текстов программ, компилятор расширенного PROLAN/F (язык дополнен, в частности, числовыми данными и арифметическими операциями); тексты программ можно сохранять в библиотеке. Имеется трассировщик исполнения программы.

Разработаны версии системы для IBM PC, «Ямахи», «Электроники-85», «Электроники БК-0010».

С вопросами о приобретении обращаться по адресу: 620219, Свердловск, ГСП-384, В. В. Прохорову.

В. ПРОХОРОВ

И. ПЕРЕХОД, В. КАСАТКИН

Обобщенный счетчик

Введем понятие обобщенного счетчика, который будет пригоден для генерации объектов в различных числовых системах. Покажем, что с его помощью можно формировать все типы классических комбинаторных объектов. Подробно рассмотрим формирование размещений и сочетаний с помощью ЭВМ.

Обобщенный счетчик — это абстрактное устройство, содержащее регистр X (векторную переменную)

$$X=(x[k],\ x[k-1],\ ...,\ x[1]),$$

каждая позиция которого $X[i]$ может содержать элемент (цифру) множества $N=\{0,\ 1,\ 2,\ ...,\ n-1\}$ и в котором текущее значение X изменяется в соответствии с процедурой обобщенного следования Generalized — Succ, вырабатывающей очередное значение X.

Будем считать, что функции $\inf(i)$ и $\sup(i)$, поставляющие границы соответствующих разрядов $x[i]$, определены для всех i от 1 до k:

$$\inf(i)\leqslant x[i]\leqslant \sup\ (i).$$

В этом алгоритме c — номер разряда, в котором завершился перенос и значение которого $x[c]$ увеличивается на единицу. Во всех позициях справа от разряда завершения переноса старшие цифры, например девятки, заменяются на младшие, например ноли.

```
PROCEDURE Generalized_Succ:
 BEGIN
   WHILE i<=n and x[i]=Sup(i)
   DO i:=i+1:
   c:=i:
   IF c<=n THEN x[c]:=x[c]+1
   FOR j:=c-1 DOWNTO 1 x[j]:=Inf(j)
 END
```

Обобщающий характер алгоритму придает использование границ. При этом обобщаются также понятия минимального и максимального значений счетчика:

$$X_{min}=(\inf(k),\ ...,\ \inf(1)),$$
$$X_{max}=(\sup(k),\ ...,\ \sup(1)).$$

Если $\inf(i)=0$ для всех i, то имеем обычный ноль.

Перенос, как всегда, выполняется при превышении $\sup(i)$, при этом следует учитывать, что допускается динамическое переопределение границ в процессе счета.

Если выполнять алгоритм следования в цикле до тех пор, пока перенос завершается в пределах разрядной сетки регистра, то получим все элементы (комбинации) множества Z комбинаторных объектов, заданного с помощью границ $\inf(i)$, $\sup(i)$.

Перебор всех значений множества Z можно записать в виде

```
REPEAT
  i:=1
  Ceneralized_Succ:
  Process(X):
UNTIL i>n
```

Таким образом, алгоритм обобщенного следования упорядочивает объекты цифровой системы, что позволяет их перенумеровать.

Продемонстрируем использование обобщенного счетчика для построения конкретных счетчиков, а именно:

генератора размещений из n элементов по k;

генератора сочетаний из n элементов по k;

генератора сочетаний из n элементов по k с повторениями.

1. Построение в порядке возрастания всех k-разрядных чисел из цифр множества N (размещения с повторениями из n элементов по k в лексикографическом порядке).

Решив задачу, получим n^k чисел со значениями от 0 до n^{k-1} в системе счисления с основанием $Q=n$.

$$\inf(i)=0,$$
$$\sup(i)=n-1.$$

для всех i.

В данном случае процедура следования реализует хорошо известные правила прибавления единицы к числу в системе счисления с натуральным основанием.

```
10 'ГЕНЕРАТОР РАЗМЕЩЕНИЙ ИЗ N ПО K
20 '          С ПОВТОРЕНИЯМИ
30 INPUT "N,K="; N,K: DIM X(K+1)
40 SUP=N-1: INF=0: 'ГРАНИЦЫ X[J]
50 FOR J=1 TO K: X(J)=INF: NEXT J
60 WHILE I<=K: 'ОСНОВНОЙ ЦИКЛ
70 'ВЫВОД СОДЕРЖИМОГО РЕГИСТРА СЧЕТЧИКА
80 FOR J=K TO 1 STEP -1: PRINT X(J):
                        NEXT J: R=R+1: PRINT
90 'СЛЕДОВАНИЕ
100 I=1: WHILE I<=K AND X(I)=SUP:
                              I=I+1: WEND
110 IF I<=K THEN X(I)=X(I)+1
120 FOR J=1 TO I-1: X(J)=INF: NEXT J
130 WEND: PRINT "R="; R: END
```

2. Генератор сочетаний из n элементов по k.

Сочетания из n элементов по k с повторениями порождаются счетчиком при

$$\inf(i)=\begin{cases}0, & \text{если } i\geqslant c,\\ x[c], & \text{если } i<0,\end{cases}$$

$\sup(i)=n-1$.

$X_{min}=(0, ..., 0)$, т. е. до начала счета $c=0$.

```
10 'ГЕНЕРАТОР СОЧЕТАНИЙ ИЗ N ПО K
15 '       С ПОВТОРЕНИЯМИ
20 INPUT "N,K="; N,K: DIM X(K+1):
   SUP=N-1: INF=0: 'ГРАНИЦЫ X[J]
30 FOR J=1 TO K: X(J)=INF: NEXT J
40 WHILE I<=K: 'ОСНОВНОЙ ЦИКЛ
50 'ВЫВОД ОЧЕРЕДНОГО ЗНАЧЕНИЯ
60 FOR J=K TO 1 STEP -1: PRINT X(J);:
                    NEXT J: R=R+1: PRINT
70 'СЛЕДОВАНИЕ
80 I=1: WHILE I<=K AND X(I)=SUP:
                                I=I+1: WEND
90 IF I<=K THEN X(I)=X(I)+1: INF=X(I)
100 FOR J=I-1 TO 1 STEP -1:
                            X(J)=INF: NEXT J
110 WEND: PRINT "R="; R: END
```

3. Сочетания из n элементов по k без повторений порождаются счетчиком при

$$\inf(i)=\begin{cases} k-i, & i\geqslant c, \\ x[i+1]+1, & i<c, \end{cases}$$

$\sup(i)=n-i$.

```
10 'ГЕНЕРАТОР СОЧЕТАНИЙ ИЗ N ПО K
20 INPUT "N,K=": N,K: DIM X(K+1):
                    DIM INF(K+1), SUP(K+1)
30 FOR J=1 TO K: SUP(J)=N-J:
       INF(J)=K-J: X(J)=INF(J): NEXT J
40 WHILE I<=K: 'ОСНОВНОЙ ЦИКЛ
50 'ВЫВОД ОЧЕРЕДНОГО ЗНАЧЕНИЯ
60 FOR J=K TO 1 STEP -1: PRINT X(J);:
                    NEXT J: R=R+1: PRINT
70 'СЛЕДОВАНИЕ
80 I=1: WHILE I<=K AND X(I)=SUP(I):
                                I=I+1: WEND
90 IF I<=K THEN X(I)=X(I)+1
100 FOR J=I-1 TO 1 STEP -1:
   INF(J)=X(J+1)+1: X(J)=INF(J): NEXT J
110 WEND: PRINT "R="; R: END
```

Все приведенные алгоритмы имеют одинаковую структуру, отличаясь друг от друга только заданием верхних и нижних границ.

В этом еще раз обнаруживается единство подхода, в основе которого лежит понятие о цифровой системе.

Ю. ЗАЛЬЦМАН, В. МИХАЙЛОВ

БК-0010 — речь и слух: возможности и реальность

Начнем с конца: синтезатор речи с неограниченным словарем для микро-ЭВМ БК-0010 прекрасно работает с 1989 г. Он может выводить в виде речевой информации (на русском языке) произвольный текст, выдаваемый микро-ЭВМ на экран, причем форма представления текста в памяти ЭВМ — самая обыкновенная, стандартный код КОИ-8, и слова — по обычным правилам грамматики, а не особая, «фонетическая», запись, как во многих известных системах, в том числе зарубежных. Каким переделкам для этого подверглась сама ЭВМ? Никаким. Какие внешние дополнительные устройства требуются для воспроизведения речи? Никакие. Сколько места в памяти занимает синтезатор? 8К байт.

Зачем понадобилось это вступление? Дело в том, что, когда мы начинали работу по созданию синтезатора речи для БК-0010, не было ни одного (!) человека, включая и нас самих, который после изложения требований к синтезатору сказал бы, что это возможно. Потому лучше сразу уверить читателя в том, что все дальнейшие рассуждения не являются голой теорией, а блестяще подтвердились на практике (чем не могут похвастаться многие и многие разработки в этой области).

Итак, с чего все началось? Казахскому обществу слепых (КОСу) потребовалось оборудование, с которым мог бы работать слепой пользователь: набирать на клавиатуре текст, редактировать его и затем распечатывать как специальным шрифтом для слепых (шрифт Брайля), так и обычным шрифтом. Что же тут такого, скажете вы? Существуют специальные пишущие машинки для слепых (действительно, существуют) и даже специальные ЭВМ (да, но только не у нас). Проблема состояла в том, чтобы слепой мог подготовить текст, а не просто его распечатать. Такое оборудование, существующее за рубежом и разрабатывающееся у нас, базируется в основном на преобразовании текста в динамическую (т. е. постоянно меняющуюся, «бегущую» по тексту) строку символов Брайля — матриц из 6 или 8 точек (оборудование класса «Версобрайль»). Определенное сочетание рельефных точек позволяет обозначить все русские и латинские буквы, цифры и даже нотные знаки и спецсимволы.

Однако шрифт Брайля слепой читает обычно двумя руками, и исправлять ошибки (ра-

ботать на клавиатуре) ему просто нечем. Нужны невероятные память и воображение, чтобы сопоставить, не видя текста, символы на бегущей строке и те изменения, которые нужно внести в текст на клавиатуре (к слову сказать, установка типа «Версобрайль» тоже была нами сконструирована и входила в комплект оборудования, но не о ней речь).

Вот тут и возникла мысль о синтезаторе речи. Если слепой сможет слышать, что выдает компьютер на экран, то внести в текст изменения ему будет гораздо легче — руки свободны! Да и пульт для работы тоже брайлевский — всего восемь клавиш.

До начала работы мы, авторы разработки, не знали о синтезе речи почти ничего. Приходилось, конечно, слышать в научно-популярных телепередачах фразы, произнесенные необычным, «машинным» голосом, и видеть за спиной ведущего машинные залы, в которых стояли электронные «чудища», этот голос производящие. Но сейчас эпоха персоналок, электронные «динозавры» вымирают... Какая ЭВМ сможет нам помочь в реализации идеи? Выбор определялся крайне скромным финансированием разработки — компьютеры класса IBC PC отпали поэтому сразу. В то время (да и сейчас мало что изменилось) единственной сколько-нибудь стоящей дешевой машиной был БК-0010. Дешевой — понятно, а почему стоящей? Да потому, что это был единственный 16-разрядный компьютер среди бытовых, все остальные — 8-разрядные. Если кто-то считает, что это не так важно, то с ним больше говорить не о чем. Ведь что было задумано? Задача на пределе возможностей средней или даже большой ЭВМ (так нам тогда представлялось). Итак, мы начали работу, не надеясь, впрочем, на полный успех.

Известно немало методов синтеза речи: спектрально-полосовой, гармонический, корреляционный, формантный, КЛП-синтез и др. Все они нам не подходили — как по причине требующихся большой памяти и быстродействия, так и потому, что их использование почти обязательно требует наличия специальных внешних устройств — цифроаналоговых преобразователей (ЦАП), формантных фильтров, специализированных сопроцессоров и т. п. Мы же были крайне ограничены в средствах (и во времени), а кроме того, хотели сделать программу, которая смогла бы работать на любом БК-0010 без всяких проблем.

Оставался так называемый фонемный цифровой синтез, т. е. «сборка» речи из отдельных звуков (фонем), как из кубиков. Прочтя несколько высокоинтеллектуальных теоретических работ, мы испугались — оказывается, в любом языке фонем гораздо больше, чем букв в алфавите! Фонем больше сотни! Да еще «стыковать» их друг с другом нужно по особым правилам, количество которых достигает несколько сотен... Куда тут с 16К байтами ОЗУ! Но после жарких споров, за которые авторы монографий и докторских диссертаций, услышь они их, придушили бы нас на месте — так рьяно опровергали мы то, на что они потратили годы труда,— мы все же пришли к выводу, что дело не безнадежно.

Во-первых, русский язык почти фонетический — пишем почти так, как говорим (не то что англичане!). Во-вторых, на БК уже были опыты прямой записи речи в память (как на цифровом магнитофоне), и довольно успешные. В-третьих, мы были уверены, что наши возможности как программистов ограничены только возможностями самого БК-0010 — мы знали его идеально. Вот это, да еще присущая любому дилетанту уверенность в том, что профессионалы всегда сгущают краски, и дало нам преимущество.

Начали мы с того, что сделали простейший «фонетический редактор» — он позволял «загрузить» в память БК (в экранное ОЗУ) речевую информацию с микрофона (продолжительностью 10—12 с), затем «вырезать» из речи отдельные звуки и «склеить» из них нужное слово. О чудный миг, когда машина произнесла первую фразу, не записанную с микрофона, а «склеенную» из отдельных фонем: «Маша ела кашу»! Мы думали, что задача решена! Но самое сложное оказалось впереди...

Поскольку мы смело отвергли опыт синтеза речи, накопленный «мировой общественностью» (так как опыт этот однозначно говорил, что создать на БК синтезатор речи невозможно), нам пришлось накапливать свой. Мы очень скоро убедились, что буква звучит по-разному в зависимости от того, где она стоит — в начале, середине, конце слова или отдельно. Кроме того, соседние буквы, как «выяснилось» (все это давно и хорошо известно), очень сильно влияют на ее звучание! К этому времени мы имели уже очень неплохой фонетический редактор, позволяющий создать настоящий «алфавит фонем» и экспериментировать с ним, синтезируя слова и фразы просто по коду КОИ-8 символов.

Начался каторжный труд создания «словаря». Мы по-прежнему надеялись подобрать такое звучание фонем, соответствующих буквам, которое будет компромиссным и удовлетворит все фонетические тонкости при числе фонем, равном 30 (твердый и мягкий знаки не в счет, а буква «ё» на компьютерах отсутствует). Мы отлично понимали, что с большим числом фонем БК просто

не справится! А ведь нужно было еще оставить в памяти место для других программ...

Цель эта была достигнута — мы подобрали такие фонемы, которые звучали удовлетворительно, в каком бы окружении ни стояла буква.

Правда, решить задачу «в лоб» не удалось: оказалось недостаточным просто «вырезать» из речи тот или иной звук или специально его «наговорить» (представьте себе, кстати, эту картину: двое программистов сидят у компьютера и орут по очереди в микрофон что-то несусветное, мычат, шипят, поют, а потом лихорадочно манипулируют с клавиатурой и слушают с блаженными улыбками, как компьютер с диким, ни на что не похожим акцентом произносит — в основном популярные скороговорки типа «На дворе трава, на траве дрова»). Литература рекомендует прибегнуть для создания словаря фонем к услугам профессионального диктора; мы обошлись без него, так как убедились, что качество речи при наших скромных возможностях (частота дискретизации порядка 8 кГц) очень мало зависит от наличия или отсутствия косноязычия. Кроме того, какой диктор выдержит три месяца фонетических «упражнений»? Но словарь фонем упорно не давался! Стоило выбрать фонемы, отлично звучащие в составе одного слова, как другое становилось совершенно неразборчивым... Особенно мучили согласные — никак не удавалось наговорить звуки так, чтобы «б» отличалось от «п», а «ж» от «ш». Вместо «в» почему-то слышалось «з», а «о» и «а» звучали, совершенно произвольно «меняясь ролями» в зависимости от окружения, в котором они пребывали... Дальнейший успех определило решающее «открытие»: очень многое зависит от длительности звуков.

Как только в редактор были внесены соответствующие изменения и мы смогли менять длительность и частоту воспроизведения звуков, дело пошло на лад. Мы даже стали получать одни звуки из других — глухие из звонких и наоборот. Но так называемые взрывные — «б», «п», «т» — не давались по-прежнему. И тут пришло второе «открытие» (а может быть, и действительно открытие, без кавычек, так как в литературе ничего подобного не нашлось). Оказывается, предшествующая звуку короткая пауза (мы назвали ее «лидером») придает необходимое «взрывное» звучание. В итоге словарь фонем, удовлетворительно звучащих в любых комбинациях без всяких модификаций и «связок», был создан.

Тут, видимо, следует объяснить, как мы вводили в БК речь. Простейшим путем — через подключенный к компьютеру магнитофон (а позже — специально созданный усилитель с полосовыми фильтрами и интеграторами) и обычный микрофон (позже — высококачественный электретный). На входе «МГ» БК, как известно, включен компаратор напряжения, преобразующий любой звуковой сигнал в двухуровневый логический. Оставалось с определенной частотой считывать этот сигнал со входа и побитно заносить его в ОЗУ (это называется дискретизацией сигнала).

Из известной теоремы Котельникова следует, что для восстановления дискретизированного сигнала без потерь частота дискретизации должна по крайней мере в два раза превышать максимальную частоту обрабатываемого сигнала. Частотный спектр речи очень широк, и, «по Котельникову», следовало выбрать частоту опроса входного сигнала как можно большей. Но словарь фонем не должен был занимать много места, и мы использовали частоту дискретизации около 8 кГц, что обеспечивало восстановление спектральных составляющих с частотой не более 4 кГц.

Тут обнаружилась еще одна проблема. Пока информация пишется по одному биту в 16-разрядное слово БК-0010, все идет нормально — выбранная частота соблюдается, так как в цикле работают однотипные операторы программы. Но затем нужно занести это слово в ОЗУ, модифицировать адрес памяти и начать «набивать» битами новое слово. А при этом образуется пауза, вроде бы ничтожная, около 35 микросекунд, но основательно портящая дело — ведь происходит сбой частоты дискретизации. При воспроизведении звуков при переходе от одного машинного слова к другому возникает такая же пауза. Складываясь, они основательно искажают звучание. Пришлось прибегнуть к специальным ухищрениям, чтобы скомпенсировать паузу при записи звуков — пауза при воспроизведении в одиночестве вреда не причиняла.

Кстати, чем выше частота дискретизации, тем заметнее ее сбои при переходе к следующему машинному слову. Это, кроме ограниченной памяти, также заставило выбрать частоту около 8 кГц — при дальнейшем ее повышении качество речи начинало падать.

Для воспроизведения мы вначале просто подключали к БК магнитофон в режиме «запись», но скоро убедились, что его частотная характеристика с сильным подъемом в области высоких частот, мягко говоря, не идеальна для этой цели. Тогда был использован обычный усилитель низкой частоты с регулировкой тембра (от проигрывателя), и звучание синтезированной речи сразу настолько улучшилось, что мы опять подумали — задача решена.

Когда был создан словарь фонем и мы убе-

дились, что он вполне работоспособен (проверяли так: один набирал на клавиатуре фразу или слово, обычно редкое и неожиданное, а другой на слух старался понять его), во весь рост встал вопрос: как будет выглядеть сам синтезатор речи? Ведь в тексте есть множество «мелких пакостей» типа цифр, ударений, латинских букв, знаков препинания... Русский язык все же не вполне фонетичен; например, редко говорят «чего», но — «чево», произносят «што» вместо «что» и т. д. И та самая пресловутая буква «ё» — давно исключенная из печатного текста, она очень сильно сказывается на произношении! Попробуйте-ка сказать «ёлка» с «е» вместо «ё»!

Как решаются эти проблемы «по-настоящему», мы знали — либо создается свод правил (например, ударений; но каково создать такие правила для русского языка?!), либо словарь, в котором есть все нужные варианты произношения, либо то и другое — свод правил и словарь исключений. Ясно, что на БК это было невозможно. Мы стали экспериментировать и убедились, что речь без ударений почти не теряет в разборчивости; что «ё» в 99 % случаев можно заменить на «о», которая в контексте воспринимается почти нормально... Очевидно, живые люди часто не соблюдают правил произношения, и все давно привыкли к «нелитературным» вариантам типа «чего» и даже не замечаем их.

Кстати, еще раньше мы убедились, что хотя почти невозможно распознать на слух отдельные звуки нашего словаря фонем, но, слитые в слова, они звучат как ни в чем не бывало! Психологи называют этот феномен «целостностью восприятия» (в данном случае звукового образа). Так, рассматривая в лупу газетную репродукцию, мы видим лишь точки, но стоит посмотреть издали — и картина предстает целой.

Итак, мы отказались от ударений и изменения звучания букв в разных словах, пожертвовав правильностью речи синтезатора в пользу его существования, но оставались еще вещи, которые никак нельзя обойти. Мы начали с твердого знака. Открытие, что пауза (почти равная по длительности звуку) прекрасно его заменяет, нас приятно удивило. После этого мы уже смело заменили мягкий знак более короткой паузой, констатировав, что он — всего лишь анахронизм, вроде буквы «ять», благополучно почившей 70 лет назад. Но про знаки препинания и цифры мы, увы, так сказать не могли. Была еще одна сложность: нужно было обеспечить возможность выделения отдельных букв текста на слух, но, как уже говорилось выше, отдельные фонемы словаря не различались — требовался звуковой образ, слово.

Мы вспомнили, как решается такая задача у радиотелефонистов в почти аналогичной ситуации — при плохой разборчивости речи. Слова называют при этом по буквам, используя обычно имена, начинающиеся с той буквы, которую необходимо обозначить, например: А — Алексей, Б — Борис, В — Василий и т. д. Взяв у знакомого радиолюбителя-коротковолновика список «ключевых слов», мы экспериментально его опробовали и заменили некоторые общепринятые слова на другие, которые наш синтезатор более четко проговаривал.

Совсем просто решилась проблема цифр — они всегда называются полностью, когда их коды встречаются в тексте, вот и все: «один», «два», «три» и т. д. От произнесения многоразрядных чисел типа «сто двадцать три» вместо «один, два, три» мы отказались сразу — сложно, да и не всегда нужно; кто, например, произносит номер телефона, как «шестьсот девяносто одна тысяча семьсот девяносто семь»?! Со знаками препинания и спецсимволами мы тоже справились — некоторые должны произноситься всегда полностью, например «плюс», а некоторые — заменяться паузами, например запятая.

Но как быть, если нужно исправить именно эти знаки в тексте? И родилась еще одна ключевая идея — два режима синтеза, слитный и посимвольный, выбираемые пользователем. В слитном слова произносятся так, как мы их читаем (в основном по правилу «один символ — один звук», исключение — цифры); вместо знаков препинания — паузы. В другом же режиме называются все символы кода КОИ-8, включая даже коды управления курсором («вверх», «влево-вверх», «сброс экрана» и т. д.). Латинские символы синтезатор всегда должен называть, как буквы английского алфавита — «эй», «би», «си», «ди» и т. д. Обычно это два звука, которые довольно легко понять на слух. От синтеза английской речи у нас хватило ума отказаться — английский язык очень далек от фонетического, и БК он не по зубам*.

Таким образом, общее представление о структуре синтезатора сложилось, но проблемы оставались. Все это должно было работать в реальном масштабе времени да еще занимать мало места в памяти — вечная проблема! Как тут быть с множеством слов — имен и названий, которые должен выдавать синтезатор вместо кодов КОИ-8? Нашелся отличный выход — синтезатор использует сам себя: все длинные слова и фразы типа «сброс экрана», «точка с запятой» написаны тоже в коде КОИ-8, и синтезатор произносит

* Если же кто-нибудь, не знающий этого, докажет обратное, мы с удовольствием опубликуем рассказ о том, как это происходило.— *Примеч. ред.*

их по звукам, используя свой же словарь фонем!

Итак, на входе синтезатора — символ в коде КОИ-8. По таблице «код-адрес» программа находит адрес строки, в которой (тоже в коде КОИ-8) записано, что должен сказать синтезатор. При этом учитывается режим (слитной речи или посимвольный) и происходит обращение к той или иной таблице фраз. Допустим, режим посимвольный, на входе — код 341 (заглавная русская буква А; синтезатор не различает заглавные и строчные буквы). Обратившись к ячейке 341 первой таблицы, программа находит там адрес, а по этому адресу — два разных указания: одно — просто «А», второе — слово «Алексей». Определив, что режим — посимвольный, синтезатор начинает выборку по буквам второго слова, сопоставляя каждой букве фонему из словаря, причем фонемы сразу переводятся в импульсы, поступающие на выход компьютера. Звучит речь. (Заметим, что сам словарь фонем — весьма сложная структура, состоящая из таблиц, где каждой фонеме сопоставлен не только ее адрес в памяти, но и длительность звучания, и частота звука — они у всех звуков разные.)

А если нужно синтезировать слова не русской речи, а другого языка? Что же, русская фонетика довольно богата, и с ее помощью можно эмулировать, видимо, любой язык. Казахский реализован, делались опыты с итальянским, французским, испанским (про английский и говорить нечего — попробовали первым!) — вполне успешно: люди, не знакомые с данным языком (целостность звукового образа отсутствует!), довольно легко понимали и повторяли произнесенные синтезатором слова и фразы. Конечно, слова иностранных языков приходится записывать подходящими русскими буквами (фонетическая запись текста, принятая во многих зарубежных системах).

Надо сказать, что к синтезированной речи приходится привыкать, чтобы понимание ее стало легким и безошибочным. Большинству людей для этого требуется не более часа, после чего навык остается на всю жизнь. Судя по монографиям, не только наш синтезатор требует тренировки для легкого восприятия синтезированной речи. Считается, что синтезатор хорош, если после такой тренировки правильно воспринимается около 80 % слов. Смеем заверить, что наш дает гораздо лучший процент восприятия, хотя специально мы его не тестировали.

Вот как будто и все. Синтезатор был создан, тема сдана, слепые довольно легко научились работать с компьютером и понимать его речь. Увы, судьба этой разработки оказалась типичной для нашей страны — о ней никто не знает. Приспособление, могущее облегчить и скрасить жизнь тысячам инвалидов по зрению, помочь в обучении слепых детей, да и мало ли в чем еще, существует в единственном экземпляре. Правда, сама программа гораздо более распространена — ее тиражируют (имя файла — ГОВОРУН, вы еще с ним не знакомы?) и покупают. Пользователи БК приходят в восторг, а те из них, кто представляет, что это за труд — создать синтезатор речи,— восторгаются энтузиазмом программистов, сделавших это «за так». Но теперь вы всю историю знаете и можете не умиляться нашему энтузиазму и бескорыстию — синтезатор ГОВОРУН создан «в порядке служебного задания». Восхититесь лучше великолепными бюрократическими навыками чиновников, которые, имея такую разработку в своем полном распоряжении, не торопятся ее внедрить в той области, для которой она предназначена. Что ж, над ними не каплет. И тратятся ежегодно большие средства на разработку компьютерных систем для слепых — на основе более дорогих и сложных машин, часто импортных. Впрочем, и это понятно — всем есть-пить надо... Такой вот немного грустный финал.

А что же аванс, данный в заглавии? Насчет слуха? Неужели можно на БК и это?! Ведь задача анализа слитной речи не решена еще нигде! Даже на мощных и дорогих машинах имеются только частные случаи решений — ограниченный словарь понимаемых машиной слов, ограниченный круг общающихся с ней операторов... Да, это так. И «аванс» пока не более чем трюк, рассчитанный на привлечение внимания читателя к статье. И все-таки... Что же скрывать — мы думаем и над этой проблемой, и есть уже кое-какие идеи. Но нет пока организации или предприятия, которым это было бы нужно, а делать такие вещи «на энтузиазме» — пусть, кто хочет, верит, что это возможно! Ведь надо работать, т. е. заниматься делом, за которое платят зарплату.

Ну а в принципе — если вы не верите, что можно создать на БК программу, понимающую речь, то, значит, рассказанная история создания синтезатора речи вас ничему не научила.

Литература

1. *Косарев Ю. А.* Естественная форма диалога с ЭВМ. Л.: Машиностроение, 1989.

2. *Обжелян Н. К., Трунин-Донской В. Н.* Машины, которые говорят и слущают. Кишинев: Штиинца, 1987.

3. *Михайлов В. Г., Златоустова Л. В.* Измерение параметров речи. М.: Радио и связь, 1987.

4. *Горелов И. Н.* Разговор с компьютером: Психолингвистический аспект проблемы. М.: Наука, 1987.

ЧТО МОЖЕТ ЭВМ?

Нам бы их заботы...

Министерство экономики Японии недавно опубликовало свою стратегию промышленной политики на 90-е гг. Третьим из недостатков сегодняшней организации «народного хозяйства» страны названо... отставание от США и Западной Европы в использовании современных методов обработки информации и коммуникационной техники, в частности банков данных. По мнению министерства, овладение этими методами позволит добиться решающих изменений в экономике.

Не ограничиваясь констатацией этого факта, власти Японии выделяют на преодоление отставания немалые суммы.

Мы также признаем свое отставание в части информатики; но похоже ли оно на японское?

RISC — без всякого риска

Любой военный подтвердит: если число команд невелико, каждая отдана коротко и четко, можно надеяться на быстрое и правильное исполнение.

Новый процессор, разработанный фирмой IBM, организован именно на таких принципах: минимальное количество самых необходимых коротких и быстрых команд. Возможно, на эту мысль конструкторов натолкнул анализ частоты использования команд процессора ЭВМ IBM-360, который показал, что в большинстве программ около 80 % процессорного времени приходится на выполнение всего 20 % команд. Следовательно, сократив время выполнения этих операций, можно резко повысить скорость обработки информации, а избавившись от остальных 80 % команд — предельно упростить арифметико-логическое устройство.

В результате фирмой Intel на основе сокращенного таким образом набора команд был разработан RISC-процессор I-80860. Все целочисленные арифметические и логические операции в этом процессоре выполняются всего за один цикл, что позволяет ему выполнять программы со скоростью до 120 млн. операций в секунду.

Для обеспечения эффективной работы RISC-процессора на этом же кристалле размещена оперативная память для запоминания состояний выполняемой программы и векторный процессор для работы с многомерными величинами.

И все же, несмотря на столь радикальные аппаратные и программные новшества нового процессора, аппаратная реализация концепции сокращенного набора команд, по утверждению сотрудника фирмы Билла Раша,— это старый добрый «фон-Нейман» (хоть и трудноузнаваемый), но никак не новая компьютерная архитектура. И уже поэтому RISCа не стоит опасаться.

Сменные диски на все вкусы

Быстрая запись на внешние носители больших объемов информации еще недавно требовала использования винчестерских дисков. Они и сейчас пользуются большой популярностью, однако невозможность легко заменить диск в компьютере все чаще представляется пользователям неприемлемой. В результате выросло число разнообразных модификаций, позволяющих осуществлять замену.

В системе SQ555 фирмы Syguest меняется только сам диск, точнее, содержащая его кассета размером 133×133 мм и толщиной 6,3 мм. Объем хранимой на диске информации — 40М байт. Однако долговечность диска и дисковода в такой «разборной» конструкции, конечно, не может быть очень высокой.

По этой причине накопитель Personal Data Pac компании Tandon сконструирован иначе: герметизированный сменный блок размером 54×120×178 мм содержит не только диск емкостью 30—40М байт, но и двигатель, и магнитные головки. Пожертвовать здесь пришлось размерами и ценой: мегабайт емкости блока стоит 14,85 доллара, в то время как для предыдущего носителя эта цифра равна 4,53 доллара.

Естественным конкурентом винчестеров являются накопители на оптических дисках. Система WORM позволяет записывать диск только один раз, однако громадной емкости все равно хватает надолго. В накопителе LF-5010 компании Panasonic, например, 133-мм диск в кассете толщиной 6,3 мм позволяет записать 940М байт. Когда диск все-таки заполнится, нетрудно купить новый: цена мегабайта его емкости всего 17 центов.

Магнитооптические диски можно и заполнять информацией, и стирать. В накопителе LSE1-100 AT компании Storage Dimensions на такой диск (в кассете 133×133×6,3 мм) записывается 840М байт при цене носителя 45 центов за мегабайт. К сожалению, быстродействие таких накопителей значительно меньше, чем у винчестеров: для стирания с магнитооптического диска требуется специальный предварительный проход головки, а большая вероятность ошибок требует контрольного считывания после записи.

Про накопитель Bernoulli Box II44 в «ИНФО» уже сообщалось. На том же принципе построен накопитель для дисков из специального материала (его назвали цифровой бумагой), состоящего из полиэфирной основы, металлического отражающего слоя, слоя полупрозрачного полимера и защитного покрытия — все вместе тоньше волоса. Специально подобранные соотношения толщин слоев делают цифровую бумагу очень чувствительной к лазерному излучению и позволяют маломощному лазеру быстро выжигать на ней микроточки. На диск диаметром 133 мм из этой «бумаги» можно записать 1Г байт информации.

МОЛОДЕЖНАЯ ИНИЦИАТИВА

J-LISP — ЛИСП для новичков

В ВЦ школы № 542 при МИФИ была разработана интегрированная система ЛИСПа для ПЭВМ «Ямаха» *MSX-2-J — LISP*. *J-LISP* является упрощением языка ЛИСП, сохраняющим тем не менее все его основные черты.

Интегрированная система включает в себя экранный редактор, интерпретатор *J-LISP*, справочник по языку и редактору (*help*), и средства общения с диском (просмотр каталога, чтение и запись файлов), объединенные в одном файле размером 33К байт. Из них 7К байт занимают редактор и интегрирующая среда, 11К байт — справочник и 12К байт — интерпретатор. 3К байта занимает заставка, появляющаяся на экране при запуске. Существует интерпретатор в отдельном файле (без редактора, справочника и интегрирующей среды), требующий лишь 64К байта для своей работы. Он может быть запущен на ПЭВМ *MSX-1*.

Внутри исходного файла система представляет собой три отдельных модуля: заставка, которая после запуска освобождает занимаемую память, редактор со справочником и интегрирующей средой, размещаемые на трех основных страницах ОЗУ, и интерпретатор — на трех страницах расширенного ОЗУ. Оставшиеся две страницы нужны для размещения операционной системы *MSX-DOS* и рабочих подпрограмм сетевого пакета для *MSX-2*, разработанного в нашем ВЦ М. Юсовым,— в случае работы на бездисководном компьютере. Описанное выше распределение памяти осуществляется системой автоматически при запуске.

Экранный редактор по командному интерфейсу является подмножеством хорошо известного TOR. Он позволяет выполнять простейшие операции, такие, как удаление символа, строки, вставка строки, рассечение и склейка строк, копирование и удаление блока. Редактор может работать в режимах вставки и замены. Есть возможность автоматической табуляции при переводе строки. Справочник представляет собой систему вложенных меню, позволяющих получить доступ к необходимой информации.

Интегрирующая среда позволяет вызывать редактор, справочник, запускать интерпретатор с загрузкой в него программы из редактора, запускать интерпретатор без загрузки программы, записывать файл из редактора на диск с резервной копией и без нее, загружать файл в

редактор со стиранием предыдущего или без стирания (подгрузка). Кроме того, можно просмотреть каталог диска и выбрать файл для дальнейшей работы.

Размеры статьи не позволяют опубликовать в полном объеме учебник *J-LISP* (он прилагается к системе в качестве сопроводительной документации, с примерами и задачами), поэтому здесь будет дано лишь краткое описание языка *J-LISP*.

ЛИСП является языком, сильно отличающимся от большинства других широко распространенных языков, таких, как Бейсик, Фортран, Паскаль или Си. Первое, бросающееся в глаза отличие — обилие скобок, часто шокирующее людей, впервые сталкивающихся с ЛИСПом. Скобки, пробелы и точки — единственные синтаксические разделители в ЛИСПе. Но это страшно только на первый взгляд. Уже небольшой практики работы с ЛИСПом достаточно, чтобы привыкнуть к скобкам. Зато способ записи программ на ЛИСПе дает многие преимущества, которые, видимо, и способствовали такой высокой популярности этого языка.

Все важнейшие преимущества ЛИСПа над другими языками можно кратко назвать одним словом: универсальность. В ЛИСПе являются универсальными форма представления данных и программы (она одна и та же, что позволяет легко модифицировать программу прямо во время ее исполнения), универсальны переменные, поскольку каждая переменная может принимать значения любого типа (например, сначала число, потом строка, потом сложная структура), универсальны функции ЛИСПа, как встроенные, так и определенные пользователем, так как многие из них обрабатывают лишь структуру данных, но не ее содержание. Можно возразить, что те же самые возможности представляет любой Ассемблер. Автор статьи не спорит с этим утверждением, но хотел бы заметить, что ЛИСП является языком гораздо более высокого уровня, чем Ассемблер, и поэтому на нем писать гораздо легче.

Рассмотрим структуры данных, поддерживаемые *J-LISP*. В нем есть пустой список: () или *NIL* — это два его эквивалентных обозначения. Пустой список — это лисповское значение, которое не может быть отнесено ни к одному из нижеперечисленных типов данных и поэтому стоит особняком. Тем не менее пустой список играет исключительно важную роль в ЛИСПе, будучи основой для построения всех непустых списков. Также это значение зарезервировано для обозначения логической лжи. Следующий базовый тип — целое число со знаком. В *J-LISP* оно может быть записано в десятичной или в шестнадцатиричной системе и принимать значения от —32768 до +32767: 1 —10 $FFFF. Еще один базовый тип ЛИСПа — идентификатор. Идентификатор в *J-LISP* — это последовательность любых символов, кроме символов, имеющих специальные значения: () пробел . " '. Идентификаторы могут использоваться сами по себе, например как значения перечисляемого типа в Паскале, либо для обозначения переменных. Идентификатор может быть присвоен как значение другой переменной (идентификатору). И последний базовый тип *J-LISP* — это строка: "", *"ABC"#$D#$A, "GLOPPING PROGRAMMING"* (примеры строк). Целые числа после знака # позволяют задавать символы по их кодам *ASCII*.

Любой набор лисповских значений (возможно — различных типов) может быть объединен в список: (1 2 3), (A1B2), (1), ((1 2) (3 4) "QWERTY" NIL) — последний список состоит из двух списков, строки и пустого списка. *NIL* и (*NIL*), очевидно, являются разными лисповскими значениями: первое — это пустой список, а второе — список, состоящий из пустого списка. К спискам можно применять операции получения первого элемента и получения хвоста (всего списка, кроме его первого элемента). Выше мы пообещали, что данные и программа записываются в одинаковом виде. Программа на ЛИСПе состоит из обращений к функциям. Обращения к функциям записываются в виде списков, где первым элементом является имя функции (идентификатор), а дальше идут ее аргументы. Например, чтобы получить первый элемент списка (1 2 3 4), хранящегося в переменной X, надо написать: (CAR X). Значением выражения (CAR X) в данном случае будет 1. Чтобы получить хвост списка, надо использовать функцию CDR. В нашем случае (CDR X) будет равно (2 3 4). Если X равно (1), то (CAR X) равно 1, а (CDR X) равно *NIL*.

На самом деле списки ЛИСПа являются частным случаем более общей структуры данных — точечной пары. В точечных обозначениях список (1 2 3) будет записан как (1.(2.(3.*NIL*))). Функции CAR и CDR, будучи применены к парам, дают их первый и последний элемент соответственно (отсюда понятно, почему они работают со списками, как описано выше).

Кажется удивительным, что списки и пары способны дать возможность удобно манипулировать сложными

структурами данных. Тем не менее это так. Доказательством этого служит широкая популярность ЛИСПа. Конечно же, ЛИСП неудобен для численных расчетов, но для обработки данных он — один из лучших языков. Здесь нет места для более подробного описания языка, но для знакомых с языком ЛИСП мы приведем описание отличительных черт.

Определяющие выражения могут задаваться как *LAMBDA*- или *NLAMBDA*-выражения, в последнем случае они не вычисляют значения своих аргументов, а получают их в «первозданном виде». Тела определяющих выражений могут содержать несколько вызовов функций, подобно *PROG*-аппарату. Функцию *QUOTE* допускается заменять апострофом. Тела условных выражений функции *COND* также могут содержать несколько вызовов. Так же реализованы функции *SELECTQ*, *PROG*-аппарат, *GO*, *RETURN*.

Понятие списков свойств в *J-LISP* отсутствует. Функции задаются путем присваивания их определяющих выражений переменным:

```
(SETQQ APPEND
  (LAMBDA (X Y) (COND
  ((NULL X) Y)
      (T (CONS (CAR X)
    (APPEND (CDR X) Y)
    ))
)))
```

J-LISP содержит набор арифметических, строковых и логических функций. Также есть средства перенаправления ввода-вывода, позволяющие работать с файлами, средства работы с экраном, вплоть до прямого обращения к видеопамяти, и средства опроса клавиатуры.

На *J-LISP* были написаны игровая программа «Удавчик», система «Животные» и программа аналитического дифференцирования простейших выражений. Реализована и оболочка простейшей экспертной системы для идентификации объектов, включающая в себя подсистему объяснения своих действий. Текст оболочки занял 3К байта. В настоящее время существует версия *J-LISP* для *IBM PC*, подобная версии для *MSX-2*.

В. Завалишин
студент III курса МИФИ

Сугубо машинное решение

Существует класс задач, решение которых связано с нахождением или подсчетом точек с целочисленными координатами. Зачастую эти задачи довольно сложны, причем для решения каждой из них требуется особый подход.

Рассмотрим математическое решение следующей простейшей задачи. Дан треугольник ABC. Требуется подсчитать число точек с целочисленными координатами, находящихся внутри и на границе области, ограниченной этим треугольником. Координаты точек A, B, C — (X1, Y1), (X2, Y2), (X3, Y3).

Напишем уравнения прямых AB, AC, BC.

$$(Y-Y1)/(Y2-Y1)=(X-X1)/(X2-X1),$$
$$(Y-Y2)/(Y3-Y2)=(X-X2)/(X3-X2),$$
$$(Y-Y1)/(Y3-Y1)=(X-X1)/(X3-X1).$$

Запишем их в форме $ax+by+c=0$. Далее воспользуемся следующим утверждением: если прямая, определенная уравнением $ax+by+c=0$, пересекает отрезок MN, то произведение левой части уравнения, вычисленной в точке N, на левую часть уравнения, вычисленную в точке M, будет отрицательно; и наоборот, если отрезок MN не пересекает прямую, описанную уравнением $ax+by+c=0$, то соответствующее произведение будет положительно.

Таким образом перебираем все точки по циклу (от минимальной до максимальной абсциссы и от минимальной до максимальной ординаты) и подсчитываем количество точек, удовлетворяющих этому утверждению.

Итак, уже в случае простейшей треугольной области задача довольно сложна. Однако, если более полно использовать возможности ПЭВМ, ее решение резко упростится.

Метод заключается в следующем. На дисплее строится заданная фигура, потом закрашивается фон вне фигуры в цвет ее границ (если требуется подсчитать количество точек с целочисленными координатами строго внутри фигуры) или закрашивается фон внутри фигуры в цвет ее границ (если требуется подсчитать количество точек с целочисленными координатами в фигуре, включая границы). Затем определяются минимальные абсцисса и ордината и организуется двойной цикл по X и Y.

Перебирая прямоугольную область, в которой содержится данная фигура, подсчитываем количество точек, удовлетворяющих условию задачи. Признаком попадания перебираемой точки в фигуру является совпадение цвета этой точки с цветом фигуры.

Используя данный метод, можно, в частности, определить площадь любого многоугольника с целочисленными координатами вершин по формуле Пика $J+R/2-1$, где J — количество целочисленных точек внутри фигуры, а R — на ее границе.

В заключение — программа, вычисляющая количество точек с целочисленными координатами внутри и на границе треугольника. Для наглядности истинные координаты увеличены в 10 раз.

```
5 OPEN"GRP:"AS#1
10 DIM X(3),Y(3)
15 'ВВОД КООРДИНАТ
      ВЕРШИН ТРЕУГОЛЬНИКА
20 FOR P=1 TO 3
30 PRINT "ВВЕДИТЕ
         КООРДИНАТЫ";P;"-й
           ВЕРШИНЫ (X,Y)"
          :INPUT X(P),Y(P)
40 NEXT P
45 'ОПРЕДЕЛЕНИЕ
            МАКСИМАЛЬНОЙ И
     МИНИМАЛЬНОЙ АБСЦИССЫ
                И ОРДИНАТЫ
50 A=X(1):B=X(2):C=Y(1)
                    :D=Y(2)
60 IF A<B THEN SWAP A,B
70 IF X(3)>A THEN A=X(3)
80 IF X(3)<B THEN B=X(3)
90 IF C<D THEN SWAP C,D
100 IF Y(3)>C THEN C=Y(3)
110 IF Y(3)<D THEN D=Y(3)
115 'ВХОД В ГРАФИЧЕСКИЙ
                      РЕЖИМ
120 SCREEN 5
125 'РИСОВАНИЕ ФИГУРЫ
130 LINE (X(1)*10,Y(1)*10)
    -(X(2)*10, Y(2)*10),15
140 LINE-(X(3)*10,
              Y(3)*10),15
150 LINE-(X(1)*10,
              Y(1)*10),15
155 'ВЫЧИСЛЕНИЕ КООРДИНАТ
     ЦЕНТРА ТЯЖЕСТИ ФИГУРЫ
     ДЛЯ ГАРАНТИРОВАННОГО
  ПОПАДАНИЯ ВНУТРЬ ФИГУРЫ
160 FOR T=1 TO 3:G=G+X(T)
    :L=L+Y(T): NEXT:G=G/3
                    :L=L/3
165 'ЗАКРАСКА ОБЛАСТИ
          ВНУТРИ ФИГУРЫ
170 PAINT(G*10,L*10),15
180 S=0
190 FOR X=B*10 TO A*10
              STEP 10
200 FOR Y=D*10 TO C*10
              STEP 10
210 IF POINT(X,Y)=15 THEN
      S=S+1: PSET(X,Y),1
220 NEXT:NEXT
230 PRESET(200,40)
                :PRINT#1,S
240 A$=INPUT$(1)
```

К. РЕШЕТИН, X класс,
г. Павлодар

Крутые игры

Вам хочется романтики и острых ощущений? Вы хотите полететь на собственной ракете в неизведанную галактику? Или, применяя приемы кунг-фу и каратэ, разделываться с любым количеством врагов? Участвовать в ночных автогонках, стать кооператором или главой «Джапан Виктори Компани», просто поиграть в шахматы, шашки, морской бой, домино, кубик Рубика? Все это даст вам программируемый калькулятор и суперпрограммы Клуба любителей игровых программ (КЛИП). В нашем клубе собрана библиотека игр на любой вкус; с нашей помощью вы сможете найти друзей-единомышленников в каждом уголке Советского Союза, узнать новые эффективные приемы программирования, модернизировать свой ПМК, принять участие в различных конкурсах, а также стать автором «ИНФО».

КЛИП организован в 1988 г., объединяет более 400 членов. Руководитель — Михаил Храмов (443081, Куйбышев, ул. Стара Загора, 53, кв. 55; тел. (8462) 51-44-85). Большинство его членов — любящие электронные игры владельцы ПМК МК61 и МК52.

Для первого знакомства — две программы Сергея Тарасова из остросюжетного цикла «Супермен».

Внимание! Безопасность игроков не гарантируется! Ваша жизнь — в ваших руках, читатель!

В одном сказочном царстве (или государстве), в живописной Долине Драконов живет молодой и сильный человек по имени Брюс Ли — Маленький Дракон (эта роль — для вас, читатель!). По 10 часов в день он занимается всеми возможными и невозможными видами единоборств, рукой ломает каменные глыбы, пьет расплавленное олово, расшибает головой сразу четыре кирпича (это перед завтраком) и т. д. Брюс также тренируется в вождении автомобиля, ракеты, велосипеда и роликовой доски. По ночам же молодой супермен играет с компьютером в шахматы, го, рэндзю и собирает кубик Рубика 120×120×120.

А невдалеке от Долины Драконов находится Холм Дьявола. На самой его вершине уперся в небо своими мрачными башнями замок злодея, чернокнижника и колдуна Во Ланда. Вокруг — дремучий лес, глубокий ров (в него сливает свои отходы ближайший химзавод), колючая проволока под высоким напряжением. Сам замок — многоэтажный лабиринт, в котором кишмя кишат вампиры, людоеды и прочая нечисть. На последнем этаже расположился сам колдун со своей пленницей (понятное дело, прекрасной) Марикой.

Брюса Ли и Марику связывает нечто большее, чем дружба. Наш (ваш) герой пытается спасти возлюбленную и проникает в ужасный замок, но... Впрочем, что будет потом — будет потом, а пока Брюс, вооруженный только собственным телом, стремится к логову врага. Вам же, читатель, остается ввести в ПМК программу «Смертельная игра» (по моти-

вам одноименного фильма с реальным Брюсом Ли в главной роли) и испытать свои силы в борьбе с обитателями замка.

Введите текст программы в ПМК.

	0	1	2	3	4	5	6	7	8	9
0	В↑	В↑	В↑	В↑	В↑	В↑	В↑	В↑	В↑	В↑
1	ИПС	Fcos	В/О	ИП3	КППС	ИП5	КИНВ	КППС	ПА	Fx<0
2	КИП2	ИПА	/-/	Fx<0	30	КП7	2	−	Fx<0	29
3	КПП7	Fx≠0	38	5	−	Kx=0B	ИПА	Kx=0B	ИП0	В↑
4	КСЧ	П0	6	X	K[x]	1	−	Fx<0	56	ИП5
5	K{x}	5	+	ИП9	:	П5	ВП	/-/	7	ИП5
6	K{x}	8	+	+	П5	КИНВ	КППС	КЗН	ПА	Fx≥0
7	93	Fx=0	74	КИП2	КИП2	ИП2	FJ	ИПА	2	−
8	F10↑	7	X	8	+	ИП8	КА	К0	ИП5	XY
9	КА	К0	П5	FL3	13	ИП1	С/П	ИП5	K{x}	ИП9
A	X	П5	K[x]	В/О						

Теперь введите

```
10 П9 100 ПС 81 ПВ 29 ПВ 97
      П7 <произвольное число> П0
```

Сформируйте текст сообщения о победе «LEE»:

```
П1 8411 КИНВ K{x} ВП 3 П1
```

Игра начинается! БП 13. Длина пути (30) в П3, запас энергии (25) в П2; нажимайте СХ П5 и С/П.

Вы бежите по лабиринту замка. Вначале на индикаторе мелькает длина пути, затем обстановка перед вами (видеоблок перемещений). Здесь переключателем угловых мер нужно выбрать способ движения вперед: Р — в прыжке, ГРД — присев (эти способы требуют затраты энергии), Г — в обычной стойке. Восьмерка на индикаторе — это вы, Е обозначает стену, тире — летящий в вас нож, Г — противника, наносящего удар рукой, [— наносящего удар рукой с одновременной подсечкой, L — наносящего удар ногой вверх.

Один участок пути — один разряд индикатора. При прыжке вы перемещаетесь через один участок пути. Перепрыгнуть можно через пустые места и стены, но не через нож или противника (наказание — ЕГГОГ, смерть). Присев, можно спастись от ножа и только.

На размышление отводится 1—3 с. Затем появляется видеоблок действий. Переключатель угловых мер позволит вам провести удар рукой (Р), ногой (ГРД) или отказаться от проведения приемов (Г). Удар рукой действует, если враг стоит вплотную. Им нельзя поразить противника, выполняющего удар с подсечкой, или проломить стену, зато можно отклонить нож. Осторожно — вместо ножа и противника «[» может появиться стена!

Удар ногой требует вдвое больше энергии, зато действует на расстоянии одного участка без всяких ограничений — им можно даже проломить стену.

При исчерпании энергии или столкновении с чем-либо вас ждет смерть. Знак победы — имя LEE на экране.

Но вот Брюс Ли разделался со всеми слугами злого колдуна и пробрался в самое его логово. Увы! Марики там не было! Хитрый Во Ланд успел убежать, и Брюс лишь увидал в окно удаляющиеся огни его автомобиля. «Он не уйдет!» — подумал молодой воин и бросился в погоню на своей гоночной машине.

А слуги колдуна не дремлют. Они пытаются скинуть преследователя в пропасть или расплющить ее своими авто. Скорее вводите в ПМК программу «Ночная трасса — Д»!

	0	1	2	3	4	5	6	7	8	9
0	В↑	В↑	В↑	В↑	В↑	В↑	В↑	В↑	В↑	В↑
1		ИПД	Fcos	КЗН	В/О	ПА	КИНВ	КППЕ	ВП	Fx≥0
2	20	ПЕ	ИПВ	F10↑	8	+	КИНВ	КППЕ	ИПВ	+
3	ПВ	Fx≠0	2С(2ВП)	7	+	Fx≥0	32	1	КП7	/-/
4	Fx=0	44	4	КПП7	ИПС	+	ПС	FJ	FBx	5
5	−	Fx≠0	ПС	ИПС	КППЕ	FL2	67	ИП4	С/П	П2
6	/-/	ПВ	0	ПС	ПЕ	БП	15	ИП6	В↑	КСЧ
7	П6	8	+	КИПЕ	КА	FBx	K[x]	:	ПА	XY
8	ИП0	КА	2	:	П9	4	:	ИПА	XY	КА
9	К0	ИП9	КУ	ВП	15	ИПВ	F10↑	X	8	+
A	ИПА	КА	K{x}	КЗН	В/О					

После ввода программы наберите

```
1.1111111 П3 4 X П1 2 X
                  П0 100 ПД 95 П7
```

и сформируйте текст сообщения о финише «ELE—ELE»:

```
88888888 В↑ 86362636 КУ
                  K{x} ВП 7 П4
```

Начинайте игру: БП 59 (так же начинается и новая гонка после аварии). Длина трассы (20) ППδ, где δ — номер дорожки (от 1 до 7), на которую вы ставите свою машину. С/П — и в путь!

В начале трасса пустая, и можно отрегулировать ее сложность (количество препятствий) и скорость автомобиля (время на раздумья при смене очередной картинки) переключателем ГРД—Р—Г: ГРД и Р — высокий уровень сложности и максимальная скорость, Г — низкий уровень сложности и минимальная скорость.

Восьмерка слева — ограждение трассы, E — автомобили, L — «жизни» (наехав на L, вы получаете некоторый иммунитет против аварий; не жадничайте чересчур — если число накопленных «жизней» превысит пять, все они аннулируются!), « » — пустые дорожки.

Через несколько минут индицируется положение вашего автомобиля. Управление — переключателем угловых мер: Р — влево, ГРД — прямо, Г — вправо на одну дорожку. Гонка прекращается при столкновении с другим автомобилем (если в запасе у вас нет жизней) или при выезде за пределы трассы. Еще несколько секунд ожидания — и высвечивается количество запасенных «жизней», а дальше все повторяется сначала.

При успешном прохождении трассы можно вводить новую длину дорожки и ее номер.

Тем временем Брюс Ли догнал Во Ланда, но злодей схватил Марику за руку и... Впрочем, пока достаточно. Продолжение — в следующем выпуске.

Напишите нам, понравились ли вам программы. Оцените их по 10-балльной системе по следующим параметрам: сервис, игра, общая оценка. Авторы рубрики ждут от вас новых идей, сюжетов, собственных разработок. Опытные программисты помогут вам в разработке ваших программ. Не забывайте кратко рассказать о себе. Писать можно в редакцию «ИНФО», М. Храмову и Вадиму Московцеву (143400, Московская обл., Красногорск-4, а/я 54).

Счастливых приключений, супермены!

В. МОСКОВЦЕВ

Учебно-игровая программа «Глаголы»

Чем может помочь компьютер на уроке английского языка? А вот чем.

Язык — это двуликий Янус. Сколько бы вы ни знали отдельных слов, связать их в предложение не сможете, не зная грамматики. И наоборот, как бы вы ни овладели правилами морфологии и синтаксиса, не имея словарного запаса, вы будете также беспомощны.

Настоящую пользу от занятий английским языком получит только тот, кто выучит глаголы, наиболее часто употребляемые в английской речи, книгах, газетах. И выучит их как полагается: если глагол неправильный, то сразу в трех основных формах. Как это ни огорчительно, но самые распространенные английские глаголы по способу образования основных грамматических форм являются неправильными, что требует их запоминания сразу в трех формах: инфинитив (*INFINITIVE*), простое прошедшее время (*PAST INDEFINITE*) и причастие прошедшего времени (*PAST PARTICIPLE*). Всего неправильных глаголов около 200.

Как всякое механическое запоминание, заучивание форм неправильных глаголов — занятие достаточно трудное, скучное и утомительное. Облегчить этот процесс помогает игра. Такую полезную и простую игру для ПЭВМ «Агат» мы вам и предлагаем. Итак, компьютерная игра под названием «*VERBS*», что означает — глаголы.

Замысел игры заключается в следующем: на экране дисплея появляется глагол в одной из трех форм, необходимых для запоминания. Он движется в направлении трех приготовленных неподвижных «корзин» — ловушек с названиями форм глагола.

Необходимо, нажимая клавиши управления курсором вверх и вниз, направить глагол в соответствующую ему «корзину». Игра автоматически увеличивает скорость движения глагола, тем самым уменьшая время на обдумывание. Общее число предложенных глаголов и число правильных действий фиксируется в таблице результатов игры.

Игровая программа «*VERBS*» состоит из двух частей (игровой и набора глаголов). Игровая часть — программные строки от 10 до 250, набор глаголов — от 260 до 290.

И далее:

строка 30 — вывод на экране дисплея названия игры и исходной таблицы результатов;

строка 50 — рисование ловушек;

строка 60 — вывод названий ловушек;

строка 80 — считывание глаголов и чисел соответственно в массивы W$(N) и C(N).

```
10 REM ...ИГРОВАЯ ОБУЧАЮЩАЯ ПРОГРАММА "VERBS"...
20 TEXT=15:NORMAL:HOME
30 HTAB7:VTAB2:RIBBON=2:PRINT">>>";:RIBBON=1:
   PRINT " GAME "CHR(34)"VERBS"CHR(34);:RIBBON=2:
                                        PRINT"<<<"
40 VTAB4:RIBBON=6:PRINT"TOTAL VERBS:0":
                                    PRINT "RIGHT VERBS:0"
50 POKE 50,4:FOR I=9 TO 25 STEP 8:HTAB23:
   VTAB I-1: PRINT"*********":HTAB23:VTAB I+1:
   PRINT"*********":HTAB31: VTAB I:PRINT"*":NEXT I
60 POKE 50,45:HTAB 23:VTAB 7:PRINT "INFINITIV":
   HTAB 23:VTAB 15: PRINT "PAST IND.":HTAB 23:
   VTAB 23:PRINT "PAST PAR.":POKE 50,4: VTAB 17:
                                          PRINT ">"
70 N=3:SK=80:REM N-ЧИСЛО ГЛАГОЛОВ В СПИСКЕ,
                                 SK- СКОРОСТЬ ПАДЕНИЯ СЛОВ
80 DIM W$(N),C(N):FOR I=1 TO N:READ W$(I),C(I):NEXT I:
   REM ОБЬЯВЛЕНИЕ МАССИВОВ И ИХ ЗАПОЛНЕНИЕ ИЗ DATA
90 FOR ZZ=1 TO 10:REM ЧЕРЕЗ СКОЛЬКО ГЛАГОЛОВ
                                    ВОЗРАСТАЕТ СКОРОСТЬ
100 E=INT(RND(1)*N+1):N$=W$(E):C=C(E):L=23-LEN(N$):
                                    Y=17:NORMAL: RIBBON=3
110 FOR F=1 TO LEN(N$):HTAB 2:VTAB 17:
    PRINT RIGHT$(N$,F);:FOR Z=1 TO 110:NEXT Z,F:
                                REM ВЫПОЛЗАНИЕ СЛОВА ИЗ ">"
120 FOR F=2 TO L:HTAB F-1+(F=2):VTAB Y:
                                  PRINT SPC(LEN(N$));" "
130 K=PEEK($C000):REM ОПРОС КЛАВИАТУРЫ
140 IF K=153 AND Y>9 THEN Y=Y-1:
                                  REM СМЕЩЕНИЕ СЛОВА ВВЕРХ
150 IF K=154 AND Y<25 THEN Y=Y+1:
                                  REM СМЕЩЕНИЕ СЛОВА ВНИЗ
160 HTAB F:VTAB Y:PRINT N$:FOR Z=1 TO SK:NEXT:
                REM ВЫВОД СЛОВА И ЗАДЕРЖКА ПО СКОРОСТИ
170 NEXT:REM СЛОВО ПОДОШЛО К КОРЗИНЕ
180 IF Y<>9 AND Y<>17 AND Y<>25 THEN HTAB F-1:
    VTAB Y:PRINT SPC (LEN(N$)):GOTO 100:
                  REM ПОПАЛО ЛИ СЛОВО В ОДНУ ИЗ КОРЗИН
190 FOR I=F-1 TO F-1+LEN(N$):HTAB I-1:VTAB Y:
    PRINT " ";N$:FOR Z=1 TO SK:NEXT Z,I:PR=0:
                              REM СЛОВО ЗАПОЛЗАЕТ В КОРЗИНУ
200 IF MID$(RIGHT$("00"+STR$(C),3),(Y-1)/8,1)="1"
    THEN O=O+1: HTAB I-1: VTAB Y:POKE 50,10:
    PRINT N$:GOTO 220:
                    REM В ТУ ЛИ КОРЗИНУ ЗАПОЛЗЛО СЛОВО
210 HTAB I-1:VTAB Y:POKE 50,9:PRINT N$
220 FOR I=1 TO 999:NEXT:HTAB  23:VTAB Y:NORMAL:
                                      PRINT SPC(8):S=S+1
230 IF S<50 THEN POKE $C010,0:HTAB 14:VTAB 4:
    RIBBON=6: PRINT S:HTAB 14:VTAB 5:PRINT O:NEXT:
    SK=SK-20:PRINT CHR$(7);:GOTO 90:
                                  REM УВЕЛИЧЕНИЕ СКОРОСТИ
240 NORMAL:RIBBON=3:HTAB 2:VTAB 17:
    PRINT " ONCE MORE ? Y";CHR$(8);:POKE $C010,0:
    GET A$:IF A$=CHR$(13) OR A$="Y" THEN RUN:
                            REM КОНЕЦ. БУДЕТЕ ИГРАТЬ ЕЩЕ?
250 END
260 REM ...ЗДЕСЬ СПИСОК СЛОВ ...
270 DATA "RUN",101,"SET",111,"DRIVEN",1
```

Единицей обозначается ловушка, соответствующая правильному положению для данного глагола;

строка 100 — выборка по случайному закону глаголов из массива;

строка 120 — организация цикла движения глагола;

строка 130—150 — опрос клавиатуры и анализ нажатия клавиши для изменения положения глагола, движущегося по вертикали;

строка 180 — анализ: находится глагол перед входом в одну из «корзин» или нет. Если нет, то глагол исчезает и игрок проигрывает этот раунд;

строка 200 — анализ правильности нахождения глагола в ловушке;

строка 240 — вывод на экран дисплея вопроса «ONCE MORE» («Еще раз»). Нажатие на клавишу «Y» — повторение игры.

А. ЗУЕВ, руководитель кружка
А. НЕДОБОЙ, ученик XI класса

Главный вычислительный центр Министерства народного образования УССР распространяет программное обеспечение

Для ПЭВМ «Ямаха»:

▲ пакет педагогических программ по основам информатики и вычислительной техники (удостоен золотой медали ВДНХ СССР);

▲ пакет программ по биологии «Растительная клетка»;

▲ комплекс из 15 психодиагностических тестов;

▲ набор программ для работы с графикой на языках программирования Си, Паскаль, ассемблер;

▲ операционную систему Nike DOS, надстройку Garsy, графический редактор GARSY, графическую библиотеку GEOMED;

▲ комплекс программ по изучению грамматики немецкого языка;

▲ набор программируемых средств поддержки процесса обучения элементам высшей математики.

Для ПЭВМ класса IBM PC:

комплекс программ по изучению грамматики и лексики английского языка для V—VI классов средних школ.

Для КУВТ «Корвет»:

пакет педагогических программ по основам информатики и вычислительной техники (при его создании учитывались недостатки разработок других организаций);

программу печати графического изображения экрана терминала на принтер.

Гарантийные письма-заявки направляйте по адресу: *252053, Киев-53, ул. Артема, 52-Д, ГВЦ Минобразования УССР.*

Справки по телефону (044) 213-97-77.

Значительно расширить возможности его использования КУВТ-86 вам поможет

ARKUSS

Высокое техническое исполнение, эстетичность оформления наших ППС признаны и зарубежными специалистами.

Предлагаемые программные средства, дополнительные устройства позволят повысить качество преподавания информатики и других предметов до совершенно иного уровня.

Обучающие программы и инструментальные средства поставляются на гибких магнитных дисках с техническим описанием и методикой применения.

Новинки ECO, ELVA и другие ждут ваших заказов!

При приобретении полного комплекта прилагаются бесплатные рекомендации по преподаванию курса информатики.

Обеспечиваются сопровождение и консультации по программным средствам.

Производится разработка пакетов программ для УКНЦ-01 (БК-0011), «Ямахи».

Заключаются договоры на тиражирование и распространение высококачественных программных средств.

Для получения подробной информации писать по адресу: 226063, Рига, ул. Кенгарага, 8, «ARKUSS». Телефон: 251691.

В. КАЙМИН

О логике, «лжецах» и доказательном программировании

Данная заметка является ответом на статью В. Белошапки «Три учебника информатики», являвшегося ученым секретарем конкурсной комиссии по отбору учебных пособий по «Основам информатики и вычислительной техники». Хочу напомнить, что в этом конкурсе лучшими были признаны три рукописи: В. А. Каймина и др., В. Н. Касаткина и О. Е. Вершинина. Первая из них получила второе место на конкурсе и издана в качестве пробного учебника в 1989 г., а рукописи, завоевавшие третье место,— в виде учебных пособий в 1990 г.

Параллельно с конкурсом подготовкой пробных учебников по заказу Минпроса СССР и Минпроса РСФСР занимались еще два коллектива: под руководством А. Г. Кушниренко и В. Б. Житомирского. Именно эти учебники, сопоставляя с нашим, анализирует В. Белошапка и предлагает свои выводы в статье, опубликованной в предыдущем номере.

Не берусь обсуждать или осуждать его выводы о других учебниках, но выводы о нашем стоит разобрать обстоятельнее. В. Белошапка утверждает, что этот учебник не отвечает требованиям, предъявляемым к учебникам, в частности не отвечают критерию «научности». Наш контрвопрос состоит в том, каким конкретно требованиям он не отвечает и что такое «научность». К сожалению, в статье В. Белошапки эти понятия не раскрыты, а отсюда, несмотря на нашу общую приверженность логике, данные выводы повисают в воздухе.

Попробуем разобраться. Требования к учебникам бывают двух видов: общие и конкретные. Общие требования связаны с технологией организации учебного процесса и роли в ней учебников. Эти требования распространяются на все учебники, используемые в школах, вузах и т. д.

Конкретные требования связаны с предметом, которому посвящается учебник. В нашем случае — это информатика и основы вычислительной техники. Они были сформулированы в условиях конкурса учебников по информатике (1987 г.) специалистами АН СССР, АПН СССР и Минвуза СССР.

Основными конкретными требованиями к учебникам по информатике были: преемственность к первому пробному учебному пособию, созданному под руководством А. П. Ершова [1], и требование систематического использования ЭВМ на занятиях. При этом основные задачи курса информатики в средних школах в условиях конкурса были определены так: формирование у учащихся компьютерной грамотности и развитие у них информационной культуры.

Именно с этих позиций можно и должно оценивать учебники на соответствие их требованиям постановки курса информатики в школе. Что же мы видим в учебниках? В пробном учебном пособии А. П. Ершова — следующие темы:

1. Основы алгоритмизации.
2. Решение задач на ЭВМ.
3. Элементы программирования (языки Бейсик или Рапира).
4. Основы вычислительной техники.
5. Применения ЭВМ.

При этом за основу обучения было принято изучение алгоритмического языка как основного средства описания самых разнообразных алгоритмов.

Все перечисленные темы мы можем найти в учебниках Б. Б. Житомирского, В. А. Каймина, В. Н. Касаткина и О. Е. Вершинина. В учебнике А. Г. Кушниренко алгоритмический язык принят в качестве средства программирования, и поэтому рассмотрение других языков программирования — опущено. Все перечисленные пять учебников ориентированы на постоянную работу учащихся с ЭВМ на каждом уроке и вполне отвечают и этому требованию.

Вопрос об обучении компьютерной грамотности оказался дискуссионным. Многие из авторов и методистов придерживаются парадигмы «Программирование — вторая грамотность» и считают, что компьютерная

грамотность — это умение составлять программы на одном из учебных языков типа Лого, Рапира или Бейсик. Другая точка зрения, представленная в новейших учебниках у нас и за рубежом, рассматривает компьютерную грамотность как умение пользоваться ЭВМ: редактировать тексты, выполнять расчеты и чертежи, искать информацию в базах данных и вести диалог с ЭВМ. Кто прав в данном вопросе — рассудит время.

По вопросу об информационной культуре также сложилось две точки зрения. Первая, изложенная в конкурсной программе, предполагала под этим формирование у учащихся умений ставить задачи, подбирать математические модели и методы для из решения, составлять алгоритмы для решения задач на ЭВМ, составлять простейшие программы, проводить их отладку на ЭВМ и знать основные возможности вычислительной техники. Данный подход в полной мере был изложен в учебниках В. Б. Житомирского, В. А. Каймина и В. Н. Касаткина.

Альтернативное понимание информационной культуры как умений работать с информацией, используя ЭВМ, состоит в следующем. Под этим понимаются умения получать, накапливать, перерабатывать и передавать информацию с помощью сетей ЭВМ, баз данных, баз знаний и различных автоматизированных систем. Одна из первых попыток реализации данного подхода была предпринята нами при переработке своего учебника [2] и подготовке его к изданию.

Общей частью этих двух подходов является приобретение навыков работы на ЭВМ и изучение элементов логики, необходимых как для освоения основ вычислительной техники и начал программирования, так и для освоения методов поиска и накопления информации в базах данных и базах знаний. Отсутствие разделов, связанных с изучением элементов логики, на наш взгляд, являлось недостатком пробного учебного пособия А. П. Ершова, перешедшим в ряд других учебников.

Естественно, изучение элементов логики в средней школе — это серьезная дидактическая проблема. Мы сами видим и знаем, что здесь еще много нерешенных проблем. В этих целях при доработке учебника мы попробовали воспользоваться в качестве тренажера диалоговыми интерпретаторами языка Пролог на школьных ЭВМ. И тут мы будем признательны всем, кто сообщит о результатах применения языка Пролог в обучении информатике и подскажет нам предложения по совершенствованию нашего учебника, поскольку соответствующими зарубежными данными мы не располагаем.

В то же время мы считаем принципиальным освоение языка Пролог с одновременным изучением логики и принципов логического вывода. Точно так же мы считаем, что изучение начал программирования с использованием любого языка — Бейсик, Паскаль, Модула, Си и т. д.— необходимо взаимоувязывать с изучением основ алгоритмизации и использовать для этого псевдокод — язык для описания структурированных алгоритмов, одна из версий которого была принята А. П. Ершовым в качестве своего учебного алгоритмического языка. Для обучения этим языкам — языкам описания алгоритмов — можно использовать самые различные тренажеры, и лучшим среди них, безусловно, является Е-практикум.

Однако нельзя забывать и о подготовке будущих программистов. За рубежом в этих целях используются языки структурного программирования — Паскаль, Модула, Си или псевдокод для традиционных языков программирования — Фортран, Пл/1, Бейсик и Ассемблер. Во втором случае псевдокод используется и как средство обучения, и как средство документирования программ, не зависимое от применяемых языков программирования. Именно этот подход реализован в нашем пробном учебнике. Он допускает простоту перехода на другие языки программирования и углубленное изучение основ программирования вплоть до создания творческих коллективов учащихся для выполнения различных заказных программ.

И все-таки необходимо вернуться к логике и обсуждению ее законов, о чем пишет В. Белошапка. Соблюдение законов логики — законов тождества, непротиворечивости, исключенного третьего и достаточных оснований — является базой любого научного обоснования. Первый среди них — закон тождества — требует согласования предмета обсуждения. И бол е всего от нарушений этого закона страдает информатика как учебная и как научная дисциплина.

Типичным нарушением этого закона является сведение информатики как учебного предмета к изучению программирования на каком-то из языков для персональных ЭВМ. Другим нарушением — сведение курса информатики только к изучению основ алгоритмизации. Сложность определения предмета информатики как учебной дисциплины связана с динамикой развития вычислительной техники, ее программных средств и индустрии накопления, обработки и передачи информации с использованием ЭВМ.

В ближайшие пять лет в содержании информатики как учебной дисциплины, на наш взгляд, стабилизируются следующие темы:

а) работа с ЭВМ,
б) информационные технологии,
в) начала программирования,
г) основы алгоритмизации,
д) решение задач на ЭВМ,
е) основы и применения ВТ.

Касаясь третьего закона Аристотеля об исключенном третьем, мы должны признаться, что в учебнике была сделана оговорка о том, что этот закон не является общезначимым. Соответствующие контрпримеры могут быть приведены из области математики и из житейской области. Тут мы должны согласиться с В. Белошапкой, в частности с парадоксом о «лжецах», и хотим разобрать предложенный им контрпример, поскольку он приводит его уже второй раз.

Нам кажется, что если кто-то утверждает, что он лжет, то тут можно высказать три сентенции. Во-первых, это — его личное дело и его нужно рассматривать не столько с позиций логики, сколько с позиций этики и межличностных отношений. Во-вторых, для того чтобы доказать, что кто-то говорит неправду, необходимо указать среди утверждений, высказываемых этим лицом, утверждение, не соответствующее действительности. Причинами тому могут быть либо незнание предмета обсуждения, либо нарушения законов логики, либо некоторый умысел.

В частности, утверждение, приведенное в статье В. Белошапки,— «Я — лгу», имеет два толкования: «всегда лгу» или «иногда лгу». Первое безусловно неверно — доля правды в нем есть. Но верна вторая интерпретация, поскольку человек, который говорит правду и только правду, такого сказать не может. Главный вопрос здесь, на наш взгляд, в соотношении понятий истины и правды и взгляд на них с точки зрения информатики. Всякая ложь — это просто недостоверная информация. Достоверную информацию мы принимаем как истину, но не всякая истина — это правда. Бывает же полуправда, когда человек не все договаривает. Вопрос тогда — почему он так делает?

Более тонкое положение возникает с утверждениями, для которых высказывающее его лицо не может изложить достаточного обоснования. В частности, мы уже показали, что утверждение о том, что учебник, получивший на конкурсе 2-е место, удовлетворял требованиям, предъявлявшимся в то время к учебникам. В то время как в статье Белошапки не раскрыто,— какие именно требования к учебнику он имеет в виду.

Авторы учебников были бы признательны В. Белошапке, если бы он раскрыл понятие термина «научность» применительно к оценке учебников по информатике. По-видимому, он имеет в виду соотношение материалов в учебниках к научным положениям информатики как фундаментальной научной дисциплины. Но многими специалистами отвергается признание информатики как научной дисциплины, подобно тому как долгое время отвергалось признание кибернетики.

Однако в отличие от кибернетики информатика имеет явно выраженный предмет — изучение информационных процессов в природе, обществе, в общении людей, в живых организмах, в технических устройствах и, наконец, в работе вычислительных машин. А. П. Ершову принадлежит определение информатики (в узком смысле слова) как науки, изучающей законы и методы накопления, обработки и передачи информации в ЭВМ.

Соответствующим математическим аппаратом являются законы и положения математической логики, теории алгоритмов и дискретной математики. Этот аппарат в элементарных частях раскрыт в нашем учебнике в той мере, которая доступна учащимся средних школ. Более того, в нем в элементарном изложении приведены элементы новейшего направления современной теории алгоритмов — основы основ доказательного программирования и теоретической информатики — основы конструирования баз знаний.

В нашем понимании «научность» — это степень раскрытия и сохранения основных научных положений научной дисциплины. Мы лично считаем, что в нашем учебнике уровень «научности» выше, чем это может быть освоено учащимися в современной средней школе. Именно в этом был наш расчет при участии в конкурсе учебников — занять 2-е место и получить рекомендацию для использования учебника в математических школах. Наш опыт показывает, что материалы учебника в полном объеме доступны учащимся математических школ. А для освоения учениками обычных средних школ элементов новейших технологий накопления и обработки информации с помощью и с использованием ЭВМ необходим полный пересмотр содержания и методики преподавания математики на основе широкого применения в этих же целях ЭВМ.

Поэтому мы считаем, что с научной точки зрения правильно говорить не о «научности», а о научном уровне изложения учебных материалов. И тут вы меня простите, но в оценке сложности доказательного программирования в статье В. Белошапки содержится ряд ошибок. Во-первых, заявлено, что в учебнике анализ правильности алгоритмов проведен только для одного

алгоритма с тремя присваиваниями, а затем признается, что для алгоритмов с циклами разбирается техника доказательств, основанная на применении математической индукции, но тут же заявляется, что этого недостаточно в общем случае.

Да, действительно, построение доказательств правильности — достаточно трудоемкая деятельность, требующая и развивающая у учащихся определенную математическую культуру. Да, действительно, для любого алгоритма решения прикладных задач может быть представлено доказательство его правильности, и это требует определенных усилий и знаний. Однако наш опыт показывает, что размер доказательств обычно не более, чем в два раза превышает размер описания алгоритмов, и после разбора двух-трех сложных примеров ребята приступают к самостоятельным попыткам построения доказательств. Но зато после этого они знают, как исчерпывающим образом проверить правильность алгоритмов и программ и как разрабатывать алгоритмы и программы, чтобы упростить доказательства и полностью устранить все ошибки из своих программ [3].

Хотелось бы надеяться, что редакция журнала «Информатика и образование» найдет возможность опубликовать развернутые методические материалы по технологии решения задач на ЭВМ, элементам доказательного программирования и методам проектирования баз знаний, предусмотренных программой по нашему пробному учебнику, а также материалы по методам систематического конструирования учебных планов и программ с одновременным обоснованием их правильности и реализуемости в тех или иных учебных заведениях.

Литература

1. Основы информатики и вычислительной техники. Ч. 1 и 2: Пробное учебное пособие для IX—X классов / Под ред. *А. П. Ершова* и *В. М. Монахова*. М.: Просвещение, 1985, 1986.

2. *Каймин В. А.* и др. Основы информатики и вычислительной техники: Пробный учебник для X—XI классов. М.: Просвещение, 1989.

3. *Каймин В. А.* Основы доказательного программирования. М.: Изд. МИЭМ, 1987.

КНИГИ

Разработчикам ППС

В настоящее время ведется активная разработка педагогических программных средств (ППС ВТ). Процесс их создания сложен и требует знаний в области педагогики и методики преподавания, психологии, эргономики, дизайна. Помочь разработчикам призвана книга «Разработка педагогических программных средств вычислительной техники для учебных заведений профтехобразования», которая вышла в издательстве «Высшая школа».

В ней описываются психологические, эргономические, дидактические и технические требования к ППС ВТ, предлагается одна из возможных технологий их создания. Рассматриваются особенности разработки ППС ВТ для теоретического и производственного обучения учащихся профтехучилищ, для подготовки инженеров и системы повышения квалификации.

В книге подробно описываются требования к оформлению программной документации при разработке ППС ВТ, приводятся примеры педагогических сценариев.

Заявки можно направлять по адресу: 191119, Ленинград, ул. Черняховского, 2, ВНИИ профтехобразования, книжный склад. Книга высылается наложенным платежом.

Кристоф Пармантье, занимающийся проблемами компьютеризации начального образования в группе известного психолога Ж. Вернье (ученика и коллеги Ж. Пиаже), представляет современную французскую школу педагогической психологии. Исследования К. Пармантье отличает практическая направленность: он пришел в университетскую науку от учительского стола. Для публикации в «ИНФО» выбрана статья, где одно из исследований автора вписывается в контекст общего состояния компьютеризации начального образования во Франции. Русский вариант статьи несколько сокращен за счет французских реалий, мало информативных для советского читателя, и некоторых общих заключений, бесспорных в условиях нашей страны.

К. ПАРМАНТЬЕ
Лаборатория психологии развития и воспитания детей,
Парижский университет

Информатика во французской школе

Во Франции дети посещают школу с 6 до 16 лет. В большинстве городов детские сады (материнские школы) принимают детей начиная с 3 и даже с 2 лет. С 6 до 11 лет дети посещают учебное заведение, с которым французы связывают название собственной школы (наиболее близкий советский эквивалент — начальная школа). В каждом классе занятие ведет один учитель. Затем в течение четырех лет ребенок продолжает обязательное обучение в колледже. Здесь учителя специализированы по предметам: французский, математика, история, география... Некоторые из учеников имеют возможность начать предпрофессиональную подготовку.

Наконец, ученики могут завершить (в течение трех лет) среднее образование в лицее, выбрав либо общеобразовательное направление, либо предпрофессиональную ориентацию. После обучения в лицее они сдают экзамены на степень бакалавра. Диплом бакалавра дает право продолжить образование в высшем учебном заведении. Процент учащихся, стремящихся стать бакалаврами, возрос с 26 % в 1969 г. до 50 % в 1989 г.

Многообразные направления среднего профессионального образования готовят молодых людей, которые не планируют получения высшего образования, к более быстрому включению в активную жизнь.

Анализ французских достижений в области компьютеризации образования может быть понятным и продуктивным, если оценивать их с двух позиций. С одной стороны, для современной французской школы характерно применение коммуникационных сетей и новых информационных технологий обучения. С другой стороны, педагоги Франции с глубоким уважением хранят восходящие к Руссо традиции: уважение и изучение личности ребенка, учет его возрастных особенностей, индивидуальных склонностей в воспитании и обучении. Этим традициям в полной мере отвечают последние работы по когнитивной психологии, которые ставят ребенка в центр педагогических дискуссий.

Внимательное отношение к обеим точкам зрения позволяет понять растущий интерес к исследованиям в области педагогической информатики в начальной школе.

Включение информатики в общее образование началось во Франции лишь после коллоквиума Европейской организации дидактических конференций (OCDE) в Севре в 1970 г., где, несмотря на многообразие научных позиций участников, было выработано понимание неизбежности воздействия информатики на школьное образование. В 80-е гг. после активных дискуссий вокруг вопроса, должен ли компьютер прийти в

школьное образование как средство обучения (например, модное в то время программированное обучение) или как объект изучения (преподавание программирования), появляются первые официальные документы по проблемам компьютеризации начального обучения.

В марте 1983 г. Министерством национального образования разработан документ, в котором впервые сформулированы основные направления компьютеризации начальной школы — этапа «ознакомления с информатикой»*. В основу документа положены «три подхода к компьютерному феномену:

гуманитарный и социальный (воздействие компьютера на человека и общество);

технологический, включающий две компоненты: практическое использование компьютера и его роль как средства управления и реорганизации научных областей;

логический, включающий все аспекты программирования».

План «Информатика для всех», утвержденный в сентябре 1984 г., существенно конкретизировал ситуацию. Этот документ регламентировал обеспечение французских школ техникой — микро-ЭВМ Thomson** (не совместимых с IBM PC):

MO5 (8-разрядный процессор 6809-E Моторола; ROM — 24K байт, RAM — 48K байт, встроенный Бейсик);

TO7—70 (8-разрядный процессор 6809-E Моторола, ROM 24K байт, RAM — 48K байт, расширяемый до 124K байт; картриджи Бейсик и Лого, световое перо).

Эти машины поступали в школу как в автономных конфигурациях, так и сгруппированными в локальную сеть Нанорезо. Сетью управляет машина типа IBM PC (процессор 8088) с системой MS DOS. Коммуникационное устройство, включающее программные средства объемом 2K байта, связывает центральную машину с ученическими рабочими местами TO7 или MO5. В каждую конфигурацию включен монитор и, как правило, матричный принтер.

На пресс-конференции, посвященной провозглашению этого плана, министр говорил, что «речь идет о приобщении к информатике всех учащихся во всех областях Франции. Одиннадцать миллионов школьников получат доступ к ЭВМ во время обучения в школе: это реальный шаг в предоставлении всем учащимся равных возможностей».

Цифры из официальных источников свидетельствуют о том, что практически было сделано за три года:

33 171 упрощенная рабочая станция распределена в начальные школы;

более 13 000 сетей Нанорезо установлены в школах, колледжах и лицеях;

более 40 000 школ и 170 000 классов ощутили воздействие компьютеризации;

200 млн. франков (примерно 20 млн. рублей) израсходовано на покупку программного обеспечения;

50 млн. франков выделено для разработки средств компьютерных коммуникаций;

111 000 преподавателей прошли курсы недельной подготовки (из 300 000 направивших заявки в региональные органы образования).

Общая стоимость плана в конечном счете достигла значительной суммы — 1,79 млрд. франков.

В плане «Информатика для всех» количество машин, поступающих в ту или иную школу, определялось в зависимости от численности учащихся. На основе этой начальной государственной поставки многие городские советы (муниципалитеты) построили стратегию дальнейших инвестиций в техническое обеспечение компьютеризации местных школ.

Несмотря на успехи в осуществлении плана, следует остановиться и на некоторых неблагоприятных моментах.

Способ поставки технических средств в значительной мере определяет особенности их эксплуатации в школе. Сети устанавливаются не в каждой классной комнате, а в кабинете, которым пользуются многие классы, а иногда — несколько близлежащих школ. Поэтому преподаватель информатики вынужден предусматривать передвижение учеников из одного помещения в другое. Кабинет информатики, тем самым, получает в школе статут таких помещений, как бассейн, гимнастический зал или библиотека. Создается впечатление, что информатика (хоть и обязательна для посещения) — нечто среднее между факультативным досугом и школьной дисциплиной.

Исходя из требований безопасности и страхования, компьютеры размещают в защищенных от ограбления помещениях. Это требует дополнительных средств, которых не всегда хватает на простейшие предметы (мячи, краски и т. д.). В результате у учителей и детей возникает настороженное отношение к информатике. И вот уже потребность использовать компьютер приходится создавать, в то время как информатика долж а вписы-

---

* Здесь и далее цитируются официальные документы.

** В последние годы фирма Thomson, выпускавшая массовые компьютеры для системы народного образования и, в частности, оборудование локальной сети Нанорезо, была вынеждена уйти с рынка информатики. В настоящее время школы выбирают в качестве компьютеров либо стандарты PC, либо Эппл-Макинтош.

ваться в деятельность школы естественным образом.

Технические средства, поступающие в школу, мало приспособлены к столь интенсивному использованию, которое предполагается кабинетом информатики. В самом деле, единственный на школу (группу школ) кабинет, обслуживающий до четырех классов в одной возрастной параллели, оказывается загруженным по 6 ч в день. Ученики полны энтузиазма, нетерпеливы. Машины хрупки, недостаточно надежны, имеют слабую вентиляцию. Поэтому, несмотря на принимаемые меры предосторожности, после двух лет эксплуатации одна ученическая машина из четырех выходит из строя. К тому же из-за низкой стоимости изготовления проблемы технического сопровождения трудноразрешимы! Такая обстановка порождает недоверие к компьютеру. Вот высказывание одного девятилетнего ученика: «Я лучше буду делать с ЭТО (создавать процедуру Лого), потому что, если она (машина), ошибется, то вся работа будет понапрасну».

Последствия всего этого нельзя оценить однозначно: не все факторы поддаются количественному учету. Только широкие (возможно, международные) исследования позволят сравнить эффективность разных решений. Тем не менее, поскольку сложившиеся условия обучения информатике сохранятся по крайней мере до тех пор, пока нынешнее поколение учащихся достигнет зрелости, последствия этих решений нельзя не учитывать при планировании будущих действий.

С появлением в 1985 г. нового (модифицированного по сравнению с документом 1983 г.) варианта основных направлений завершился этап «ознакомления с информатикой». Однако информатика не уходит из учебной программы, а в соответствии с ранее сформулированными подходами появляется в трех важнейших разделах:

чтении и практике письменного языка: компьютер следует рассматривать «как средство воспроизведения знаков... Учитель не должен забывать, что обучение чтению и письму можно вести с помощью экрана телевизора и вычислительной машины»;

науке и технике: «...важность информатики оправдывает выделение ей по крайней мере 50 ч в курсе средней школы»;

математике: «...использование компьютера для решения арифметических и геометрических задач в средней школе, особенно в возрасте 8—10 лет, побуждает учеников к поиску алгоритмов и развивает их логические способности».

В июне 1986 г. в дополнение к основным направлениям 1985 г. появляется директивный документ, «имеющий целью дать общее представление о месте информатики в школе». В нем отражены две точки зрения на информатику: компьютер как средство и как объект.

Рассматривая компьютер как средство обучения, документ подчеркивает: «его использование не регламентируется какими-либо указаниями», «учитель свободен в выборе педагогических методов» и «в любом случае решающим является не само учебное программное средство, а способ работы с ним». Варианты использования компьютера как средства обучения объединены в четыре группы:

помощь в простых педагогических актах, таких, как упражнения, повторение, закрепление и контроль усвоения;

предъявление учебной информации и моделирование изучаемых процессов и явлений;

работа с редактором текстов и другими прикладными программами;

использование некоторых языков, например Лого, позволяющих реализовать оригинальные подходы к конструированию знаний и формированию способов рассуждений.

Вторая точка зрения, характеризующая компьютер как объект изучения в школе,— это позиция, с которой рассматриваются технологическая, научная и социальная перспективы школьной информатики.

Процитируем, наконец, последние абзацы этого документа: «Учитель должен отчетливо понимать, с какой целью он обращается к компьютеру — для усвоения понятий информатики или для усвоения знаний, относящихся к другим дисциплинам. От этого существенным образом зависит, как оценивать полученные учащимся знания».

Официальные педагогические инструкции коснулись также и обучения детей в детских садах, в частности (январь 1986 г.): «Всесьма эффективным может быть использование в учебном процессе детских садов всевозможных информатизированных дидактических материалов, таких, как роботы и автоматы (Черепашка, программируемые игрушки и т. д.)». Об этом свидетельствуют, в частности, работы Р. Коэн.

Работ[illegible] ученых

Четыре группы исследователей в первую очередь связаны с внедрением вычислительной техники в школу.

Программисты. К дискуссиям и подготовке различных проектов по информатизации образования правительственные учреждения широко привлекают специалистов в области программирования. Так, работы В. Меркулова и Ж. Ебенстрейта заложили основы компьютерной педагогики и позволили создать оригинальный учебный язык программирования с французской лексикой — LSE. Из их работ следует вывод, что преподавание программирования как самостоятельной части программы в начальной школе не представляется ни полезным (для всех), ни необходимым в этом возрасте, учитывая темпы технологических перемен.

Однако умы педагогов продолжают волновать опасения, выраженные известным фантастом Дж. Оруэллом и резюмированные видным французским специалистом в области информатики Ж. Арсаком: «Невежество масс широко открывает двери компьютерократам».

Психологи. Изучая интеллектуальные механизмы, а подчас и медицинские аспекты относящихся к информатике видов деятельности до и после общения с компьютером, психологи предлагают и используют целый ряд ценных моделей. Ж. М. Хок, например, выявляет когнитивные подходы, применяемые в планировании как виде деятельности, различая, в частности, планы процедурные и декларативные. Касаясь в большей мере проблем детства и подросткового возраста, П. Мендельсон посвятил свои работы описанию, анализу и моделированию когнитивных процессов, имеющих место в программировании. Эксперименты, проведенные в таких центрах, как, например, Клиника биопсихопатологии ребенка им. Св. Анны в Париже, показывают, сколь благотворное терапевтическое воздействие на ребенка может оказывать общение с компьютером.

Педагоги. Различают два типа педагогических подходов к проблеме: с одной стороны, это подход дидактов, разрабатывающих содержание обучения, с другой стороны — педагогов, работающих над совершенствованием существующих методик.

В своих исследованиях Ж. Верньо, прямой продолжатель идей Пиаже, рассматривает элементы введенного им понятия концептуального поля: знания и навыки формируют модель, с помощью которой можно понять технологию изучения и освоения новых знаний. Изучение состояния концептуального поля, по Верньо, должно предшествовать всякому дидактическому исследованию. Ему удалось показать эффективность нового понятия. Анализ концептуального поля, связанного с информатикой и возникающими здесь дидактическими проблемами, определяет рамки преподавания программирования в школе.

Важно отметить, что теоретические результаты этих исследований находят прямое отражение в практике педагогической работы. Национальный институт педагогических исследований и Министерство национального образования предприняли ряд мер, способствующих адаптации к быстро меняющейся технологии и создающих на базе указанных представлений благоприятные условия для обучения программированию в начальной школе:

перевод Лого на французскую лексику;

создание аппаратно реализованных Черепашек;

разработка учебных программных средств, использующих понятия программирования (например, Шлюз, Лифт — учебные прикладные программы, широко известные по Франции);

управление программируемыми роботами.

Другие исследователи (Виве, Боссю) подчеркивают важную роль учителя в условиях использования новых информационных технологий обучения.

Социологи. Внимание к проблеме недопустимости «отставания школы в удовлетворении социальных заказов общества» привлекалось в целом ряде исследований, посвященных процессам информатизации. Среди них выделяются работы Нора-Минка и Ж.-К. Симона. Все же практические выводы о преподавании информатики могут быть сделаны только после серьезных предварительных педагогических обсуждений. «Кнопочное» обучение, использование программных безделушек неизменно толкает преподавателей, даже привыкших следовать за модой и министерскими установками, к активному сопротивлению таким методикам.

Учебные заведения с опаской относятся к компьютерному обучению, полагая, что оно опирается на методики «бедные и к тому же ограниченные возможностями технических средств». Тем не менее следует отметить, что программные средства компьютерного обучения могут найти применение при формировании ряда умений и навыков. Эти средства построены на использовании возможностей языков программирования: последовательностей, циклов, подпрограмм, итераций. Эти механизмы программирования, полезные для построения методического обеспечения учебного процесса, позволяют опосредовать определенные умения.

Как обеспечить наилучшее использование ресурсов компьютера

Обсуждение можно вести в двух главных направлениях:

А — техническое направление, на котором возможности вычислительных машин могут быть расширены за счет их объединения в пары, группы или организации сетей, а также за счет использования новых программных средств и новых периферийных устройств;

Б — гуманитарное направление, на котором обеспечивается педагогическая эволюция учебного процесса; к наилучшему использованию компьютера идут через исследования как в области содержания, так и в области теории обучения.

А.1. Сети. Франция имеет развитую информационную сеть, потенциальные возможности которой могут быть использованы и в педагогических целях. План «Информатика для всех» акцентирует внимание именно на этом аспекте педагогических применений компьютеров, устанавливая сети Нанорезо во многих школах. Возможности компьютера увеличиваются за счет объединения машин, позволяющего организовать диалог между учительским компьютером и ученическими рабочими местами. Педагогическая эффективность сетей пока еще фактически не оценена, не проведены сравнительные научные исследования, которые позволили бы оценить результаты, полученные на сетях.

Различают два уровня использования сети: локальную сеть класса и соединение класса с внешней средой.

В кабинете информатики архитектура системы, к сожалению, не позволяет вести диалог учеников, находящихся на своих рабочих местах. Все сообщения необходимо передавать через центральную машину. Тем не менее целый ряд приемов и педагогических подходов были опробованы в Нанорезо с использованием некоторых специфических возможностей этих сетей. Например, в Высшей нормальной школе (пединституте) в Версале используют сеть на уроках по машинной графике и конструированию с учениками в возрасте от 8 до 10 лет. С помощью графического редактора, в котором используется световое перо, каждый ученик или группа на своих рабочих местах вносит свою лепту в разработку, частичную реализацию и оценку коллективного графического проекта.

Некоторые программные средства были созданы специально для Нанорезо. Например, пакет прикладных программ «Журналист» позволяет моделировать работу агентства печати, выпускающего газету. В соответствии с принципами педагогики Френе ученики распределяют между собой рабочие места, функциональные задачи и готовят статьи, которые затем группируются при коллективном выпуске газеты. Одна из первоочередных задач в информатизации школы — широкое распространение сетевых конфигураций, чтобы наилучшим образом использовать возможности общения между ученическими машинами.

Стены перестают быть границами, когда один класс получает возможность общения с другим посредством телекоммуникационных сетей. Используя модемы, можно соединить вычислительные машины, расположенные в разных классах. Новый носитель письменности является дополнительным стимулом для обмена информацией между учениками. Эта технология позволяет через классный компьютер обращаться за консультацией в центр документации или пополнять базу данных центра. Использование компакт-дисков облегчит подобные коммуникации.

А.2. Программные средства. В результате научных исследований появляются все новые средства программного обеспечения. Как следствие, эволюционируют и их педагогические приложения.

Автоматизация конторской деятельности привела к созданию ряда инструментов — средств обработки текстов, электронных таб-

лиц, систем управления базами данных, применимость которых в школьной практике тоже стала предметом изучения. В частности, Министерством национального образования была создана группа для обсуждения программных продуктов такого типа. Разработаны адаптированные школьные версии текстовых редакторов, включающие словари и редакторы планов (TGV-Текст или ПИСАТЕЛЬ). Другие адаптации ориентированы на работу с текстами специфических жанров. Например, пакет программ СКАЗКА позволяет «сочинять» сказки, генерируя их из заранее заготовленных (стандартных) элементов.

Перспективны и другие учебные прикладные программы. После волны популярности авторских систем все больше внимания уделяют системам искусственного интеллекта (ИИ). Пока внедрены лишь несколько реализаций, поскольку стоимость разработки весьма велика. По своей форме программные средства, используемые ИИ, привносят в прикладные педагогические исследования идеи, относящиеся к моделированию некоторых знаний, так же, как в реальной жизни учитываются действия преподавателя и учащегося. Два типа программ иллюстрируют эти соображения:

средства методического обеспечения построены на использовании экспертных систем. Они ориентируются, в первую очередь, на передачу знаний, зависящих от конкретной предметной области, например, энциклопедической (пример типовой учебной программы — распознавание грибов) или иерархизированной по древообразной модели (пример программы — поиск в генеалогическом дереве). Экспертные системы извлекают знания из специфических правил рассматриваемой области, которые все же должны подчиняться строгому синтаксису, как, например, синтаксис условий ветвления;

внедрение экспертных систем позволяет также вмонтировать средства помощи учителю в состав программно-методического обеспечения учебного процесса. Такие системы позволили наметить эволюционный переход от программированного обучения к интеллектуальному обучению с помощью ЭВМ. Экспертная система может, например, обработать на основании диагностики ошибку ученика и обоснованно предложить ему средства обработки. Такие исследования ведутся в лаборатории информатики в Ле Мане. Другие перспективы открываются, например, автоматическим генерированием упражнений на основе модели ученика. Базовая система знаний реализована на языке SNARK, она работает на естественном языке, исходя из формул (в зависимости от объема и области знаний), а также варьируемого уровня трудности.

Вычислительные машины пятого поколения являются объектом интенсивных исследований. Они позволят использовать экспертные системы на промышленном уровне. Эти новые технологии окажут воздействие на все области, включая учебное программно-методическое обеспечение.

А.3. Периферийные средства. Компьютер управляет различной периферией в кабинетах информатики, и прежде всего принтером. С успехом могут быть использованы и другие устройства. Национальный институт педагогических исследований изучает возможности лазерных дисков, позволяющих интегрировать компьютер с аудиовизуальными средствами. Компьютер управляет многими лабораторными классами для изучения иностранных языков. В Тулузе, Лионе, Дижоне и Ле Мане исследователи занимаются педагогической робототехникой, эти работы ведут преподаватели-практики или специалисты по информатике в рамках курсов обучения техническим специальностям. В качестве эксперимента наборы элементов аппаратных роботов распространяются по школам. При наличии адаптированных интерфейсов с их помощью дети конструируют роботы и управляют ими. Во Франции, как и всюду на Западе, на рынке конструкторов автономных и программно управляемых игрушек и роботов конкурируют известные фирмы Лего и Фишер.

Однако качественное осуществление этого вида деятельности требует высокого уровня квалификации в области электроники. Зачастую учителя имеют лишь самую малую подготовку и опыт в этой специфической области. Представления о технических компонентах воспринимаются ими и, следовательно, передаются ученикам на уровне «черных ящиков». Такой подход противоречит желанию исследователей педагогического процесса сформировать компетентных пользователей. Все же реализуемый многочисленными роботами переход от интерпретации мира двух измерений, каким он предстает на экране, к трехмерному миру мог бы облегчить понимание геометрических и алгебраических проблем, связанных с понятием объема или третьего измерения.

Наконец, соединение компьютера с мощными периферийными средствами (визуально воспринимаемые команды, распознавание речи и звуковой синтез и т. д.) может способствовать возвращению к активной жизни детей с некоторыми недостатками и включению инвалидов в сферу пользователей ЭВМ. Тем самым многие педагогические задачи, касающиеся детей с физическими недо-

статками, могут быть возвращены школе.

Б. Гуманитарное направление. Интеллектуальные возможности человека в отличие от машины с ее техническими ограничениями практически беспредельны. Поэтому в человеко-машинном общении весьма важен гуманитарный фактор. Используя новые педагогические подходы, адаптируя их к техническим средствам, преподаватель может рассчитывать на дополнительную эффективность вычислительных машин.

Б.1. Моделирование. Вычислительные машины позволяют моделировать алгоритмы, которым подчиняются некоторые бытовые ситуации, законы. Моделирование, которое уже часто используется на уроках по различным предметам в средней школе, может иметь конкретные педагогические цели. Речь идет о том, чтобы ученик мог провести приближенный эксперимент, моделирующий реальную ситуацию, которую ему было бы опасно или даже невозможно изучить другим путем. Если военные науки активно оперируют методами машинного моделирования, то учебное программное или методическое обеспечение школы в этом направлении продвинуто очень мало.

В учебной практике встречаются устройства, моделирующие управление велосипедом или автомобилем. Задуманные как игровые автоматы, они позволяют ученику экспериментировать, генерируя ситуации на многочисленные правила уличного движения. Существуют другие модели, занимающие промежуточное положение между игрой и обучающей программой. Они позволяют юному гражданину, например, уяснить некоторые социальные механизмы окружающего его мира. Выборные кампании, экономика рынка становятся в этих случаях поводом для применения вычислительной машины, которая обучает школьника, скрывая от него собственную специфику. Расширяя понятие модели, можно иллюстрировать различные сферы деятельности. Пакет прикладных программ «Шлюз», разработанный в Министерстве национального образования, предлагает возможности моделирования шлюза с помощью набора макропримитивов, смысл которых понятен: заполнить, слить, открыть ворота, закрыть ворота и др. Такого рода пакеты стимулируют мышление, и в этом их педагогическая ценность.

Однако использование подобного программного обеспечения формулирует проблему разрыва между реальностью и миром модели и, в конечном счете, условий переноса знаний из действительной ситуации в модель и обратно. Как погруженный в реальность ученик узнает, что правила игры изменились, что речь идет уже не о модели? Могут ли быть сняты некоторые априорные запреты, когда педаль акселератора становится объектом игры? Конструируется другой эксперимент из реального мира, и еще только предстоит оценить последствия подобного переноса. К сожалению, немногие исследования рассматривают подобные ситуации, и социальный профиль ученика, который получит преимущества от такой формы обучения, нечетко определен. Пока в этом направлении не будут проведены исследования, преподавателям-экспериментаторам следует учитывать, что некоторые модели могут оказаться не оправданными с точки зрения их применимости в классе.

Б.2. Метод переноса. Наряду с моделированием обучение с помощью метода переноса, начинающееся экспериментами в концептуальных полях, главным образом математических или лингвистических, является одним из актуальных поисковых направлений. Многие такие исследования уже начаты. Обязательным здесь является очень точное определение содержания обучения и условий переноса. Примером может служить экспериментальное исследование, которое ведет автор в Лаборатории психологии развития и воспитания ребенка.

Обсуждаемый эксперимент проводится в рамках школьного курса программирования и имеет целью изучить эффективность метода переноса и взаимодействий с другими областями знаний. Обучение осуществляется в интерактивном режиме путем решения задач, когда ребенок усваивает не только представления, но и знания и умения, а также способность передавать знания, учить другого (умение алгоритмизировать изучаемый процесс). Обучение начинается с формулирования постулатов о реальной действительности, необходимых при конструировании схем, которые могут быть построены с помощью теорем «в действии» (теорем, которые приводят от действия к формализованным выводам). Эти теоремы вытекают из распознавания инвариантов, или, более широко, из концептуальных аспектов, выделенных, классифицированных и организованных с помощью свойств и отношений на основе некоторых предположений, которые считаются истинными в реальном мире. Усвоенные навыки могут быть использованы при выполнении формально сходных задач, это предполагает явное или неявное распознавание инвариантов. Именно это распознавание позволяет по-настоящему овладеть методом переноса.

Эксперимент затронул 48 учеников в

8—9 лет из двух классов одной из школ в пригороде Парижа. Такой возрастной уровень был выбран для эксперимента потому, что, с одной стороны, ученики к этому времени еще не обращались к информатике, с другой стороны, этот уровень — начало обязательного знакомства с информатикой в рамках школьной программы. Кроме того, дети в этом возрасте мало знакомы с используемыми в эксперименте геометрическими понятиями.

Были построены и проверены по репрезентативному образцу задачи-тесты, которые позволяют учесть состояние осведомленности при вычислении периметров и углов. К тому же они позволяют сделать некоторые выводы о способности итерировать последовательность или различать правую и левую стороны на ориентированной плоскости. Наконец, они оценивают некоторые умения алгоритмизировать в Лого. После первого тура испытаний ученики были разделены на две одинаковые группы — А и Б. Затем они одновременно прошли две заранее предопределенные серии занятий, длительностью по 12 ч каждая, но в разном порядке. В конце каждой серии они вновь проходят одинаковые испытания. Результаты, полученные в ходе этих испытаний для 24 учеников, познакомившихся сначала неявно с некоторыми геометрическими понятиями благодаря изучению программирования средствами графики Лого, а затем явно, без вычислительной машины, сравнивались с результатами 24 других, которые изучали те же понятия в обратном порядке по следующему плану:

Тест 1		Тренировка	Тест 2	Тренировка	Тест 3
	Группа А Группа Б	Лого Геометрия		Геометрия Лого	

Занятия были спланированы с учетом школьного расписания. В серии занятий по Лого после двух уроков знакомства с фундаментальными примитивами языка предлагается нарисовать на экране фигуры из заданного набора в порядке возрастающей трудности. Этот порядок позволяет усвоить сначала понятие цикла, а затем и определение процедуры. Предложенные фигуры — это многоугольники, фигуры, составленные из многоугольников, отрезки, прямые и непрямые углы. Серия занятий по геометрии без компьютера подводит учеников к понятию угла, которое используется как элемент для описания ситуации. В сочетании с другими элементами разрабатывается классификация многоугольников и фигур, включенных в программу.

Лого приводит к линеаризации изображения, т. е. преобразует его в контур, который надо обойти; затем проводится обсуждение о переводе обхода контура в программу. В Лого необходимо учитывать длины сторон. Этим можно объяснить более уверенное вычисление периметра учениками группы А в тесте 2. Кроме того, существует тесная связь между умножением и циклом. В ходе разработки программы, рисующей квадрат, переход от

[ВП 30 ПР 90 ВП 30 ПР 90 ВП 30 ПР 90 ВП 30]

к

ПОВТОРИ 4 [ВП 30 ПР 90]

предполагает, что учитывается равенство четырех сторон квадрата. Вычисление его периметра может выражаться как переход от 30+30+30+30 к 4×30. Усвоение циклов, работа с ними в Лого способствует знакомству в форме «теоремы в действии» с инвариантами многоугольников. Количественное увеличение успешных результатов приводит к качественному улучшению.

Лого — это система без единиц измерения, где команды записываются следующим образом:

ВП 30 ПР 30

Учащийся должен понять, что сама команда содержит понятие единицы измерения. Наименьший геометрический элемент, обрабатываемый на листе бумаги,— это точка. На экране это пиксель (точка растра). Разрешение экрана определяет значение реальной единицы измерения, что составляет известную трудность для ученика. С другой стороны, система измерения углов в Лого основывается на делении круга на 360°. Действие примитивов поворота неоднозначно в системе с основанием 360, где значения 0 и 360 дают одинаковый результат. Работа с углами в Лого опирается, таким образом, на понятия деления и основания системы счисления и может порождать продуктивные гипотезы при переходе к безмашинной геометрии. Тем не менее эта работа не приносит пользы слабым школьникам. Обработка результатов тестов и наблюдения за детьми между тестами свидетельствуют о том, что образовательные качества Лого оказываются эффективными главным образом для сильных учеников.

Заключение

Чтобы в попытках повышения эффективности образования перешагнуть через известные границы, следует учитывать и фактор человека (в первую очередь), и исполь-

зуемые им инструментальные средства. Как и в других странах, художники Франции — музыканты, скульпторы, архитекторы, дизайнеры, писатели — не раз обращались в своем творчестве к компьютеру. Образование должно воспользоваться возможностями информатики не только для того, чтобы научить детей ценить новое в искусстве, но главным образом для того, чтобы это новое стало источником вдохновения, служило для выражения их собственного внутреннего мира. «Компьютер — средство самовыражения человека, и, если у него нет пока своих шекспиров, микеланджело или эйнштейнов, они появятся в будущем» *(С. Пейперт)*.

Материал подготовил докт. пед. наук
Ю. ПЕРВИН

Авторский вариант статьи содержит объемный список литературы на французском языке. Желающие ознакомиться с ним могут обратиться в редакцию.

ЧТО МОЖЕТ ЭВМ?

Новая опасность

Излучение высоковольтного ускорителя электронов — кинескопа — и воздействие высокочастотного электромагнитного поля перестали пугать пользователей компьютеров: по всей видимости, приносимый ими здоровью вред минимален. С ухудшением же зрения легко бороться, используя высококачественные мониторы.

Однако эти мониторы оказались источником новой опасности — низкочастотных электрического и особенно магнитного полей.

Еще в 1979 г. было показано, что дети, проживающие вблизи высоковольтных линий электропередач, вдвое чаще среднего умирают от рака. Исследователи тогда связали это явление с длительным воздействием магнитного поля частотой 60 Гц (американский стандарт частоты промышленного тока) и напряженностью 2 мГс. А ведь напряженность низкочастотного магнитного поля вблизи строчных трансформаторов некоторых дисплеев равно 9 мГс, а на расстоянии 30 см от экрана превышает 2 мГс!

Производители дисплеев, естественно, пытались игнорировать эти данные и объявить вредное воздействие низкочастотного магнитного поля недоказанным. Однако подтверждения опасности копились. В частности, при анализе причин смертей 438 тыс. рабочих было показано, что вероятность смерти от лейкемии выше нормы для 10 из 11 профессий, связанных с пребыванием в сильных электромагнитных полях (электрики, электроинженеры и т. п.). Вероятность заболевания лейкемией для работников Нью-Йоркской телефонной компании, проводящих много времени вблизи высоковольтных кабелей, оказалась в 7 раз выше, чем для остальных работников этой компании. Эти люди работали в поле напряженностью 4,3 мГс, а напряженность поля на расстоянии 30 см от 33-см экрана монитора Apple и дисплея Color-Page 15 фирмы E-Machines составляет 4—15 мГс.

Опыты на животных также подтвердили опасность магнитного поля низкой частоты.

Производители традиционных дисплеев на электронно-лучевых трубках пока не прилагают достаточных усилий для устранения «магнитного фактора». Впрочем, уже рекламируются модели, имеющие сниженный уровень излучения. Пока эта проблема не решена, можно порекомендовать пользователям ПЭВМ сидеть не ближе 1,2 м от чужих мониторов и 70 см от своего (магнитное поле значительно сильней сзади и с боков монитора).

Как решить задачу

Сколько будет: кажется, два умножить на примерно два? Ответить на этот вопрос однозначно ни один компьютер не сможет. А ведь в жизни вопросы чаще всего ставятся именно в такой форме. Если добавить, что при формулировке проблем часто используются такие неопределенные термины как «похожий», «быстро», «примерно», то становится понятно, насколько сложна работа программистов-прикладников.

Чтобы облегчить поиск решений для задач, начальные условия которых определены не полностью или расплывчато, а также упростить формализацию неопределенных систем, японская фирма Omron спроектировала программно-аппаратную компьютерную систему для работы с нечеткой логикой LUNA-Fuzzy RON. Система стоимостью чуть более 15 тыс. долларов (из них 4 тыс.— программное обеспечение) принимает решения на основе собственных умозаключений и сопоставления фактов. Никакой мистики здесь нет — качество выводов определяется умелым выбором для решения той или иной задачи одной из пяти заложенных в компьютер математических систем, которые используют минимаксную логику, весовые вычисления и еще три детерминированных метода.

Используя простые правила «если — то», другие логические операции и операции отношения, вы можете сформулировать до 300 правил, описывающих проблему в привычных вам терминах. А уж найти единственное в этих условиях решение поможет LUNA-Fuzzy.

Все свое ношу с собой

Энергия в наше время — это не только самолеты, автомобили и тепло в доме. Это еще и работоспособность многочисленных компьютеров, управляющих и самолетами, и светофорами, и мощными котельными. И от того, насколько они надежны, порой зависит наша жизнь. Но даже если энергия поступает постоянно, к сбою компьютера может привести работа мощного станка, сварочного аппарата и даже выключение холодильника.

Чтобы компьютеры не были столь зависимы от скачков сетевого напряжения, их снабжают автономными источниками питания, а также схемами, переключающими питание с сетевого на автономное, если напряжение в сети упадет ниже допустимого, и следящими за своевременной подзарядкой аккумуляторов.

Инженеры японской фирмы Mitsumi Electric разработали компактные источники питания в интегральном исполнении. В многоножечном корпусе расположен не только емкий аккумулятор с большим сроком службы, но и схема подключения источника тока к нагрузке и даже зарядное устройство.

Фирма начала массовое производство батарей поддержки в различных стандартных корпусах. Разработан и вариант толщиной 1 мм для питания интегральных карт памяти и электронных визиток. Примерная стоимость одной батареи — 3 доллара.

В соответствии с планом Ассоциация учителей информатики (АУИнформ) провела в конце прошлого года свои первые мероприятия.

Семинар Ассоциации учителей информатики

В ноябре в Миассе прошел первый семинар Ассоциации учителей информатики, посвященный вопросам использования ПЭВМ «Агат». На нем встретились преподаватели школ, техникумов, вузов, инженеры и программисты — 30 энтузиастов из разных уголков страны.

Участники встречи поделились опытом использования «Агата» в учебном процессе, обсудили различные методические и организационные подходы, обменялись программами. Большой интерес вызвали аппаратные доработки компьютера А. Н. Филиппова из Москвы и В. М. Ильинского из Павлодара. Интересные физические модели показал А. С. Шипилевский из Чебоксар. Хозяева представили программное обеспечение для новых учебников информатики (исполнитель «Робот» и система «Пролог-Д»). Одна из разработчиков системы «Школьница», В. А. Цикова из Новосибирска, продемонстрировала операционную систему для «Агата», не уступающую *MS—DOS* для *IBM PC*. Обсуждены вопросы создания в рамках ассоциации фонда программ, проведения независимой экспертизы программного обеспечения и аппаратуры, издания каталогов.

Материалы семинара собраны в сборник, который будет распространяться в виде брошюры или на дискете. Заявку на сборник (с указанием формата записи — 35 (80) дорожек или «брошюра») направлять по адресу: 428034, Чебоксары, а/я 99.

М. АЛЕКСЕЕВ

Первый командный турнир по информатике

В ноябре в Красноярске проведен финал первого Всесоюзного командного турнира по информатике, организованного АУИнформ совместно с Красноярским управлением народного образования. В турнире могли принять участие команды (по 4 человека) из школ, ПТУ, кружков информатики и т. д.

Турнир проводился в два этапа. Первый — заочный — определил 9 команд участников финала: школ № 57 Москвы, № 1 Чебаркуля, № 1 Чебоксар, № 3 Протвино, № 21 Иркутска, № 18 Миасса, № 3 Челябинска, № 14 Томска, № 10 и № 41 Красноярска, СПТУ № 6 Переславля. Каждая команда получила ПЭВМ «Ямаха», на решение отводилось 8 ч.

Жюри предложило участникам две задачи с предоставлением права самим выбрать одну из них. Задачи подбирались так, что была работа и аналитику, разрабатывающему алгоритм, и «хакеру», знающему машину «от и до».

Задачи всесоюзного командного турнира школьников по программированию

Задачи заочного этапа

№ 1, автор Алексеев А. В.

Вычислить сумму K-х степеней всех натуральных чисел, запись которых в десятичной системе счисления состоит ровно из l цифр, отличных от нуля.

Числа K *и* l вводятся.

Для проверки результатов работы программы найти возможно больше элементов в таблице результатов

l	K					
	1	2	3	4	5	...
1						
2						
3						
4						
5						
...						

Определить, при каких K и l программа выдает все цифры результата.

Задачи финального этапа

№ 1, автор Прохоров В. В.

PROLAN/M

Операционный блок новейшего *RISC*-микропроцессора концерна *NePhiggah* умеет выполнять единственную команду — контекстную замену в текстовой строке. Микропроцессор признает один тип данных — текстовую строку (более того, он оперирует лишь одной строкой переменной и «практически неограниченной» длины).

Микропроцессор работает с двумя устройствами памяти. Одно содержит программу обработки, другое — данные, т. е. текстовую строку R (вначале в нее заносится исходная информация, в последующем — промежуточные данные, а в конце — результат работы).

Программа для этого микропроцессора пишется на специальном языке *PROLAN/M* и имеет вид:

⟨комм⟩ (⟨$L1$⟩, ⟨$R1$⟩) ⟨комм⟩ (⟨$L2$⟩, ⟨$R2$⟩) ⟨комм⟩...
...(⟨Ln⟩, ⟨Rn⟩) ⟨комм⟩ (,)

Здесь ⟨Li⟩, ⟨Ri⟩ — некоторые текстовые строки, не содержащие символов ',', ')', причем ⟨Li⟩ — не пустые; ⟨комм⟩ — текстовая строка комментария (возможно, пустая), не содержащая символа '('.

Исполнение программы. Для заданной начальной строки R среди строк ⟨Li⟩ отыскивается первая, входящая в R. В случае, когда R не содержит ни одной ⟨Li⟩, исполнение программы завершается, а текущая R считается ее результатом. Если же поиск оказывается успешным, то вхождение ⟨Li⟩ в R (при неединственности самое левое) заменяется на соответствующее ⟨Ri⟩ (длины строк могут различаться) и все начинается с начала, но уже для изменившейся R.

Пример.

Программа «пример» (*dd, d*) (*ac, d*) (*ab*) (*c, ac*) (*de, END*) (*bc, t*) (,) — для исходного значения строки R *abcdabe* будет получен результат *END*.

Строка R в процессе работы этой программы последовательно примет значения *abcdabe cdabe cde acde dde de END*

1. Составьте на языке *PROLAN/M* следующие программы (⟨число⟩ везде обозначает десятичную запись натурального числа):

а. Копия. В результате работы программы содержимое *R* вида ⟨число⟩! должно быть преобразовано в ⟨число⟩#⟨число⟩.

Например, строка «1990!» должна превратиться в строку «1990#1990».

б. Инкремент. ⟨число⟩ ++ преобразуетяя в ⟨число+1⟩

Например, «1990++» превращается в «1991».

в. Разность. ⟨число1⟩ — ⟨число2⟩=? преобразуется в ⟨число1⟩—⟨число2⟩=⟨число1—число2⟩.

Например, «1990—1917=?» превращается в «73».

г. Полная разность. ⟨число1⟩ — ⟨число2⟩=? преобразуется в ⟨число1⟩ — ⟨число2⟩=⟨число1 — число2⟩.

Например, «1990—1917»=? превращается в «1990—1917=73».

2. Разработайте интерпретатор программ, написанных на языке *PROLAN/M*. Желательно предусмотреть режим трассировки исполняемых программ.

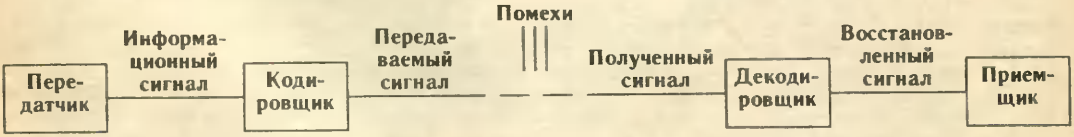
№ 2, автор Суэтов А. П.

Испорченный телефон

По длинной линии связи побитно передается информационный сигнал. По пути следования каждый бит сигнала с вероятностью *p* может быть искажен помехами. При искажении 1 заменяется на 0 и наоборот.

Для предотвращения потерь информации следует кодировать информационный сигнал до передачи и декодировать его после приема.

Процесс передачи и приема изображен на схеме:



Эксплуатационные затраты на передачу пропорциональны количеству передаваемых бит. Кроме того, приходится платить штрафы за неисправленные ошибки в восстановленном сигнале.

Общая сумма затрат выражается формулой:

$$\text{затраты}=C1\frac{k}{n}+C2\frac{i}{n},$$

где k — количество бит в передаваемом сигнале;

n — количество бит в информационном сигнале;

i — количество ошибок в восстановленном сигнале.

Программа должна запрашивать величины *p*, *C*1, *C*2. Для каждого набора этих величин может вводиться произвольное число информационных сигналов.

Коэффициенты *C*1 и *C*2 определяют цену передачи и размер штрафа.

Разработайте программу, моделирующую все стадии данного процесса.

Придумайте алгоритмы кодирования и декодирования сигнала, уменьшающие общие затраты.

Проанализируйте задачу для различных соотношений входных параметров.

Для каждого информационного сигнала должны печататься передаваемый, полученный и восстановленный сигналы, а также сумма затрат.

Все, кроме одной, команды остановили свой выбор на интерпретаторе. Разработанные школьниками интерпретаторы выглядели вполне профессионально. В проектах было предусмотрено: меню, возможности экранного редактирования, запись и чтение с диска, трассировка программ.

В итоге *1-е место — у команды школы № 27, Чебоксары,*

2-е место — у команды школы № 18, Миасс,

3-е место — у команды школы № 14, Томск.

После турнира осталось много неясных вопросов: какого типа должны быть олимпиадные задачи по информатике, сколько и каких машин предоставлять командам и т. д. В связи с этим во время турнира родилась идея создания при АУИнформ олимпиадного комитета.

С. СЕМЕНОВ

Об олимпиадном комитете

Организован олимпиадный комитет Ассоциации учителей информатики, действующий на общественных началах. Его цели: пропаганда олимпиадного движения, создание банка задач, проведение турниров и олимпиад, работа со способными детьми.

Председателем комитета избран сотрудник ИПС АН СССР Я. Н. Зайдельман (г. Переславль-Залесский).

Комитет начинает сбор и систематизацию задач по информатике с целью их издания и использования в работе со школьниками.

От корреспондентов ждем задач с указанием авторов или источников их появления, мероприятий, где они использовались, решениями и пожеланиями по их дальнейшему применению. Авторство гарантируется, изыскиваются способы оплаты труда по составлению и комментированию задач.

Присылать задачи по адресу ассоциации: 660100, Красноярск, ул. Волочаевская, 44. Фонд задач АУИнформ.

В журнале «Информатика и образование» мы планируем публикацию и разбор наиболее интересных задач из фонда.

Совещание заведующих кабинетами информатики

В декабре 1990 г. в Москве в ЦИУУ РСФСР состоялось совещание заведующих региональными кабинетами информатики ИУУ РСФСР. По итогам совещания было принято решение, которое публикуется ниже.

На первом этапе информатизации образования (1985—1990 гг.) основной задачей сотрудников кабинетов информатики ИУУ была постановка курса информатики в школе. На базе кабинетов обеспечены подготовка и повышение квалификации учителей информатики, налажены сбор и тиражирование программного обеспечения в регионах. Обеспечена методическая работа с учителями информатики. За этот период коллективами кабинета создана материальная база, сформированы квалифицированные кадры методистов и учителей.

На втором этапе информатизации образования (1991—1995 гг.) на первый план выдвигаются задачи внедрения новых информационных технологий обучения в систему образования в целом. Соответственно на ИУУ ложится функция подготовки педагогических кадров в области информационных технологий обучения и управления, разработки программно-методических комплексов компьютерного обучения, создания информационно-справочных систем и т. д.

В этой связи целесообразно начать процесс создания на базе кабинетов информатики и ТСО общеинститутских информационно-технических центров (ИТЦ). Основными направлениями деятельности ИТЦ могут быть:

информатизация управленческой деятельности органов народного образования и ИУУ;

ввод в эксплуатацию, повышение эффективности использования, ремонт и обслуживание ТСО и компьютерных классов;

повышение квалификации работников народного образования в области использования компьютерных и аудиовизуальных средств обучения;

разработка, формирование, внедрение и информационное сопровождение программно-методических комплексов;

участие в решении общегосударственной проблемы переквалификации кадров в области новых информационных технологий.

Создание ИТЦ позволит осуществлять эффективное информационно-техническое обеспечение всех структурных подразделений институтов и вести разнообразную хозрасчетную деятельность.

Информационный центр по проблемам профтехобразования

В научно-исследовательском институте профессионально-технического образования АПН СССР в Ленинграде функционирует специализированный информационный центр по вопросам подготовки рабочих кадров.

Центр осуществляет сбор и обработку публикаций и неопубликованных материалов по вопросам профтехобразования. В их числе: книги, периодические издания (отечественные и зарубежные), отчеты о научно-исследовательских работах, авторефераты диссертаций, учебные планы и программы, переводы зарубежных публикаций, материалы педагогических чтений, конференций, семинаров, педагогические программные средства, учебные диафильмы и диапозитивы, рационализаторские предложения, справочники и другие материалы. В Центре депонируются рукописи.

С 1986 г. действует автоматизированная система обработки информации «Профтехобразование». Библиографическая база данных содер-

жит свыше 8 тыс. описаний вышеперечисленных документов, и ежегодно пополняется на 2 тыс. записей.

На основе базы данных в автоматизированном режиме осуществляется подготовка информационных изданий, ретроспективный поиск информации по разовым запросам, избирательное распространение информации по постоянно действующим запросам; аналитическая работа по изучению структуры документального информационного потока, терминологической базы материалов по профтехобразованию и некоторые другие услуги.

Центр выполняет заявки на копирование материалов, содержащихся в его фонде, и тех, информация о которых опубликована в информационных изданиях, а также заявки на книги и методические материалы, изданные институтом. Ежегодно в Центр обращаются свыше 1 тыс. учреждений и учебных заведений СССР и зарубежных стран.

Центр заключает договоры на информационное обслуживание, в том числе на поставку библиографической базы данных на магнитных носителях. Учреждения и учебные заведения, имеющие ПЭВМ ДВК-2, ДВК-2М, «Ямаха», IBM PC, могут приобрести полную копию базы или отдельных ее частей по государственным расценкам.

Подписка на текущие информационные издания (библиографические, реферативные и экспресс-информацию) осуществляется через «Союзпечать».

Наш адрес: 191119, Ленинград, ул. Черняховского, 2, ОНТИ НИИ ПТО АПН СССР. Телефон для справок: 112-24-66, 164-08-27, 112-24-14.

Межреспубликанская конференция в Тбилиси

В октябре 1990 г. в молодежном центре «Шавнабада», близ Тбилиси, состоялась Межреспубликанская конференция «Новые информационные технологии в образовании», на которой присутствовали представители Азербайджана, Белоруссии, Молдовы, России, Украины и Грузии.

В конференции, организованной УНПО «Информатика» Минобразования Грузии и Республиканским институтом переподготовки и повышения квалификации учителей, приняли участие ведущие специалисты страны по всем уровням образования, а также президент шведской фирмы «Дата Констракшн» Берт Лангстром и президент финской фирмы «Микронет» Иоуни Карвонен.

Направления работы конференции были определены в докладе Генерального директора УНПО «Информатика», лауреата Государственной премии Грузии Гии Гварамия: информационная технология как предмет и средство изучения в школах и профессионально-технических училищах, как средство повышения эффективности научно-технических исследований, повышения квалификации педагогических кадров, средство управленческой деятельности, применение ЭВТ в дошкольных учреждениях.

В рамках конференции была организована демонстрация готовых ПС по разным предметным дисциплинам и для различных типов машин.

Большой интерес у участников конференции вызвали выступления Ю. М. Горвица, представителя ассоциации «Компьютер и детство», об использовании компьютерного игрового комплекса в детских дошкольных учреждениях и докт. пед. наук, сотрудника НИИ психологии МНО УССР Л. В. Артемовой, занимающейся исследованиями в области использования ЭВТ в детских садах.

Применение ЭВТ в общеобразовательной школе было отражено во многих сообщениях, в частности — доцента ЛГПИ им. Герцена И. В. Готской и сотрудника УНПО «Информатика», лауреата Государственной премии Грузии И. И. Марквилашвили о разработке ППС для изучения физики и электротехники.

Многие участники семинара впервые познакомились с работой АСП «Кудиц» (акционерное социалистическое предприятие «Компьютерный учебно-демонстрационный и информационно-издательский сервисный центр»), сотрудничающего с европейским отделением фирмы *IBM*, и пришли в недоумение, почему Гособразование СССР, заключая договор о закупке техники для школ, не организовало закупку локальных сетей. Как можно организовать полноценный учебный процесс без эффективного использования ЭВТ? Создается впечатление, что путь проб и ошибок, пройденный народным образованием с 1985 г., забыт.

Вопросы преемственности между общеобразовательной и высшей школой были освещены профессором Тбилисского государственного мединститута Г. Ш. Васадзе.

Неоднократно обсуждались вопросы разработки ППС, методики их применения в учебно-воспитательном процессе профтехучилищ, организации работы отраслевого фонда алгоритмов и программ профтехобразования, созданного при НИИ профобразования молодежи АПН СССР.

О содержании и методике применения созданных пакетов обучающих программ по различным дисциплинам, изучаемым в ПТУ, рассказали сотрудник УНПО «Информатика» А. Е. Махалдиани и студент Грузинского технического университета (ГТУ) В. В. Гоцiридзе.

Интересен опыт создания пакета программ «Трудный подросток» (С. В. Кривцова, Н. Л. Волский, А. М. Семибратов — сотрудники кафедры новейших технологий ЦИУУ Минобразования РСФСР), позволяющего проводить психодиагностику подростков и давать конкретные рекомендации учащимся, педагогам и психологам по корректировке отрицательных и развитию положительных черт характера.

Об опыте использования ЭВТ в вопросах управления народным образованием сообщили работники лаборатории новых информационных технологий Минобразования Азербайджанской ССР. Сотрудники учебно-методического кабинета робототехники и ЭВТ ЦИУУ Минобразования РСФСР рассказали о применении ЭВТ при повышении квалификации.

Большой интерес вызвал доклад О. Давиташвили,

доцента ГТУ, по разработке некоторых приемов защиты программного продукта от несанкционированного копирования.

На закрытии конференции заместитель генерального директора УНПО «Информатика», лауреат Государственной премии Грузии З. И. Чхаидзе отметил, что свободная дискуссия позволила участникам увидеть реальное состояние дел, оценить рыночные возможности и перспективы внедрения НИТ на всех уровнях народного образования в Советском Союзе.

По оценке специалистов, присутствие на конференции представителей предприятий нового типа (СП «Ланит», СП «Свелен», АСП «Кудиц») позволило острее почувствовать все сложности, возникающие в связи с централизованным управлением. Эти предприятия ведут самостоятельную научную и прикладную деятельность, работая на конечный продукт и подключая к своей работе все новые организации и институты, что позволяет лучше понимать связанные с хозрасчетом механизмы взаимодействия.

Л. В. ЛИБИЗОВА, НИИ профтехобразования АПН СССР,
Л. К. ЕСИНА, ЦИУУ

Научно-практическая конференция в Минске

В ноябре 1990 г. в Минске проходила ежегодная Всесоюзная научно-практическая конференция «Электронно-вычислительная техника в учебно-воспитательном процессе профтехучилищ». В конференции принимали участие представители Гособразования СССР, инженерно-педагогические работники профтехучилищ, специалисты по вопросам разработки и внедрения электронно-вычислительной техники в народном образовании.

На конференции были рассмотрены проблемы совершенствования преподавания информатики в профтехучилищах, разработки педагогических программных средств, повышения квалификации преподавателей информатики, продемонстрированы программные средства.

В докладе А. Т. Глазунова, зам. директора ВНМЦентра, подчеркивалось, что в условиях компьютерного обучения чрезвычайно важно определять реальную педагогическую эффективность программных средств. Предложение докладчика выпускать научно-практический сборник по проблемам повышения педагогической эффективности было поддержано участниками конференции.

Оживленную дискуссию вызвало рассмотрение комплекта учебной документации по предмету «Информатика и автоматизация производства» для профессионально-технических училищ, разработанного Всесоюзным научно-методическим центром профессионально-технического обучения молодежи. Основной заслугой разработчиков комплекта является реализация блочно-модульного подхода к построению интегрированного курса информатики и автоматизации, позволяющего преподавателю, учитывая исходный уровень знаний и предъявляемый к подготовке учащихся (ознакомительный, пользовательский или профессиональный), самостоятельно конструировать содержание обучения, выбирать последовательность изучения тем.

На секции «Методика преподавания информатики, вычислительной техники и автоматизации производства» были продемонстрированы интересные педагогические программные средства для КУВТ «Ямаха», представленные ВНМЦентром, Московским филиалом института повышения квалификации руководящих работников и специалистов ПТО. Примечательно то, что существенно повысилось педагогическое качество разработок, лучше учитываются психологические особенности учащихся, программная продукция, принимая все более товарный вид, становится конкурентоспособной.

Участники секции «Использование ЭВТ в преподавании специальных, общетехнических, общеобразовательных дисциплин и в производственном обучении» посетили интересный открытый урок в профтехучилище № 40 Минска, где широко используется при изучении общетехнических и специальных предметов КУВТ «Ямаха».

М. Б. ЛЕБЕДЕВА

Советско-американский симпозиум

В сентябре 1990 г. в НИИ ССО АПН СССР проходил первый советско-американский симпозиум «Развитие технологии обучения в профессиональной школе», проведение которого стало возможным после подписания соглашения о создании совместной советско-американской лаборатории по проблемам совершенствования системы профессионального образования.

Соглашение было заключено в августе 1989 г. между НИИ ССО АПН СССР и американской Ассоциацией коммунальных и младших колледжей, которую представляли: Флоридский коммунальный колледж в г. Джексонвилле (Флорида), Региональный коммунальный колледж в г. Сент-Луисе (Миссури) и Окружной технический колледж в г. Уокеше (Висконсин). В 1990 г. к соглашению присоединился Центр профессионального образования при Колумбийском университете (штат Огайо). Соглашение предусматривает участие сторон в национальных конференциях и семинарах по проблемам профессионального образования, а также проведение ежегодных симпозиумов в СССР или США для обмена опытом, подведения итогов исследований, координации и планирования дальнейших работ. Симпозиум был посвящен актуальным проблемам;

разработке и внедрению в учебный процесс новых перспективных технологий обучения (в том числе и компьютерной), на базе проблемного, деятельностного подхода, реализующих цели раз-

вития у учащихся критического, творческого мышления;

путей и способов интеграции общего и профессионального образования;

разработке методики оценки качества профессионального образования;

социально-психологическим аспектам управления профессиональными учебными заведениями.

Как известно, педагоги обходятся двумя терминами: «дидактика» и «методика». Не претендуя на полноту анализа, можно заметить, что разрыв между этими понятиями и еще больший разрыв между ними и реальной педагогической практикой привели к тому, что большинство учителей учит «чему-нибудь и как-нибудь». В этом смысле «технология обучения» (и это звучало подтекстом как в докладах, так и в дискуссиях) должна представлять настолько целостную педагогическую систему, что «вырвать» из нее каким-либо образом дидактический, методический или же психологический компонент было бы просто невозможно. Такой подход к определению понятия «технология обучения» дает ключ к пониманию интереса, проявленного участниками к теоретическим докладам академика М. И. Махмутова и профессора Леона Лессинджера, эксперта Флоридского университета по вопросам образования. Проблемный подход, предложенный М. И. Махмутовым, и теория обучения различным стилям мышления (Де Боно, Л. Лессинджер), могли бы составить дидактическую основу новой технологии обучения.

В теории проблемно-развивающего обучения (М. И. Махмутов) существует своя классификация методов обучения. Выбор преподавателем того или иного метода не может не влиять на формирование определенного стиля мышления. Совместные исследования в этом направлении, возможно, дадут нам новые знания о механизмах мышления и способах его формирования.

Низкое качество программного обеспечения, разрабатываемого для компьютерной технологии обучения КТО (кстати, на него сетовали и американцы), кроме отсутствия достаточного количества «хороших» ЭВМ объясняется еще и отсутствием «хороших» идей, которые можно положить в основу разработки педагогических программных средств (ППС). Поэтому большой интерес вызвал доклад «Содержание, структура ППС и методы проблемного обучения» Д. М. Шакировой, Л. А. Струковой, Ф. Р. Суфияровой, И. Н. Нарыковой.

Доклады американских коллег показали, что им удалось продвинуться гораздо дальше нас. И дело не только и не столько в оснащенности учебных заведений техникой (Флоридский коммунальный колледж, например, насчитывающий 14 тыс. студентов, имеет 26 компьютерных лабораторий, в них — 350 компьютеров IBM, 70 — Apple и 60 терминалов). Дело в масштабах компьютеризации с точки зрения многообразия тех задач и проблем в колледжах, для решения которых привлекается техника. Именно под таким углом зрения рассматривают компьютеризацию наши американские коллеги, и, поскольку это стратегическое направление было выбрано ими давно, им было чем поделиться. Доклад руководителя отдела компьютерной технологии Флоридского коммунального колледжа Дугласа Керли называется: «Обзор состояния компьютерной технологии обучения в американских коммунальных колледжах». Автор выделил ряд основных направлений использования ЭВМ в колледжах и более подробно осветил использование ВТ для создания компьютерных курсов (CAI-курсов) по различным предметам. Практически к 1990 г. в американских колледжах не осталось учебных предметов, для которых не была бы создана поуровневая CAI-программа. К тому же американские специалисты с начала 80-х гг., когда был поставлен диагноз неудовлетворительного качества CAI-программ, произвели их тщательную селекцию. В каждом американском колледже сейчас существует и применяется программное обеспечение самых различных типов: тренажеры, компьютерные упражнения, учебные игры, программные системы по моделированию и имитации, программы, дающие консультацию, и т. д. Кроме программ, рассчитанных на обучающихся, используется много прикладного программного обеспечения, облегчающего труд преподавателя: различные графические программы, позволяющие мгновенно иллюстрировать лекционный материал графиками, диаграммами и пр., электронные справочники по учебной документации, базы данных по учебной и методической литературе, системы электронного контроля и т. д. Таким образом, вычислительная техника в американских колледжах имеет полифункциональную нагрузку и используется не только в процессе обучения, но и для организации и управления этим процессом. Именно в этом особенно заметны успехи американской системы образования. К этому необходимо стремиться и нам. Отказавшись, возможно, от единого унифицированного курса информатики в том виде, в каком он существует сегодня, и высвободив тем самым наши ограниченные человеческие и машинные ресурсы, мы можем с их помощью решить задачу внедрения информационной технологии в систему образования.

На симпозиуме были представлены доклады, описывающие последние достижения американской системы образования в освоении новых информационных технологий. Это — внедрение в практику аудиовизуальных систем на основе ПЭВМ (Д. Керли) и систем телекоммуникаций (Л. Паттон). Автор последнего доклада считает, что, «возможно, телекоммуникационная технология станет единственным приемлемым методом обучения в будущем», так как это связано с возрастанием потребностей общества в быстром и надежном обмене информацией». С системами телекоммуникационного обучения американские ученые-педагоги связывают возможность дистанционного обучения, т. е. получения образования в удобном для обучающегося месте и в удобное для него время.

Организация различного рода совместных предприятий и проектов практикуется сейчас довольно широко, и, наверное, в условиях поиска нашим обществом путей выхода из кризиса заимствование иностранных достижений («интеллектуальных» и «материальных») оправдано. Многие видят в этом лишь дань моде. Однако не следует изобретать собственный «велосипед» в самых различных областях, в том числе и в педагогике. Здравый смысл подсказывает, что истинный путь, ведущий к прогрессу, это взаимодействие, взаимообмен и взаимообогащение.

И. НАРЫКОВА,
Л. СТРУКОВА

Семинар в Бердянске

В сентябре 1990 г. в Бердянске проходила школа-семинар «Компьютеризация образования: психология, эргономика и гигиена», организованный лабораторией психолого-эргономических исследований Киевского научно-производственного объединения «Горсистемотехника», Бердянским филиалом Института проблем информатики АН СССР, Республиканским Домом экономической и научно-технической пропаганды Общества «Знание» УССР.

Сохранение здоровья пользователей вычислительной техники — важнейшая проблема успешной компьютеризации общества и овладения компьютерной грамотностью. Поэтому тема семинара вызвала широкий интерес специалистов из разных городов страны — Москвы, Ленинграда, Новосибирска, Херсона, Запорожья и др.

В семинаре приняли участие научные сотрудники, педагоги, гигиенисты, медики и психологи, разработчики программных средств, программисты, преподаватели школ, профессионально-технических учреждений, детских садов, работники санэпидемстанций.

На семинаре рассматривался широкий круг проблем компьютеризации образования и связанных с ним междисциплинарных вопросов гигиенических, психологических, эргономических и медицинских, в частности, вопросов сохранения здоровья учащихся — пользователей ПЭВМ. Участники семинара ознакомились с программными разработками в области компьютеризации дошкольного, среднего и высшего образования.

С докладом по актуальным проблемам компьютеризации дошкольного образования выступила зав. лабораторией НИИ дошкольного воспитания АПН СССР канд. психол. наук С. Л. Новоселова. В докладе изложены основные положения психолого-педагогической концепции компьютеризации дошкольного образования. Особое внимание уделено психолого-педагогическим проблемам разработки совместно с коллективом специалистов из НПО «Микроэлектроника» и внедрения компьютерно-игровых комплексов (КИК) для дошкольников.

Проблемам и перспективам информатизации профессионально-технического образования и задачам недавно организованного отраслевого центра ПТО посвящен доклад зав. отделом ВНИИ ПТО канд. тех. наук В. И. Сопина (Ленинград).

Основным задачам и перспективам развития Всесоюзного Центра новых информационных технологий, созданного в Москве на базе НИИ ВШ, вопросам координации работ по компьютеризации образования и, в частности, в области психологических исследований был посвящен доклад М. А. Сухинской.

Психологические проблемы, связанные с внедрением ПЭВМ в сферу образования, рассмотрены в докладах специалистов факультета психологии из МГУ. Канд. психол. наук Л. П. Гурьева остановилась на важнейших психологических последствиях компьютеризации, а также уделила внимание вопросам диагностики одаренности и ее компьютеризации.

Большой интерес вызвал доклад докт. пед. наук Ю. А. Первина об особенностях включения информатики в систему обучения в начальной школе и программно-методической разработке «Роботландия» (Переславль-Залесский, Институт программных систем АН СССР).

Проблемы влияния ПЭВМ на организм ребенка были подняты в докладах специалистов по физиологии и гигиене детей. Сотрудник Института гигиены детей и подростков Минздрава СССР А. Ф. Богданов рассказал о новейших исследованиях функциональных состояний детей при работе с различными ПЭВМ. Отмечалось, что наибольшая утомляемость выявлена при работе с ПЭВМ «Агат». Слушатели школы-семинара ознакомились с приемами снятия напряжения у пользователей и режимами их работы.

Выступления научного руководителя семинара канд. психол. наук Н. И. Повякель были посвящены вопросам эргономики рабочих мест и сохранения здоровья пользователей, проектирования и экспертизы программных средств учебного назначения и психологическим проблемам, а также важности и необходимости совместных усилий специалистов по психологии, эргономике, педагогике, медицине, разработчиков ТС и ПС при решении проблем компьютеризации образования.

Вызвал интерес слушателей и доклад канд. психол. наук С. Н. Козловского из НИИ общей и педагогической психологии АПН СССР, посвященный психологической оценке изображения на экране ПЭВМ и значении ее в экспертизе ПЭВМ.

Доклад канд. техн. наук Р. Н. Бабкиной из Бердянского филиала ИПИ АН СССР был связан с эргономическими вопросами разработки интеллектуального интерфейса «человек-ПЭВМ». Доклад сопровождался демонстрацией программных продуктов.

Слушатели и участники школы-семинара высказали пожелание регулярно проводить семинары по междисциплинарным проблемам компьютеризации образования.

Всесоюзный институт повышения квалификации руководящих работников и специалистов профтехобразования

ПРЕДЛАГАЕТ преподавателям и руководящим работникам учебных заведений пройти переподготовку и повышение квалификации по следующим направлениям:

методика преподавания предмета «Основы информатики и вычислительной техники»;

технология разработки и методика применения педагогических программных средств вычислительной техники.

Стоимость обучения от 400 до 600 руб. в зависимости от продолжительности. Слушатели обеспечиваются научно-методической литературой.

ПРОВОДИТ разработку комплексного методического обеспечения по предмету «Основы информатики, вычислительной техники и автоматизации».

Программно-методический комплекс по предмету включает:

учебную программу по предмету «Основы информатики, вычислительной техники и автоматизации»;

сборник опорных конспектов по предмету;

методическое пособие для преподавателя по преподаванию курса;

методическое пособие по проведению лабораторно-практических работ;

методические рекомендации по организации внеурочной работы;

сборник программ для использования на занятиях по предмету «Основы информатики, вычислительной техники и автоматизации»;

сборник задач, заданий и упражнений по курсу;

рабочую тетрадь по предмету;

комплект плакатов;

комплект транспарантов для графопроектора;

комплект педагогических программных средств вычислительной техники.

Программное обеспечение поставляется для следующих видов техники: «Ямаха», УКНЦ, «Корвет», ДВК, КУВТ-86 (Бейсик и Фокал).

Ориентировочная стоимость комплекса от 650 до 900 руб. в зависимости от комплектации программным обеспечением. Возможно изменение стоимости при переходе на рыночные цены. Заявки на приобретение комплекса и аванс (250 руб.) направлять до 1 июня 1991 г.

Поставка программно-методического комплекса в июне-августе 1991 г.

Наш адрес: *192007, Ленинград, Воронежская ул., 79.*

Расчетный счет 30000608176 во Фрунзенском отделении Промстройбанка Ленинграда.

Справки по телефону 166-58-74.

Цена 1 р. 20 к.
70423

Куб 10-11

ИНФО

2 '91

ИНФОРМАТИКА И ОБРАЗОВАНИЕ

ISSN 0234—0453. Информатика и образование, 1991, № 2, 1—128