ISSN 0234—0453

ИНФО 2 1989

ИНФОРМАТИКА И ОБРАЗОВАНИЕ

Сотрудничество:
оно и в совместных исследованиях
по компьютеризации образования,
и в организации международных выставок,
и во встречах советских и американских школьников

«Телекоммуникации и компьютерные среды»
Репортаж номера
Молодежная инициатива

Плакат фирмы «Диалог», совместного
советско-американского предприятия

Уважаемые читатели!
Редакция обратилась к трем авторским коллективам,
написавшим пробные учебники по информатике,
с предложением подготовить для нашего журнала
цикл статей под условным названием
«Методические рекомендации для учителей».
Мы надеемся, что наши «опережающие» публикации
не только позволят получить представление
о предлагаемых авторами
методологических подходах к пониманию
содержания предмета, его места
в системе среднего образования,
связи с другими учебными дисциплинами,
но и помогут тем, кто в будущем учебном году
начнет работать с новыми учебниками.
Ждем ваших откликов с оценкой,
насколько интересны и полезны
эти материалы.

Научно-методический журнал
Государственного комитета СССР
по народному образованию

Издается с августа 1986 г.
Выходит шесть раз в год
Москва «Педагогика» 1989

ИНФОРМАТИКА И ОБРАЗОВАНИЕ

2 '89
март — апрель

Содержание

Точка зрения

Репортаж номера

Нам пишут

Информация

Обложка *Э. Бажилина*

В оформлении номера принимали участие *Э. Бажилин, Н. Горбунова, А. Пономарев, С. Расторгуев*

Редактор отдела *А. Кравцова*
Научный редактор *Т. Драгныш*
Зав. редакцией *Н. Игнатова*
Художественный редактор *Л. Розанова*
Корректор *О. Пурлова*

Сдано в набор 19.01.89. Подписано в печать 09.03.89. А 03545.
Формат 70×100/16. Бумага офсетная № 1. Печать офсетная. Усл. печ. л. 10,40
Усл. кр.-отт. 42,88. Уч.-изд. л. 13,30. Тираж 83130 экз. Заказ 171. Цена 60 коп.

Издательство «Педагогика» Академии педагогических наук СССР и Государственного комитета СССР по делам издательств, полиграфии и книжной торговли

Адрес для переписки: 107005, Москва, Лефортовский пер., 8
Адрес редакции: Студенческая ул., 37.
Телефон редакции: 249-97-77

Ордена Трудового Красного Знамени Чеховский полиграфический комбинат ВО «Союзполиграфпром» Государственного комитета СССР по делам издательств, полиграфии и книжной торговли
142300, г. Чехов Московской обл.

А. БЕЛЯЕВА, М. КОШЕЛЮК, Г. СОЛДАТОВА

Телекоммуникации и компьютерные среды

Надежды на повышение качества образования путем насыщения школ компьютерной техникой в развитых западных странах в последнее время сменились разочарованием. Это стимулирует поиск новых подходов к использованию компьютеров с опорой на перспективные модели организации учебной деятельности. Для их разработки полезна координация усилий не только представителей различных научных дисциплин, но и сотрудничество ученых разных стран, опирающихся на национальный опыт использования компьютерных технологий в школьном образовании.

Примером такого сотрудничества является реализуемая с 1985 г. советско-американская исследовательская программа по разработке оптимальных способов применения компьютерных (в том числе коммуникационных) технологий в обучении младших школьников. С советской стороны программу возглавляет академик Е. Велихов, с американской — доктор Д. Гамбург (проект «Велихов—Гамбург»). Ядро исследовательской группы проекта составляют психологи Лаборатории сравнительных исследований человеческого познания Калифорнийского университета, г. Сан-Диего, возглавляемые профессором М. Коулом (координатор проекта с американской стороны), и исследователи группы психологических проблем общения в условиях современных коммуникационных технологий Института психологии АН СССР. Руководитель этой группы кандидат филологических наук А. Беляева — координатор проекта с советской стороны. В работе также принимают участие специалисты в области образования, программного обеспечения, коммуникационных технологий из ряда других научно-исследовательских институтов СССР и исследовательских центров США. Полноправные участники проекта — учителя и дети из школ Москвы, Переславля-Залесского, Сан-Диего и Нью-Йорка.

Благодаря совместным усилиям советских и американских исследователей проект непрерывно развивается и в настоящее время получил статус проекта Международного фонда за выживание и развитие человечества.

Основной целью проекта является разработка оптимальной модели международного научного сотрудничества с использованием современных коммуникационных технологий. Конкретный объект совместного исследования — проблема использования компьютера как средства познавательного и социального развития детей в начальных классах школы. Особенностью проекта является использование новых коммуникационных технологий как основного способа обеспечения организации и осуществления научно-исследовательской работы на всех ее этапах. Ежедневная телекоммуникационная связь обес-

печивает совместное планирование и подготовку экспериментов, согласование стратегий сбора и способов обработки эмпирических данных, оперативное обсуждение теоретических проблем и практических результатов и др.

Технически телекоммуникация осуществляется следующим образом. Отправитель вводит текст сообщения через свой терминал (компьютер). С помощью специальной коммуникационной программы и соответствующего устройства — модема (модулятора — демодулятора) информация, закодированная в компьютере, преобразуется в сигналы телефонной линии и по каналам телефонной связи (кабельным и спутниковым) передается к терминалу получателя, где с помощью модема и коммуникационной программы происходит обратное преобразование и декодирование. При этом, как правило, в качестве промежуточного звена используется головной компьютер, обладающий высоким быстродействием, большой памятью и оснащенный специальной программой, обеспечивающей хранение, обработку и пересылку поступающих пользователям системы сообщений, а также ряд дополнительных возможностей (доступ к базам данных, игровым программам и т. д.). Это обеспечивает возможность разделения во времени моментов отправки и получения сообщений (за счет хранения их в памяти головного компьютера), сохранения протоколов опосредованного общения как побочного продукта самого общения, гибкости в выборе конкретных форм организации контура «отправитель(и) — адресат(ы)» (сообщение может быть отправлено определенному адресату, группе адресатов, любому пользователю системы). Кроме того, имеется возможность живого синхронного диалога и полилога — группового общения одновременно работающих людей.

Конкретной формой организации взаимодействия отправителей и адресатов в телекоммуникации выступает телеконференция, которая представляет собой длительное (от нескольких недель до нескольких месяцев и больше) коммуникативное событие, протекающее в форме группового общения (полилога) и подчиненное совместному решению тех или иных задач (обсуждению тех или иных проблем).

В рамках данного проекта телеконференции выступают в качестве основной формы опосредованного общения. При этом используются как длительные телеконференции, посвященные текущим вопросам организации работы по проекту, так и краткосрочные, организуемые на отдельных его этапах. Телеконференции также различаются тематически. Их число постоянно пополняется за счет выделения в ходе обсуждения самостоятельных тем (организация экспериментальной работы по отдельным обучающим программам, теоретические и технические вопросы организации телекоммуникации и ее использования в обучении и др.).

Участниками телеконференций являются исследователи, учителя, дети. Наряду с этим широко используются другие формы организации опосредованного общения, в частности персональный обмен сообщениями между отдельными участниками проекта в режиме реального и отсроченного времени. Гибкое использование различных форм организации опосредованного общения, анализ их эффективности в зависимости от конкретных условий и решаемых задач позволяет строить многоуровневую и многоаспектную модель телекоммуникации.

Самостоятельной целью проекта «Велихов—Гамбург» является исследование новых возможностей организации совместной учебной деятельности в условиях научно обоснованного использования современных коммуникационных компьютерных технологий в образовании. Эта работа осуществляется на основе методологических и теоретических положений советской психологии об определяющем влиянии деятельности и общения на психическое развитие ребенка.

Как известно, в работах советских психологов (работы Л. Выготского, А. Леонтьева, Д. Эльконина; теория поэтапного формирования умственных действий П. Гальперина; теория учебной деятельности В. Давыдова и др.) было установлено, что на основе внеш-

них предметно-практических действий путем их последовательных изменений и сокращений формируются внутренние, идеальные действия, совершаемые в умственном плане. Этот процесс перехода от внешнего (интерпсихического) плана к внутреннему (интрапсихическому) в психологии обозначается термином «интериоризация».

Однако познавательное развитие ребенка обусловлено не столько его индивидуальной деятельностью, сколько общением и совместной деятельностью с другими детьми и взрослыми. Данные многочисленных исследований показывают, что включение обучаемых в совместную деятельность способствует обогащению и ускорению развития индивидуальной познавательной активности. Иными словами, интерпсихическое задано первоначально именно в форме общения с другими людьми. В процессе интериоризации происходит переход от управления психической деятельностью ребенка со стороны взрослого к ее саморегуляции, т. е. осуществляется детерминация сознательной деятельности развернутыми формами социального общения (Л. Выготский, А. Лурия). Согласно Л. Выготскому, взаимодействие ребенка со взрослыми и другими детьми опосредовано использованием исторически выработанных культурных орудий-средств, на основе которых формируются внутренние средства психической деятельности.

Согласно такому подходу к учебной деятельности, когда общение рассматривается как средство формирования и форма проявления познавательных психических процессов, в рамках проекта разрабатывается экстремальная парадигма, опорными компонентами которой являются ряд когнитивно-коммуникативных ситуаций, организуемых в обучающей компьютерной среде с включением элементов телекоммуникации. При этом компьютер в обучении рассматривается нами как продуктивное средство организации совместной познавательной деятельности и общения детей. Его эффективное использование предполагает не вытеснение учителя и традиционных средств развития познавательной деятельности из учебного процесса, а, наоборот, обогащение и дополнение обучающей среды за счет тех новых возможностей, которые предоставляет компьютер. В процессе совместной учебной деятельности, осуществляемой в обучающей компьютерной среде, ребенок решает ряд не только когнитивных, но и коммуникативных задач. Применение современных коммуникационных технологий, связанных с использованием компьютерных сетей (типа «электронной почты», систем телеконференций и т. п.), позволяет расширить границы того социального окружения, в рамках которого осуществляется познавательное развитие ребенка. Иными словами, современные коммуникационные технологии выступают в качестве нового дополнительного средства развертывания и фиксации интерпсихического плана. В целом такой подход мыслится как реализация функциональной обучающей среды, в которой работа ребенка с компьютером не является замкнутой системой, а встроена в более широкое взаимодействие ребенка с миром.

Зависимость особенностей психического развития детей от способа включения взрослых в процесс обучения, от характера организуемых с их помощью форм взаимодействия детей друг с другом определяет необходимость поиска конкретных путей включения взрослых в обучающую компьютерную коммуникационную среду. При этом возникает ряд серьезных теоретических и методических проблем. Несмотря на то что использование телекоммуникации в образовании приобретает в мире за последние годы все большее распространение, вопросы специфики такого обучения, его отношения к традиционным формам обучения, создание оптимальных методов его организации, разработка специального коммуникационного программного обеспечения для детей еще далеки от окончательного решения.

В рамках проекта «Велихов—Гамбург» в качестве одной из конкретных форм включения взрослых в обучающую компьютерную среду исследуется возможная роль так называемого

Электронного Волшебника, существующего для детей только в опосредованном общении. Его функции могут выполнять психологи-исследователи, преподаватели, старшеклассники. Для детей Электронный Волшебник — это особый участник совместной деятельности, к которому дети могут обращаться лишь посредством телекоммуникации. Волшебник вносит игровой компонент во взаимодействие детей с компьютером, вводит новые учебные задачи и оказывает помощь в их решении, направляет процесс опосредованного общения детей друг с другом. Взаимодействие с Электронным Волшебником способствует распределению функций между детьми в процессе совместной деятельности, организации как соревнования, так и сотрудничества в группе. Необходимость обращаться посредством телекоммуникации за помощью к лицу, наделенному особыми функциями, но отсутствующему в непосредственном окружении, способствует более активному использованию письменной речи для общения и соответственно стимулирует усовершенствование навыков владения клавиатурой у детей. Обмен с Электронным Волшебником письменными сообщениями по конкретным «рабочим» ситуациям способствует развитию процессов вербализации, более глубокому осознанию и обобщению выявляемых детьми закономерностей. Таким образом, в нашей экспериментальной модели Электронный Волшебник оказывается значимым центром коммуникативной активности детей и взрослых, универсальным механизмом, переопосредствующим их взаимодействие в процессе совместной учебной деятельности.

Организация опосредованного общения детей на разных этапах существования проекта различалась по значимости и характеру поставленных задач, по количеству участников — взрослых и детей, по продолжительности и объему телекоммуникационных контактов. При постоянной ежедневной коммуникации исследователей дети участвовали периодически как в отдельных, краткосрочных экспериментах — пилотажных исследованиях для подготовки последующих циклов работы или контрольных экспериментальных сериях, осуществляемых с целью перепроверки и уточнения получаемых результатов, так и в относительно длительных (до нескольких недель) экспериментальных циклах. Во время такой работы группы детей из обеих стран включались в совместную учебную деятельность, в процессе которой использовались конкретные обучающие программные средства и осуществлялся телекоммуникационный обмен. При этом осуществлялась телекоммуникация детей не только с их зарубежными партнерами, но и с другими детьми внутри страны.

Впервые такой длительный экспериментальный цикл (в течение месяца) был реализован на базе Международного компьютерного лагеря в августе 1986 г. в г. Переславле-Залесском. Исследование осуществлялось сотрудниками группы психологических проблем общения в условиях использования современных коммуникационных технологий Института психологии АН СССР совместно с сотрудниками Института программных систем АН СССР. В экспериментальном исследовании принимали участие 12 детей в возрасте 9—10 лет. Параллельно аналогичная работа проводилась в Кейлер Скул (Сан-Диего) с участием психологов из Калифорнийского университета и из других научных центров США.

В настоящее время на основе осуществленных методических и экспериментальных разработок проводится уже постоянная еженедельная работа детей в детском компьютерном клубе проекта «Велихов—Гамбург» на базе Центра компьютерного обучения ИАЭ им. И. В. Курчатова и в компьютерном кружке школы № 45. Экспериментальная работа с младшими школьниками ведется параллельно и совместно с аналогичной работой в детских компьютерных клубах в г. Сан-Диего и Нью-Йорк. В ближайшее время в экспериментальную учебную деятельность и телекоммуникационный обмен учащихся младших классов по программе проекта включается детский компьютерный клуб «Байтик» (г. Троицк).

При осуществлении телекоммуника-

ционного обмена между детьми на всех его этапах активную помощь детям оказывают взрослые. Наряду с выполнением своих основных исследовательских задач они участвуют в обсуждении содержания получаемых и посылаемых детьми сообщений, подготовке детьми файлов в текстовом редакторе, отсылке и приеме сообщений, их распечатке на принтере, переводе в случае необходимости с русского языка на английский и наоборот. В системе телеконференций проекта для обмена сообщениями между детьми специально выделена отдельная подконференция. В ее рамках дети обсуждают конкретные вопросы и проблемы, возникающие в ходе их учебной работы, здесь разворачивается процесс вербализации детьми гипотез и стратегий решения задач. Кроме того, часто возникает инициируемый самими детьми, взрослыми или Волшебником обмен вопросами, загадками и занимательными задачами, стихотворениями собственного сочинения, разными впечатлениями. Параллельно дети осваивают начальные навыки работы с программами разного класса (текстовый редактор, коммуникационные программы и др.).

Реализуемый в рамках проекта теоретический и экспериментальный подход к исследованию возможностей использования телекоммуникации для организации совместной учебной деятельности и общения учащихся позволяет по-новому взглянуть и на используемые в процессе обучения программные средства. Апробация конкретных обучающих программ, разработка методических рекомендаций по организации учебной деятельности на их основе, а также доработка этих программ с учетом опыта экспериментальной работы выступают в качестве следующей важной цели проекта «Велихов—Гамбург».

В процессе экспериментальной работы нами используются перспективные образцы обучающих программ, направленных на развитие у детей навыков решения задач проблемного характера и получивших высокую оценку специалистов и учителей в США. По мнению многих исследователей, логика решения задач этого типа предполагает не индивидуальные формы обучения, а активное общение, сотрудничество учеников с учителем, взаимодействие между учениками. При этом учащиеся получают возможность научиться выделению некоторого качества как общего, что предполагает его отчленение от других качеств и рассмотрение его как самостоятельного и особого предмета последующих действий, а также научиться фиксировать это общее в знаковой форме. Таким образом, при решении задач проблемного характера основной целью обучения является не столько формирование и отработка (тренировка) частных навыков, сколько освоение в процессе взаимодействия с другими обобщенных стратегий решения широкого класса задач.

Примерами программ такого типа могут служить обучающие программы «Пруд» (Pond), «Фабрика» (Factory), «Выживание на острове» (Island Survivors) производства компании «Санбёрст Комюникейшн» (США), которая в настоящее время является полноправным участником проекта «Велихов—Гамбург». Эти программы предназначены для работы на компьютерах типа «Apple» и IBM. Они могут также использоваться на компьютерах «Правец» производства НРБ, имеющихся в ряде советских школ.

При работе с программой «Пруд» от ребенка требуется правильно провести лягушку через лабиринт, составленный из зигзагообразно расположенных листьев водяной лилии, на другой берег пруда. Для решения этой задачи, имеющей разные уровни сложности, учащийся должен вычленить в результате исследования лабиринта повторяющийся в нем конечное число раз фрагмент и составить простую алгоритм-программу (например, →3/↓4), выполняя которую лягушка проходит весь лабиринт. Программное средство «Фабрика» имитирует действие автоматизированного конвейера, который может включать станки, осуществляющие операции трех типов — пробивание отверстий разных форм, проведение линий разной толщины и поворот за-

готовки. От учащегося требуется на основе анализа заданного образца выбрать необходимые для его изготовления станки и установить режимы и последовательность их работы. Учащийся также может планировать и изготавливать собственную продукцию. При работе с этими программными средствами учащиеся решают близкие по содержанию задачи: осуществление перцептивного анализа, вычленение необходимого набора операций и установление отношений между ними, построение гипотез и их проверка, составление алгоритмов. Несколько иной тип обучающего программного средства представляет программа «Выживание на острове», предлагающая решение задач экологического характера с использованием моделирующих систем. Сюжет программы состоит в следующем: трое потерпевших кораблекрушение людей попадают на необитаемый остров и должны прожить на нем в течение года до прихода спасательного судна. Их цель состоит в том, чтобы спланировать и организовать свою деятельность на каждый месяц таким образом, чтобы, с одной стороны, постоянно поддерживать необходимый уровень пищи, топлива, к сроку построить жилье, а с другой стороны, не нарушить в результате своей деятельности — охоты, рыбной ловли и сбора трав и ягод — экологическое равновесие в наземной и водной средах острова. При работе с данной программой от учащихся требуется совместно планировать свою деятельность, принимать групповые решения и распределять между собой функции. При этом обучающая игра позволяет ввести начальные понятия об экологии (такие, как экологическое равновесие, экологическая цепочка, уровень популяции и т. д.).

Опыт работы с данными образцами обучающих программных средств показал перспективность их применения на начальных этапах работы с персональным компьютером в младших классах общеобразовательной школы. Их использование действительно способствует формированию у детей стратегий решения проблемных задач, навыков перцептивного анализа, активизирует процесс построения гипотез и прививает культуру их систематической проверки, развивает навыки выработки совместного решения.

Вместе с тем обнаружился ряд недостатков самих программных средств, затрудняющих их эффективное использование. Так, предельная простота управления в программах «Пруд» и «Фабрика» оборачивается недостаточной гибкостью и адаптивностью данных программных средств к индивидуальным особенностям процесса усвоения необходимых познавательных навыков. Логика нарастания уровней сложности в обеих программах задается механически (числом элементов в фрагменте лабиринта пруда, количеством станков в конвейере), без учета психологических особенностей познавательного развития ребенка. Недостаточно продумано смысловое содержание задач, что отрицательно сказывается на мотивации детей и затрудняет перенос формируемых навыков. Существенным недостатком данных программ, основное содержание которых сводится к постановке задач, требующих выработки соответствующих стратегий, является отсутствие в них таких необходимых компонентов, которые могли бы выступить реальной основой для формирования познавательных стратегий. В частности, такими компонентами должны быть следующие: возможность фиксации промежуточных результатов в «Фабрике», визуализация выделенных фрагментов в программе «Пруд» и др. Их отсутствие затрудняет использование программ «Пруд» и «Фабрика» в качестве моделей для усвоения выделенных качеств как действительно общих отношений.

При работе с программой «Выживание на острове» выявлена необходимость дополнить методические руководства рекомендациями по организации ряда когнитивно-коммуникативных ситуаций, позволяющих глубже отработать способы обмена возникающими в процессе игры экологическими ситуациями на «острове» посредством телекоммуникации.

Результаты проведенного анализа данных программных средств, накопленный опыт их использования в обуче-

нии позволили выработать методические рекомендации по организации на их основе совместной учебной деятельности, а также выдвинуть ряд предложений по их усовершенствованию. На этой основе в настоящее время участниками проекта совместно с представителями фирмы «Санбёрст Комюникейшн» ведется работа по подготовке новых методических руководств к программам, усовершенствованию самих программных средств и разработке новых образцов (в том числе и версий на русском языке) с учетом возможности их использования в условиях локальной компьютерной сети и телекоммуникации.

В заключение хотелось бы отметить плодотворность сотрудничества ученых СССР и США. В рамках проекта «Велихов—Гамбург» накоплен опыт использования средств телекоммуникации для организации международного научного взаимодействия. Завершенные к настоящему времени циклы экспериментальных исследований показали, что использование возможностей современных коммуникационных компьютерных технологий позволяет существенно расширить границы социальной микросреды, в рамках которой осуществляется психическое развитие ребенка, повысить уровень мотивации детей при работе с компьютером, уровень осознания и обобщения благодаря формированию новых типов вербализации. Включение элементов телекоммуникации, разработка методических рекомендаций по организации учебной деятельности и усовершенствованию используемых программных средств позволяет существенно продвинуться в построении научно обоснованных современных эффективных обучающих компьютерных средств. Работа в этих направлениях продолжается.

ЧТО МОЖЕТ ЭВМ?

ЭВМ — не игрушка!

США и Канада, Мексика и Аргентина, Гонконг и Израиль... 4000 школьников четвертых-шестых классов из этих стран весной 1988 г. проводили совместные экологические наблюдения. Это не значит, что они собрались в гигантском международном лагере. Школьники у себя дома собирали дождевую воду, измеряли ее кислотность и по компьютерной информационной сети передавали результаты в центральный банк данных.

Туда же можно было «заслать» вопросы к ученым, следившим за ходом работ и комментировавшим получаемые результаты. Через компьютерную сеть можно было «считать» ответы, узнать о ходе работ участников эксперимента из других стран.

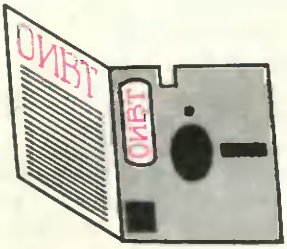
Ценность подобного эксперимента очевидна, но все же приведем высказывание одного из участвовавших в нем педагогов — Ричарда О'Грэди: «Дело не только в том, что ребята занимаются прикладной наукой. Важно, что они поняли: наука — дело коллективное, а природа — для всех одна... Ценным достоинством этой программы является то, что можно проанализировать данные, полученные кем-то еще, и сравнить их со своими. Дети вели себя, как настоящие ученые. Они делали открытия».

Конечно, подобное использование мировых информационных сетей — дело не дешевое, и выполнение программы стало возможно только благодаря крупным пожертвованиям — по 2,3 млн. долларов выделили Национальный фонд науки США и Национальное географическое общество США, а фирма «Эппл компьютер» предоставила 175 ПЭВМ.

Твердая рука

Тот, кто видел, как работает настоящий художник, наверное, ничуть не удивится, что точность нанесения штрихов на холст даже у самого талантливого мастера кисти не превышает долей миллиметра. Да и скорость рисования далека от «крейсерской скорости» пишущего узла 14-цветного плоттера фирмы «Хьюстон инструмент». Почти метровую линию может провести за одну секунду этот неутомимый прибор. Даже ребенок, рисующий первую в своей жизни «каляку-маляку» на новых обоях, не угонится за графопостроителем. Плоттер позволяет достигать разрешения до 0,0127 мм и повторяемости $\pm$0,05 мм. Ни один, даже очень хорошо заточенный карандаш не позволит нарисовать две линии на таком расстоянии друг от друга так, чтобы они не слились в одну. А пишущие узлы плоттера еще и выдерживают ускорение до 4g, которое возникает при их разгоне и торможении.

Итак, высокая скорость, недоступная человеку разрешающая способность, богатая цветовая гамма — неужели из-под перьев плоттера фирмы начнут выходить бессмертные творения формата от А∅ до А4? Увы (или к счастью?), нет. Прежде чем машина сотворит уникальное произведение, чудо должен сотворить программист. И хотя фирма предлагает для этого удобный язык, на котором можно описать графический шедевр во всех цветах и деталях, необходимо, чтобы это описание было составлено мастером программирования.



А. ГЕЙН, Е. ЛИНЕЦКИЙ, М. САПИР, В. ШОЛОХОВИЧ

Информатика: как решать задачи с помощью ЭВМ

Памяти
Владимира Габриэлевича Житомирского

В настоящее время преподавание информатики в школе осуществляется по нескольким учебным пособиям [1—4]. Каждое из этих пособий построено на своей концепции курса ОИВТ, имеет свою дидактическую структуру, содержит свои методические приемы обучения. Большинство из них хорошо известны читателям благодаря соответствующей методической литературе и публикациям в журнале «Информатика и образование». В частности, в статье [5] большое внимание уделено особенностям нашего подхода к изучению курса ОИВТ, **принципиальной установкой которого является обучение основным принципам решения задач на ЭВМ.**

Этой статьей мы открываем серию публикаций, о методике преподавания курса ОИВТ с использованием учебного пособия [4, в которых содержатся подробные методические рекомендации для учителей информатики по проведению уроков. Уделено внимание вычленению и обоснованию основных идей курса ОИВТ, умений и навыков, которые должны быть выработаны у учащихся, а также соответствующих дидактических и методических средств. Порядок рассмотрения тем курса ОИВТ в основном соответствует пособию [4]; ниже мы постараемся обосновать именно такой порядок изложения материала.

Мы обсудим общие методические проблемы, представляющие, по нашему убеждению, интерес для широкого круга читателей. В частности, будет разобрано довольно большое количество задач, которых, как известно, всегда не хватает учителю. А ведь при подготовке учителя к уроку чрезвычайно важно методически правильно подобрать задачи — не ошибиться в уровне их сложности, позаботиться о полноте системы навыков, формируемых при решении этих задач. Чтобы помочь учителю, мы классифицируем задачи и по сложности, и по дидактической насыщенности.

ВВЕДЕНИЕ

В каждой статье данного цикла (посвященной одной или нескольким темам курса ОИВТ) можно выделить три составные части. Это

1) методическая структура темы (вычленение ядра — основных понятий, определений и правил, умений и навыков, которые учащиеся должны приобрести в первую очередь, а также дополнительного материала);

2) подробный разбор решений задач, включающий, как правило, возможный процесс формирования решения в ходе диалога «учитель — ученик» (в ряде случаев приводится текст такого диалога);

3) классификация задач по сложности, типу и дидактической насыщенности, общие методические указания по их решению.

Охарактеризуем вкратце каждую из этих частей. Рассказывая о методике изложения нового материала, мы уделя-

ем особое внимание созданию как мотивации к изучению курса ОИВТ в целом, так и мотивации к каждой конкретной теме в отдельности. Общая мотивация создается последовательным проведением линии на применение ЭВМ к решению «жизненных» задач. Это достигается различными средствами. Во-первых, в каждой главе пособия [4] имеется специальный параграф, в котором строится математическая модель реальной задачи и приводится план соответствующего вычислительного эксперимента. Эти вычислительные эксперименты проводятся затем учащимися на лабораторных работах. Во-вторых, в пособии предлагается довольно большое число задач на составление математических моделей для самостоятельного решения. В-третьих, большое внимание уделяется использованию наиболее распространенных типов прикладных программ (информационно-поисковых систем, редактора текстов, электронной таблицы и т. п.) для решения соответствующих классов задач. На наш взгляд, именно такой подход к преподаванию курса ОИВТ способен уберечь этот курс от превращения в еще одну оторванную от реальной жизни дисциплину, каковыми, к сожалению, пока являются многие школьные предметы.

Мотивация к изучению конкретных тем создается и постановкой проблемы, и апелляцией к опыту учащихся, и демонстрацией тех или иных возможностей ЭВМ. Основную роль в создании мотивации играют, разумеется, объяснительные тексты. Кроме того, имеется целый ряд задач, которые тоже выполняют мотивационную функцию; мы будем указывать на нее при методическом разборе задач.

Классифицируя задачи из пособия [4] по сложности, можно выделить **четыре уровня**: к *первому* уровню относятся задачи, которые каждый ученик должен решать без особого труда; ко *второму* — задачи, доступные большинству учащихся; задачи *третьего* уровня доступны хорошо успевающим ученикам; задачи *четвертого* уровня предназначены для сильных учащихся. В нашем пособии в основном выдержана пропорция 6:4:3:1 распределения задач по этим уровням сложности. Разумеется, мы не сможем привести на страницах журнала разбор всех задач из [4] — их около пятисот. Полный разбор всех этих задач и методические указания по их решению приведены в пособиях [6—9].

Как было отмечено в [5], предлагаемые в [4] задачи можно условно разделить на четыре типа.

К первому типу относятся задачи, решения которых предполагают «ручную прокрутку». Например: «Какие значения примут переменные A и B в результате выполнения алгоритма?»

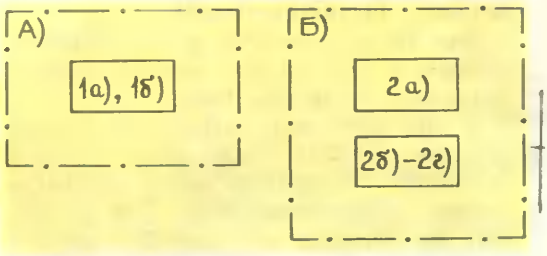
Одной из важных дидактических целей обучения является профилактика ошибок. Поэтому в пособии немало задач на нахождение ошибок (в модели, алгоритме, программе и т. д.) — это задачи второго типа. Той же цели служит и указание в объяснительном тексте приемов самоконтроля (например, методы отладки алгоритмов).

Третий тип задач — задачи на самостоятельное определение тех или иных элементов математических моделей, составление алгоритмов и программ, описание требований к вспомогательным алгоритмам и т. д.

Наконец, четвертый тип задач (быть может, самый важный) — задачи, в которых требуется осуществление всех этапов решения задач с помощью ЭВМ.

В данном цикле статей мы следуем структуре пособия [4], сохраняя названия глав и параграфов. В методических указаниях [6—9] дается распределение задач по блокам, объединяющим задачи, предназначенные для достижения конкретных дидактических целей. Для наглядности эти блоки изображаются прямоугольниками со штрихпунктирными границами. Внутри каждого блока задачи объединены в группы задач, эквивалентных с дидактической точки зрения. Эти группы заключены в прямоугольники со сплошной границей, расположенные один под другим: группа задач, расположенная ниже, предназначена для отработки более полной системы навыков. В свою очередь, задачи из группы, расположенной выше, могут рассматриваться как подготовительные к усвоению более на-

сыщенных задач. Мы считаем, что на занятиях должна быть разобрана как минимум одна задача из каждого блока, при этом обязательно разбирается задача из самой нижней группы блока.



Приведем пример. Эта схема показывает, что имеется два блока задач: А) и Б), предназначенных для отработки разных систем навыков. Решение задачи 2а) дает возможность отработать меньше навыков, чем решение любой из задач 2б)—2г). На занятиях должны быть разобраны задача 1а) или 1б) и одна из задач 2б)—2г). Задача 2а) может использоваться учителем для подготовки к решению задач 2б)—2г).

Данная статья посвящена разбору двух параграфов первой главы пособия [4].

Глава 1. Знакомство с ЭВМ

§ 1. Что умеет ЭВМ

Параграф носит ознакомительный характер, и его изучение рассчитано на один урок. Основная цель параграфа — пробудить интерес учащихся к предмету «Основы информатики и вычислительной техники». Обычно этот урок строится в форме беседы так, чтобы задаваемые учителем вопросы актуализировали знания учащихся о широких и всевозрастающих масштабах применения ЭВМ в различных областях жизни, почерпнутые ими из газет, радио- и телепередач. При этом следует обратить внимание учащихся на то, что сами по себе ЭВМ обладают лишь весьма ограниченными возможностями (грубо говоря, они «умеют» лишь складывать и вычитать). Тем самым создается проблемная ситуация: *как применять ЭВМ к решению реальных задач?* По нашему мнению, эта проблема является центральной в курсе ОИВТ.

§ 2. Этапы решения задач с помощью ЭВМ

В большинстве учебных пособий после начального рассказа о применениях ЭВМ тема «Этапы решения задач на ЭВМ» (т. е. обучение методологии применения ЭВМ) надолго откладывается. Например, в [1] она рассматривается в середине первой части (§ 5), в [3] — в начале второй части (глава 5), а в [2] отмечен единственный этап — составление алгоритма (§ 2.2).

Мы выносим тему «Этапы решения задач на ЭВМ» в начало курса по ряду причин.

1. Основную цель нашего курса ОИВТ — обучить школьников применению ЭВМ при решении реальных задач — нельзя достичь, не решая конкретных «жизненных» задач. Рассказывая с самого начала об этапах решения задач на ЭВМ, мы демонстрируем учащимся тот универсальный подход, который они осваивают в течение всего курса обучения.

2. Для достижения основной цели курса ОИВТ важно как можно полнее познакомить учащихся с основными классами задач, решаемых в настоящее время с помощью ЭВМ. Изучив эту тему с самого начала, учащиеся получают возможность достаточно глубоко и подробно разобрать задачи из самых различных областей человеческой деятельности (в пособии [4] разбираются задачи по физике, химии, биологии, экологии, криминалистике, лингвистике, математике и т. д.). Если же отложить эту тему на более поздний срок, то учащимся просто не хватит учебного времени.

3. Применение ЭВМ для решения «жизненных» задач играет большую мотивационную роль. Поэтому курс ОИВТ, в котором тема «Этапы решения задач на ЭВМ» разбирается в самом конце, можно уподобить пьесе, главный герой которой вообще не участвует в действии, а только выходит на поклоны в конце спектакля.

Разумеется, мы сознаем, что тема,

о которой идет речь, нуждается (как и любая другая тема) в хорошей методической проработке. Остановимся подробнее на методическом построении занятия по этой теме.

Созданная в конце предшествующего урока проблемная ситуация — как применять ЭВМ к решению реальных задач — подкрепляется конкретными примерами (скажем, задача управления установкой непрерывной разливки стали, планирования производства и реализации продукции какого-либо предприятия, организации продажи авиабилетов в кассах Аэрофлота и т. п.). Делается вывод о необходимости «перекидывания мостика» от реальной задачи к вычислениям на ЭВМ: Реальная задача →.....→ Вычисления на ЭВМ

Ясно, что *первым шагом в строительстве этого «моста» будет «перевод» задачи на язык математики, замена ее некоторой математической задачей.* Очевидно, что эта математическая задача должна отражать существо исходной задачи, как говорят, быть ее **математической моделью.**

Естественность употребления слова «модель» в данном контексте также нуждается в пояснении учащимся. Примеры физических моделей, приводимые учителем и самими учениками, позволяют сформулировать главное свойство любой модели — упрощать изучаемое явление с сохранением его существенных свойств.

Далее учитель рассказывает о том, как строить математическую модель. Примеры «жизненных» задач [4] показывают, что они, как правило, сформулированы весьма нечетко. Поэтому построение модели нужно начинать с четкой постановки задачи. Смысл слова «модель», разобранный ранее, подсказывает учащимся, что такая четкость достигается выделением существенных свойств реального объекта и формулированием упрощающих предположений (например, в задачах по кинематике обычно предполагают выполнимость законов Ньютона, а в задачах по химии закона Ломоносова—Лавуазье о сохранении массы). Искусство составления моделей как раз и заключается в том, чтобы, не переусложнив модель, учесть в ней все существенное и отбросить второстепенное.

Опыт решения задач по математике подсказывает учащимся, что в любой математической задаче требуется четко определить, что дано и что нужно получить. Поэтому, после того как сделаны предположения, нужно определить, что является исходными данными, а что — результатами.

Наконец, для завершения построения модели необходимо выразить в математических терминах связи между исходными данными и результатами.

Типичная ошибка, которую совершают учащиеся при построении модели,— это определение исходных данных и результатов до формулирования предположений. Зависимость выбора исходных данных от предположений легко проиллюстрировать [4, 10, 11]. Кроме того, желательно продемонстрировать учащимся, что неправильное или неполное формулирование предположений может приводить к неправильному выбору исходных данных и результатов. Вот возможный пример. Допустим, что при решении с помощью ЭВМ задачи составления школьного расписания забыли включить предположение об ограниченности числа уроков в день. Тогда и среди исходных данных не будет фигурировать ограничение на количество уроков и, скажем, у IX А в понедельник может оказаться 16 уроков, а во вторник и среду ни одного.

Возможность самоконтроля для учащихся обеспечивается четким пошаговым построением модели с указанием в явной форме: 1) предположений, 2) исходных данных и результатов, 3) связей между исходными данными и результатами.

Как правило, учащиеся хорошо знают из популярной литературы, что для решения задачи на ЭВМ необходимо составить инструкцию, в которой указывается последовательность действий, необходимых для решения задачи. Такая инструкция называется **алгоритмом.** *Составление алгоритма — второй этап решения задач на ЭВМ.* Запись алгоритма можно уподобить записи композитором сочиненного музыкального произведения на нотной бумаге. Такой «алго-

ритм» может исполнить, к примеру, симфонический оркестр. Если же мы это произведение хотим услышать «в исполнении» стереопроигрывателя, то записать его придется на грампластинку и вовсе не нотными знаками. Вывод: чтобы алгоритм был исполнен, его надо записать на понятном исполнителю языке. Здесь уместно попросить учащихся привести другие примеры исполнителей алгоритмов. Фактически это подготовка к введению понятия исполнителя, которое осуществляется в § 5 и 6.

После такого обсуждения учащимся уже несложно воспринять мысль о том, что для общения с ЭВМ нужен специальный язык. Алгоритмы, записанные на таком языке, называются программами. Поэтому сам язык общения с ЭВМ называют языком программирования.

Написание программы и проведение вычислений являются составной частью третьего этапа решения задач на ЭВМ. Обычно учащимся кажется, что на этом решение задачи завершается (здесь, по-видимому, отрицательную роль играет их опыт решения задач по другим предметам, поскольку учащиеся, по существу, имеют дело с уже готовыми моделями объектов и явлений). На самом же деле результаты расчетов на ЭВМ сопоставляются с экспериментальными фактами, теоретическими воззрениями и другой информацией об изучаемом объекте. Ведь сколько бы свойств объекта мы ни учитывали, модель всегда основана на некотором упрощении, и трудно быть абсолютно уверенным, что модель соответствует реальному объекту. И если обнаружатся противоречия, то производится корректировка модели. К примеру, ясно, как нужно скорректировать модель в задаче составления расписания уроков, о которой шла речь выше. Другой пример: допустим, нужно проложить кратчайший маршрут для самолета от одного города до другого. Если речь идет о Москве и Ленинграде, то мы не слишком ошибемся, предполагая, что Земля плоская, проложив и прямолинейный маршрут. Если это Москва и Владивосток, то прямолинейный маршрут будет отличаться от кратчайшего на сотни километров. Чтобы получить приемлемый результат, надо предположить, что Земля — шар, и проложить маршрут по дуге большого круга. Ведь хорошо известно, что на шаре кратчайший путь между двумя точками — дуга большого круга. Однако можно вспомнить про знаменитый полет Чкалова в Америку. Чкалов летел через Северный полюс, а вовсе не по дуге большого круга. Почему? Да потому, что, прокладывая столь дальние маршруты, приходится учитывать, что Земля сплюснута у полюсов и имеет форму не шара, а геоида. В этом случае на «конкурсе» моделей побеждает геоид.

Учащихся надо подвести к выводу, что выбор той или иной модели зависит от точности, с которой мы хотим получить результат. Для повышения точности модель приходится усложнять, учитывая все новые и новые особенности изучаемого объекта (правда, чем больше свойств мы учитываем в модели, тем, как правило, сложнее решение получающейся математической задачи).

Таким образом, *в третий этап решения задачи с помощью ЭВМ помимо написания программы и расчетов входит проведение вычислительного эксперимента и анализ результатов.*

Итак, полная схема решения задачи с помощью ЭВМ выглядит так:

В качестве дополнительной литературы по данной теме можно порекомендовать, например, [11] и [10].

Изложенный в пособии [4] теоретический материал закрепляется решением задач. В результате изучения материала параграфа и решения задач к нему учащиеся должны знать названия и содержание этапов решения задач с помощью ЭВМ, последовательность действий при построении мате-

матической модели. Учащиеся должны уметь выделять предположения и определять исходные данные и результаты в процессе построения простейших моделей.

Приведем условия и решения задач (классификация их по сложности и распределение на блоки и группы в соответствии с дидактическими целями приведены после разбора решений).

Задача 1. В комнате, имеющей двери и окна, будет проведен ремонт. Для расчета потребности в строительных материалах сделаны следующие предположения: пол, потолок, стены и окна имеют форму прямоугольников; дверь имеет форму прямоугольного параллелепипеда; батареи центрального отопления, расположенные под окнами, имеют прямоугольную форму.

Какие из этих предположений существенны для решения каждой из следующих задач:

1а) рассчитать количество мела, необходимое для побелки стен и потолка комнаты;

1б) рассчитать количество краски, необходимой для покраски пола;

1в) рассчитать количество обоев, необходимое для оклейки стен;

1г) рассчитать количество древесины, необходимое для изготовления двери.

Разбор задачи

При обсуждении задач 1а)—1г) надо иметь в виду, что они допускают несколько правомерных подходов, приводящих к разным решениям. Например, участок стены за батареями можно оклеить обоями, а можно побелить. Наличие различных подходов вообще типично для задач на построение моделей, и школьники в дальнейшем не раз столкнутся с такой ситуацией. Мы приводим одно из возможных решений.

1а) При побелке стараются, чтобы слой мела покрывал поверхность равномерно. Это значит, что количество мела пропорционально площади, которую надо побелить (к такому выводу учащихся не всегда бывает просто подвести!). Поэтому для решения задачи нужно знать площадь стен, двери, окон и потолка. Следовательно, существенными являются такие предположения:

потолок, стены и окна имеют форму прямоугольников, дверь — форму прямоугольного параллелепипеда.

Остальные предположения несущественны.

1б) Аналогично решению задачи 1а) делается вывод о пропорциональности количества краски площади пола. Следовательно, существенным является предположение:

пол имеет форму прямоугольника.

Остальные предположения несущественны.

1в) Площадь обоев равна площади стен за вычетом площади участков стены, расположенных за батареями, площади окон и дверей. Следовательно, существенными являются такие предположения:

стены и окна имеют форму прямоугольников;

дверь имеет форму прямоугольного параллелепипеда;

батареи центрального отопления, расположенные под окнами, имеют прямоугольную форму.

Остальные предположения несущественны.

1г) Количество древесины измеряется объемом двери. Поэтому существенно такое предположение:

дверь имеет форму прямоугольного параллелепипеда.

Остальные предположения несущественны.

Задача 2. Выскажите предположения, существенные для решения следующей задачи.

Участок цеха по производству туристского снаряжения выпускает брезентовые палатки. Требуется определить количество брезента, необходимое для выполнения участком месячного плана.

Разбор задачи

Эта задача допускает много различных решений, притом разной степени трудности. Самые простые предположения состоят в том, что выпускаются

палатки только одной модели, каждая палатка сшита из частей, имеющих форму прямоугольников, треугольников и трапеций. Школьницы обычно добавляют предположение о необходимости припусков на швы. Как правило, учащиеся неявно предполагают, что раскрой ткани производится без остатка. Целесообразно обсудить с ними правомерность такого предположения.

Задача 3. Для каждой из задач 1а)—1г) определите исходные данные и результаты.

Разбор задачи

Напомним, что задачи 3а) и 3б) недоопределены: в задаче 3а) не указаны предположения о расходе мела при побелке поверхностей, в задаче 3б) не указаны аналогичные предположения о расходе краски, хотя фактически они сформулированы в ходе решения задач 1а) и 1б).

3а) Исходными данными являются количество дверей и окон; длина, ширина и высота комнаты, ширина и высота двери и окна; масса мела, расходуемого на 1 м^2 площади. Результатом является масса мела.

3б) Исходными данными являются длина и ширина комнаты; масса краски, расходуемой на 1 м^2 площади. Результатом является масса краски.

3в) Исходными данными являются количество дверей и окон; ширина и высота стены, двери, окна и батарей. Результатом является площадь обоев.

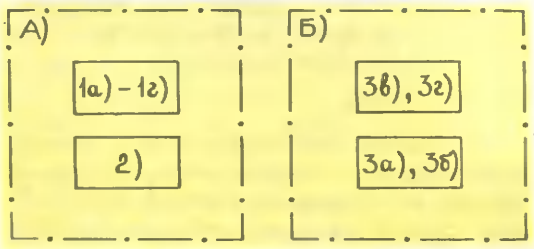
3г) Исходными данными являются ширина, высота и толщина двери. Результатом служит объем двери.

Все задачи данного параграфа относятся к первому уровню сложности.

Приведем распределение этих задач по блокам и группам.

При решении задач блока А) отрабатываются навыки выделения предположений, а в задачах блока Б) требуется определить исходные данные и результаты. При этом в задаче 1 учащиеся должны просто выбрать нужные предположения из уже имеющегося списка, а в задаче 2 — сформулировать их самостоятельно. При решении задачи блока Б) следует еще раз обратить внимание школьников на то, что в задачах на построение математических моделей исходные данные и результаты существенно зависят от сделанных предположений. Кроме того, в задачах 3а)—3в) сформулирован заведомо неполный набор предположений и учащиеся должны его самостоятельно дополнить. Такая недоопределенность — также общая черта «модельных» задач, на которую следует обратить внимание школьников.

Продолжение следует.



Литература

1. Основы информатики и вычислительной техники: Пробное учебное пособие. Ч. I, II / Под ред. А. П. Ершова, В. М. Монахова. М.: Просвещение, 1985, 1986.

2. *Ершов А. П., Кушниренко А. Г., Лебедев Г. В.* и др. Основы информатики и вычислительной техники. М.: Просвещение, 1988.

3. *Каймин В. А., Щеголев А. Г., Ерохина Е. А.* и др. Информатика: Методическая разработка. Ч. I, II. М.: МИЭМ, 1987.

4. *Гейн А. Г., Житомирский В. Г., Линецкий Е. В.* и др. Основы информатики и вычислительной техники. Свердловск: УрГУ, 1989.

5. *Гейн А. Г., Житомирский В. Г., Линецкий Е. В., Сапир М. В., Шолохович В. Ф.* Программно-методический комплекс по курсу школьной информатики // Информатика и образование. 1988. № 3, № 4.

6. *Гейн А. Г., Сапир М. В., Шолохович В. Ф.* и др. Дидактические материалы по курсу информатики. Гл. 1—3. Свердловск: СвГПИ, 1988.

7. *Гейн А. Г., Сапир М. В., Шолохович В. Ф.* и др. Дидактические материалы по курсу информатики. Гл. 4—6. Свердловск: СвГПИ, 1989.

8. *Гейн А. Г., Линецкий Е. В., Сапир М. В.* и др. Дидактические материалы по курсу информатики. Гл. 7, 8. Свердловск: СвГПИ, 1989.

9. *Гейн А. Г., Линецкий Е. В., Сапир М. В.* и др. Дидактические материалы по курсу информатики. Гл. 9, 10. Свердловск: СвГПИ, 1989.

10. *Гейн А. Г., Ковальджи А. К., Сапир М. В.* Задачи, модели и ЭВМ // Квант. 1989. № 3.

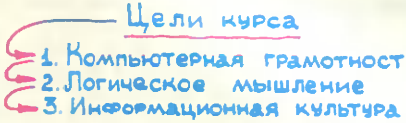
11. Компьютеры, модели, вычислительный эксперимент: Введение в информатику с позиций математического моделирования. М.: Наука, 1988.

В. КАЙМИН, Н. УГРИНОВИЧ

О преподавании курса ОИВТ по машинному варианту

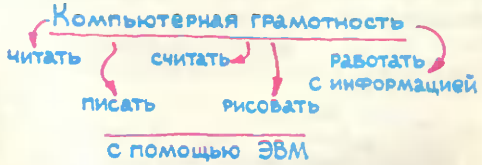
Оснащение более половины школ Москвы кабинетами вычислительной техники позволило в 1988/89 учебном году полностью перейти на преподавание курса ОИВТ по машинному варианту как в IX, так и в X классах. Учителям было предоставлено право выбора преподавания информатики по одному из двух учебников, ориентированных на машинный вариант курса,— переработанному учебнику под редакцией А. П. Ершова или экспериментальному учебнику под редакцией В. А. Каймина.

Большинство районных методических объединений учителей информатики (31 из 33) выбрали экспериментальный учебник «Основы информатики и вычислительной техники» (авторы В. А. Каймин, А. Г. Щеголев, Е. А. Ерохина, А. П. Федюшин), который занял второе место на конкурсе учебников по информатике, проводившемся МП СССР летом 1987 г. Таким образом в Москве начался крупномасштабный эксперимент по преподаванию курса ОИВТ по данному учебнику.

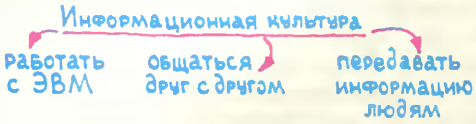


Основные цели курса в этом учебнике совпадают с требованиями конкурсной программы (опубликованной в журналах «Информатика и образование», 1986, № 2 и «Математика в школе», 1986, № 3). Задачей-минимум является овладение всеми учащимися компьютерной грамотностью, а задачей-максимум — формирование у них информационной культуры.

Понятие компьютерной грамотности в экспериментальном учебнике связывается с общим понятием грамотности, формируемой в начальных классах, где детей учат читать и писать, считать и рисовать, работать с книгами и тетрадями. Таким образом, компьютерная грамотность заключается в формировании аналогичных навыков при работе с ЭВМ, т. е. умений читать и писать, считать и рисовать, искать информацию и работать с программами на ЭВМ.

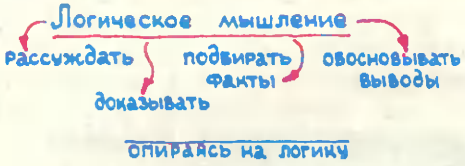


Понятие информационной культуры в экспериментальном учебнике самым тесным образом связывается с предметом информатики как научной дисциплины, изучающей законы и методы накопления и обработки информации на ЭВМ, в общении людей и в жизни общества.

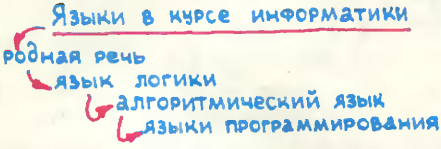


В работе с ЭВМ информационная культура раскрывается в умениях решать задачи с помощью ЭВМ, что, в соответствии с конкурсной программой, предполагает формирование умений ставить задачи, создавать математические

модели, составлять алгоритмы решения задач на ЭВМ, записывать алгоритмы в форме программ и проводить их отладку, знать возможности ЭВМ и уметь интерпретировать результаты, получаемые с их помощью.



Не менее важной составляющей в информационной культуре является гуманитарная часть — умение общаться друг с другом и предоставлять людям информацию в различных формах. В общении друг с другом важны умения выслушать чужую точку зрения и с уважением относиться к чужому мнению, умения излагать свою точку зрения и доказывать свою правоту, умения находить общие решения и составлять программы совместной деятельности для достижения общих целей. Перечисленные умения в свою очередь предполагают необходимость определенной логической культуры — умений рассуждать, ставить вопросы, давать четкие ответы, доказывать и обосновывать свои выводы.



Материалы экспериментального учебника формировались с 1983 г. в процессе работы основной группы авторов в физико-математической школе при Московском институте электронного машиностроения. В течение 1987/88 учебного года этот учебник проходил интенсивную экспериментальную проверку в 20 школах Москвы. В процессе этой работы дорабатывались учебник и программное обеспечение к лабораторному практикуму, а также подготавливались задачник и методическое пособие для учителей.

Для постановки обучения информатике по экспериментальному учебнику существенно, что методика преподавания по нему машинно независима и обучение можно проводить на различных видах учебной ВТ: КУВТ-86, «Агат», ДВК, «Ямаха», «Корвет», УКНЦ — на основе единой гибкой методики. Это обеспечивает возможность достижения учащимися единого уровня знаний и умений, отработку методики преподавания и использования передового педагогического опыта.

Опыт преподавания по учебнику в московских школах и выявившаяся необходимость формирования определенного уровня логической культуры у учащихся для овладения информационной культурой обусловили внесение изменений в программу курса ОИВТ по сравнению с конкурсной программой. Уточненная программа курса представляет некоторую перекомпоновку конкурсной программы с добавлением элементов логики, языка ПРОЛОГ и методов анализа правильности алгоритмов.

Уточненная программа курса «Основы информатики и вычислительной техники» IX—X классы (102 ч)

1. Введение в предмет информатики. Информационные процессы. Понятие и виды информации. Количество информации. Сведения, суждения и рассуждения. Элементы логики. Законы логики.

2. Основные возможности ЭВМ. Персональные ЭВМ. Правила работы с ними. Редактирование текстов на ЭВМ. Выполнение рисунков и чертежей на ЭВМ. Поиск информации на ЭВМ. Организация баз знаний и элементы языка Пролог. Численные расчеты на ЭВМ. Основные типы программ на персональных ЭВМ.

3. Основы алгоритмизации. Задачи и их решение на ЭВМ. Понятие, примеры и свойства алгоритмов. Средства ввода-вывода информации. Выбор и повторение действий в алгоритмах. Вспомогательные алгоритмы. Использование массивов. Организация диалоговых алгоритмов.

4. Начала программирования. Язык программирования Бейсик и правила работы с ним на персональной ЭВМ. Графические программы на Бейсике. Программы с циклами и ветвлениями. Организация подпрограмм на Бейсике. Переменные и массивы в Бейсике. Организация диалоговых программ.

5. Решение задач на ЭВМ. Этапы решения задач. Математические постановки и методы решения задач. Сценарии диалоговых программ. Конструирование и пошаговая детализация алгоритмов. Приближенное решение задач. Подбор тестов для проверки программ.

6. Основания информатики. Понятие правильности алгоритмов и программ. Анализ правильности выполнения линейных, разветвляющихся и циклических алгоритмов. Анализ правильно-

сти вспомогательных алгоритмов. Вычислительные эксперименты на ЭВМ. Понятие и виды моделей. Информационно-логические модели и задачи. Законы информатики.

7. Основы вычислительной техники. Архитектура и виды ЭВМ. Логические элементы ЭВМ. Команды и представление данных. Внешние устройства ЭВМ. Программное обеспечение ЭВМ. Языки и технология программирования. История развития ЭВМ.

8. ЭВМ в развитии общества. ЭВМ в сельском хозяйстве и на производстве. ЭВМ в проектировании и управлении. ЭВМ в науке и образовании. ЭВМ и информационные коммуникации.

Учебное пособие, написанное в соответствии с приведенной уточненной программой, рассчитано на систематическое использование ЭВМ на каждом уроке. В этих целях для ЭВМ «Ямаха» созданы образцы программ, необходимых для постановки лабораторного практикума по овладению компьютерной грамотностью и изучению применений ЭВМ. Для практикума по составлению алгоритмов и программ предполагается использование диалоговых языков программирования — Бейсика, Рапиры или Фокала, имеющихся на школьных ЭВМ.

В целях поддержки этих практикумов подготавливается сборник задач и упражнений и методическое пособие с рекомендациями по проведению уроков в кабинетах вычислительной техники.

Основной особенностью постановки преподавания курса в условиях систематического использования ЭВМ является взаимосогласованное изложение теории и практики с опережающим изложением практических вопросов работы на ЭВМ и апробации учащимися теоретических положений в упражнениях на машине.

С этой целью по требованию московских учителей и методистов в учебнике синхронизировано изложение теоретических и практических вопросов в 1-й и 2-й, 3-й и 4-й, 5-й и 6-й главах.

Структура распределения времени на уроках определяется учителем. Однако рекомендуется соотношение 1:1:1 между лекционной, семинарской и лабораторной формами проведения урока. Это соответствует соотношению объема знаний, умений и навыков, определяемых в предмете «информатика» как учебной дисциплине. Такое распределение времени желательно сохранять на каждом уроке, если занятия проводятся в кабинете вычислительной техники.

Если уроки проводятся попеременно в классе (теория) и в кабинете ВТ или в школе и в учебном компьютерном центре, рекомендуется применять блочное группирование уроков с соотношением времени 1:1:1 в изучении теоретических сведений, решении задач и работы на ЭВМ.

При работе в кабинете с 12—15 компьютерами класс должен быть разделен на две подгруппы. Каждый ученик должен иметь возможность проработать на машине весь цикл лабораторных работ по овладению компьютерной грамотностью и применением ЭВМ, а также практикум по составлению и отладке на ЭВМ простейших программ.

Общее количество часов, отводимых на курс информатики, составляет 102, как и в государственной программе, однако желателен перенос 1 ч в неделю из второго полугодия X класса во второе полугодие IX класса. Это позволяет изучать наиболее сложные и важные темы курса в наиболее подходящее время, а курс разделить на 4 (по полугодиям) относительно самостоятельных блока.

IX класс

IX(1) — компьютерная грамотность — 17 ч.

IX(2) — начала программирования — 34 ч.

X класс

X(1) — решения задач на ЭВМ — 34 ч.

X(2) — возможности и применение ЭВМ — 17 ч.

Первое полугодие IX класса посвящается овладению учащимися компьютерной грамотностью. Стержнем этого полугодия служат лабораторные работы с клавиатурным тренажером, редактором текста, графическим редактором, учебной базой данных, электронными таблицами и базой знаний на основе языка Пролог. Главной задачей является приобретение учащимися устойчивых навыков в работе с информацией на ЭВМ. Вторая и не менее важная задача — усвоение основной системы понятий ин-

форматики и формирование минимального уровня логической культуры. Общее число часов в полугодии — 17 (1 урок в неделю).

Задачи IX (1)

1. Умение подготовки информации на ЭВМ
2. Знание элементов логики и основных понятий информатики

Второе полугодие IX класса направлено на формирование умений составления алгоритмов и отладки на ЭВМ простейших программ с наглядным представлением результатов. Принципиальным с методической точки зрения является обязательное составление учащимися под руководством и контролем учителя алгоритмов и сценариев программ. Это позволяет выявить ошибки в программах до их ввода в компьютер и оценивать правильность самостоятельно составляемых учениками программ. Важно, что элементы языка программирования вводятся последовательно в процессе кодирования все более сложных алгоритмов. Целью полугодия является поэтапное освоение элементов языка программирования, алгоритмического языка и приобретение умений самостоятельно составлять простейшие программы для ЭВМ. Объем работы этого полугодия — 34 ч (2 урока в неделю, желательно сдвоенных).

Задачи IX (2)

1. Умение составлять простейшие алгоритмы и сценарии работы с ЭВМ
2. Умение записывать простейшие программы и производить их отладку на ЭВМ

Первое полугодие X класса опирается на навыки и умения, приобретенные в IX классе и направлено на решение главной задачи курса — формирование информационной культуры, прежде всего приобретение представлений о постановках задач, математических и информационно-логических моделях, методах задач и оценки их правильности, о проведении экспериментов и получении результатов с помощью вычислительных машин. Объем времени — 34 ч (2 сдвоенных урока в неделю).

Задачи X (1)

1. Представления о постановке задач, подборе моделей и методов их решения
2. Опыт проведения вычислительных экспериментов на ЭВМ

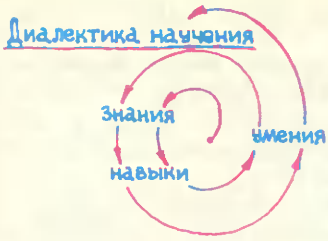
Второе полугодие X класса служит в основном для демонстрации возможностей применений ЭВМ в различных сферах деятельности и знакомству с основными видами вычислительных машин, перспективами их развития и использования в обществе. Для этого во втором полугодии достаточно 1 ч в неделю (17 ч).

Задачи X (2)

1. Знание структуры и принципов работы вычислительных машин
2. Знание возможностей применения ЭВМ

Особое значение имеет зачетная форма контроля знаний и умений учащихся. В курсе информатики целесообразно отказаться от такой формы итогового контроля, как контрольная работа в традиционном ее понимании (убрать с парт учебники, тетради и т. д.), когда проверяется преимущественно степень механического запоминания учащимися материала. Наоборот, контроль должен ориентировать учащихся на умение пользоваться необходимыми в процессе решения задачи источниками информации, а в перспективе должен прививать учащимся умения пользоваться современными информационными технологиями.

Такую роль могут выполнять тематические и итоговые зачетные работы. Тематику таких работ, причем обязательно вариативную, желательно дать учащимся в начале прохождения темы. Например, при изучении графики можно в качестве зачетной предложить учащимся составить программу, демонстрирующую графические возможности ЭВМ. Такая постановка задачи обеспечит вариативность, более того, индивидуальность решения, каждый учащийся представит свою уникальную программу, в которой продемонстрирует свой уровень усвоения материала.



В соответствии с изложенным распределением времени и ориентацией на блочно-урочное построение занятий с использованием ЭВМ предлагается следующее планирование курса.

Тематическое планирование занятий

IX КЛАСС (1)

ТЕМЫ: «Введение в информатику» и «Работа на ЭВМ» 17 ч

1.1. *Информация и ее подготовка на ЭВМ* 6 ч

Знакомство с ЭВМ. Техника безопасности и правила работы на ЭВМ. Виды и понятия информации. Количество информации

1.2. *Законы логики и поиск информации на ЭВМ* 7 ч

Учебная база данных. Вопросы и суждения. База знаний и элементы языка Пролог. Утверждения и рассуждения. Работа с языком Пролог. Законы логики

1.3. *Численные расчеты на ЭВМ* 3 ч

Калькуляторы и электронные таблицы на персональных ЭВМ. Задачи и их решения. Численные расчеты на компьютерах

1.4. *Зачет по самостоятельно созданной базе данных* 1 ч

IX КЛАСС (2)

ТЕМЫ: «Начала программирования» и «Основы алгоритмизации» 34 ч

2.1. *Понятие алгоритма и подготовка программ на ЭВМ* 4 ч

Понятие алгоритма. Элементы языка программирования. Правила работы с программами на ЭВМ. Порядок составления программ

2.2. *Средства ввода/вывода и машинной графики на ЭВМ* 3 ч

Средства ввода/вывода. Составление и отладка графических программ. Кодирование алгоритмов на Бейсике. Понятие сценария

2.3. *Ветвление и циклы в алгоритмах и программах* 7 ч

Алгоритмы и программы с выбором действий. Кодирование условий и выбор действий на Бейсике. Алгоритмы и программы с циклом. Кодирование циклов

2.4. *Вспомогательные алгоритмы и подпрограммы* 6 ч

Вспомогательные алгоритмы. Их кодирование на Бейсике. Аргументы и результаты выполнения вспомогательных алгоритмов. Составление и отладка программ и подпрограмм

2.5. *Обработка таблиц с использованием массивов* 6 ч

Понятие таблицы и массивов. Алгоритмы поиска информации в массивах. Составление и отладка программы с использованием массивов.

2.6. *Диалоговые алгоритмы и программы* 6 ч

Средства организации диалога с персональными ЭВМ. Составление и отладка диалоговой программы на основе задуманного сценария.

2.7. *Зачет по самостоятельно составленной диалоговой программе* 2 ч

X КЛАСС (1)

ТЕМЫ: «Решение задач на ЭВМ» и «Основания информатики» 34 ч

3.1. *Постановки задач и методы их решения* 6 ч

Этапы решения задач на ЭВМ. Анализ выполнения алгоритмов и классификация ошибок. Постановки задач

3.2. *Методы решения задач и анализ алгоритмов* 8 ч

Анализ правильности линейных и ветвящихся алгоритмов. Реккурентные методы. Анализ правильности циклических алгоритмов

3.3. *Сценарии программ и конструирование алгоритмов* 6 ч

Сценарии диалоговых программ. Систематическое конструирование и пошаговая детализация алгоритмов. Анализ правильности сложных алгоритмов

3.4. *Постановка вычислительных экспериментов* 6 ч

Приближенные методы решения задач. Понятие и примеры вычислительных экспериментов. Понятие и виды моделей

3.5. *Информационно-логические задачи и модели* 6 ч

Информационно-логические модели. Решение задач машинами. Законы информатики

3.6. *Зачет по самостоятельно составленной программе и эксперименты с ней на ЭВМ* 2 ч

X КЛАСС (2)

ТЕМЫ: «Основы вычислительной техники» и «Применения ЭВМ в обществе» 17 ч

4.1. *Архитектура и принципы работы ЭВМ* 3 ч

Архитектура ЭВМ. Физические принципы работы ЭВМ. Процессор и программный принцип работы вычислительной техники. Внешние устройства ЭВМ. Типы ЭВМ

4.2. *Логические принципы работы ЭВМ* 3 ч

Логические элементы ЭВМ. Логические функции и логические блоки ЭВМ. Примеры логических блоков

4.3. *Программное обеспечение ЭВМ* 3 ч

Базовое программное обеспечение ЭВМ. Типы языков программирования. Понятие о технологии разработок программных средств ЭВМ

4.4. *Применения ЭВМ в обществе* 7 ч

ЭВМ: история, настоящее и будущее. ЭВМ в сельском хозяйстве и на производстве. ЭВМ в управлении и обслуживании. ЭВМ и информационные коммуникации

4.5. *Зачет: доклад или реферат на тему о применениях ЭВМ* 1 ч

Продолжение следует.

Литература

1. *Каймин В. А., Щеголев А. Г., Ерохина Е. А., Федюшин Д. П.* Информатика: Методическая разработка для учащихся XI-X классов средних школ и техникумов. М., 1987.

2. Основы информатики и вычислительной техники: Пробное учебное пособие для IX—X классов / Под ред. А. П. Ершова, В. П. Монахова. Ч. 1, 2. М.: Просвещение, 1985, 1986.

3. *Каймин В. А., Питеркин В. М.* Основы информатики и вычислительной техники: Пособие для учителей средних школ. М.: МИЭМ, 1985.

4. *Каймин В. А.* Методы разработки программ на языках высокого уровня. М.: МИЭМ, 1985.

5. *Каймин В. А.* Основы доказательного программирования. М.: МИЭМ, 1987.

6. *Арменский Е. В., Каймин В. А., Питеркин В. М.* // Проблемы компьютерного обучения. М.: Знание, 1986.

7. *Афанасьев В. Н., Каймин В. А., Питеркин В. М.* Опыт МИЭМ в разработке единой методики обучения информатики и программирования в вузах и школах // Применение ЭВМ в учебном процессе. М.: ВЗМИ, 1986.

8. *Каймин В. А.* Статьи по методам составления и анализ правильности алгоритмов // Квант. 1986. № 10, 11; 1987. № 6.

9. *Угринович Н. Д., Козбенко О. Ю.* Преподавание курса «Основы информатики и вычислительной техники (машинный вариант)»: Методические рекомендации. М.: МГИУУ, 1988.

10. *Угринович Н. Д.* Информатика: теория, методика, задачи: Методические рекомендации. М.: МГИУУ, 1988.

А. КУШНИРЕНКО

О новом учебнике информатики

В 1988 г. издательство «Просвещение» выпустило новый пробный учебник тиражом 139 тыс. экземпляров*. К 1990 г. планируется выпустить переработанный вариант учебника, методическое пособие для учителей, комментарии с решением всех задач и упражнений.

В предлагаемом вниманию читателей новом учебнике информатики для старших классов сохранена преемственность с первыми, ныне действующими пробными учебными пособиями для IX и X классов, используется тот же школьный алгоритмический язык, та же система понятий. Однако методика введения этих понятий существенно переработана: они вводятся на наглядном геометрическом материале, что заметно упрощает изложение.

Опыт показал, что учебник можно использовать как в условиях ограниченного доступа к вычислительной технике, так и при обучении по безмашинному варианту. Но наибольшая эффективность может быть достигнута при условии регулярного доступа к КУВТ «Корвет», УКНЦ, «Ямаха» MSX-1, MSX-2 и КУВТ-86. В настоящее время материал IX класса на 90 % поддерживается программным обеспечением на указанных КУВТ.

Требования к новому учебнику

Основной недостаток первых пробных учебных пособий — чрезмерная математизация. В них делается попытка объяснить основы алгоритмизации на математических примерах: игра Баше, наибольший общий делитель, решение квадратного уравнения и т. д. Однако в настоящее время значительная часть учащихся имеет слабую математическую подготовку, плохо владеет математической символикой. Для таких учащихся даже простое упоминание математических понятий или обозначений приводит

* *Ершов А. П., Кушниренко А. Г., Лебедев Г. В., Семенов А. Л., Шень А. Х.* Основы информатики и вычислительной техники: Пробный учебник для средних учебных заведений / Под ред. А. П. Ершова. М.: Просвещение, 1988. 207 с.: ил.

к потере интереса, неверию в собственные возможности. Если школьник не знает определения НОД и, главное, совершенно не понимает, зачем нужен этот НОД, то ему нельзя объяснить необходимость введения команды повторения на примере алгоритма НОД. Если школьник с трудом решает квадратные уравнения, то попытка объяснить команду ветвления на примере алгоритма КВУР заранее обречена на неудачу. Поэтому первое требование к учебнику, сформулированное авторским коллективом, следующее:

сделать учебник доступным реальной массовой школе, ориентируясь на средний уровень математической подготовки учащихся.

Главное методическое новшество при решении этой задачи — попытка изложить основные понятия информатики и основные конструкции программирования, опираясь на наглядные, житейские, геометрические представления школьников, не привлекая на первых порах ни математических понятий, ни, по возможности, математической символики. Для этого выбран такой подход: все конструкции алгоритмического языка вводятся не на примерах алгоритмов вычисления значений тех или иных величин, а на примерах алгоритмов управления несколькими автоматическими устройствами. В основном тексте учебника таких устройств четыре.

Робот — идеализированная транспортная тележка, перемещающаяся по клетчатой плоскости в четырех направлениях;

Чертежник — идеализированный графопостроитель, умеющий рисовать отрезки на бесконечной плоскости (бесконечном листе бумаги);

Измеритель — идеализированный набор датчиков, который может быть установлен на Роботе, перемещаться вместе с ним и измерять различные величины при движении Робота;

Черепашка — исполнитель с системой команд обычной «черепашки» в языке Лого, воспроизведенной в пробном учебном пособии для IX класса.

В классе алгоритмов управления перечисленными автоматическими устройствами можно сформулировать широкий спектр задач, начиная от элементарных упражнений на закрепление материала и кончая трудными творческими задачами олимпиадного уровня сложности. Это позволяет варьировать индивидуальную глубину проработки материала, сохраняя единым для всего класса темп прохождения новых понятий.

Бесспорная удача первых пробных учебных пособий — школьный алгоритмический язык. С одной стороны, он весьма прост и доступен школьникам не меньше, чем Бейсик (чего не скажешь о Паскале). С другой — он содержит «общий знаменатель» всех современных процедурных языков программирования: команды ветвления и повторения, процедуры с параметрами и т. д., что позволяет понятно и кратко записывать алгоритмы. По мнению авторов, в настоящее время алгоритмический язык освоен корпусом преподавателей информатики нашей страны (хотя преподавание на этом языке ведется далеко не всюду). Отсюда вытекает второе требование к учебнику:

сохранение школьного алгоритмического языка, описанного в первом пробном учебном пособии.

Это требование удалось выполнить на 99 %: языки отличаются лишь несколькими второстепенными деталями.

Трудной для повседневной работы учителя оказалась некоторая расплывчатость алгоритмического языка, предложенного в первых пробных учебных пособиях. Это не позволяло вводить и требовать от учеников четкую форму записи алгоритмов.

Справедливые нарекания вызывало изложение трех языков: школьного алгоритмического языка, Бейсика, Рапиры. При отсутствии компьютера разницу между ними понять трудно. При наличии компьютера понять, почему на доске мы пишем на алгоритмическом языке, предназначенном для людей, а на компьютере работаем на Бейсике, еще труднее. Попытки внедрения методики перевода с алгоритмического языка на производственные (Бейсик, Фокал, Паскаль) наталкивались на справедли-

вое негодование: заставляют делать лишнюю работу. Итак, третье требование:

в учебнике должен описываться только один алгоритмический язык — школьный. Именно он должен использоваться для записи алгоритмов и для их выполнения на ЭВМ. На школьных ЭВМ этот язык не должен уступать Бейсику по основным производственным характеристикам.

Часть требований выходит за рамки учебника и связана с эффективной реализацией школьного алгоритмического языка на школьных ЭВМ. Эта задача для «Корвета», УКНЦ и «Ямахи» силами временного коллектива «Школа-1» и механико-математического факультета МГУ должна быть решена к лету 1990 г. (Приверженцы Бейсика смогут назвать соответствующую программную систему Русский Бейсик и тем самым поместить ее в ряд языков, далеких от Бейсика, но развивающих его положительные качества: Быстрый Бейсик, Истинный Бейсик, Турбо-Бейсик и т. д.).

Козьма Прутков заметил: «Многие вещи нам непонятны не потому, что наши понятия слабы; но потому, что сии вещи не входят в круг наших понятий». Непонимание алгоритмов, неумение их составлять при работе с первыми пробными учебными пособиями были часто связаны с тем, что основные конструкции алгоритмического языка вводились формально, без предварительной мотивации. Четвертое требование:

предварительная мотивация всех конструкций и понятий алгоритмического языка.

Для его выполнения изложение в новом учебнике ведется проблемно, *«от частного к общему»*. Все конструкции алгоритмического языка вводятся по схеме:

наглядная задача, требующая для решения введения новой конструкции;

пример использования конструкции при решении этой задачи;

общий вид конструкции, ее формальное описание.

Рассмотрим нашу оценку места предмета «информатика» в школьном образовании. Среди многообразия целей общего среднего образования выделим две: а) дать общую картину окружающего мира, показать наиболее существенные процессы, протекающие в природе и обществе; б) дать набор знаний, умений, навыков, без которых нельзя обойтись человеку в современном обществе.

В XX в. компьютеры и управляющие ими алгоритмы стали существенной частью окружающего нас мира. Не овладев понятием алгоритма теоретически, нельзя представить себе этот мир во всей его полноте. С другой стороны, широкое внедрение компьютеров во все сферы жизни общества требует от его членов новых навыков — навыков «алгоритмирования». Без этих практических навыков в современном обществе ни управиться с собственными часами, ни купить билет на поезд, ни справиться с редактором текстов на собственном компьютере. Составить алгоритм — значит обдумать план некоторой будущей деятельности и фиксировать его с помощью системы обозначений, с помощью какого-то языка. В этой формуле важны оба члена: и обдумывание алгоритма, и его фиксация на некотором заранее выбранном языке. И тому, и другому мы должны научить в курсе информатики, избрав в качестве способа фиксации алгоритма школьный алгоритмический язык. Отсюда следует наше последнее требование:

учебник должен сформировать теоретическое представление об алгоритмах, способах их записи. Учебник должен обеспечить практическое освоение алгоритмического языка, навыков алгоритмизации. В частности, он должен быть снабжен запасом задач и упражнений, достаточным для формирования домашних заданий к каждому уроку, фронтальных опросов и практикумов, контрольных и т. д.

Особенности изложения алгоритмического языка, правил работы с автоматическими устройствами

В отличие от пробных учебных пособий в новом учебнике школьный алгоритмический язык излагается не как система обозначений, границы ко-

торой расплывчаты, а как раз и навсегда фиксированный язык программирования. Разумеется, не все возможности этого языка будут использоваться и изучаться в школьном курсе одинаково интенсивно. Тем не менее психологически очень важно, что «правила игры» заранее определены: любые вопросы о допустимости той или иной формы записи, смысле той или иной конструкции могут быть решены однозначно. (Формальное описание языка будет опубликовано вместе с описанием его реализации на школьных ЭВМ.)

В новом учебнике отвергнута метафора исполнения алгоритма собственно автоматическим устройством, т. е. «обучение» исполнителя новым командам за счет составления новых алгоритмов. Методически это неудобно по многим соображениям: трудно понять, кто же исполняет алгоритм одновременного управления несколькими автоматическими устройствами; для каждого автоматического устройства возникает свой, особый алгоритмический язык и т. д.

Основная метафора учебника такова: *«ЭВМ исполняет алгоритм, записанный на школьном алгоритмическом языке, отдавая при этом команды автоматическим устройствам и получая от них информацию»*. Тем самым автоматическое устройство выполняет лишь простые команды, предусмотренные в его конструкции, а вся работа по исполнению алгоритма, по исполнению команд алгоритмического языка проводится внутри ЭВМ. Сам школьный алгоритмический язык одинаков при работе с любыми автоматическими устройствами: подключение к ЭВМ нового устройства лишь добавляет возможность вызова команд этого устройства. И по форме записи, и по смыслу вызовы команд автоматических устройств идентичны вызовам вспомогательных алгоритмов. Каждое из автоматических устройств, в свою очередь, также имеет раз и навсегда фиксированный набор команд (предусмотренный «заводом-изготовителем»). Этот набор команд не расширяем. Написание новых вспомогательных алгоритмов приводит к расширению библиотеки этих алгоритмов, а вовсе не к изменению системы команд автоматического устройства. Такой подход позволяет четко разделить изучение собственно алгоритмического языка и изучение возможностей тех или иных автоматических устройств, управляемых алгоритмами на этом языке.

Машинный и безмашинный варианты работы с учебником

Результаты работы перечисленных автоматических устройств легко изображаются на классной доске, в тетради. Это позволяет вести безмашинное изложение, выполняя алгоритмы «вручную», т. е. моделируя работу ЭВМ и автоматических устройств. Оказался плодотворным следующий методический прием: на уроке один ученик выступает в роли ЭВМ и исполняет алгоритм управления некоторым автоматическим устройством. Другой ученик выступает в роли этого автоматического устройства и «выполняет команды ЭВМ» (изображая результаты их исполнения на доске). Принятая в учебнике модель памяти ЭВМ также наглядна, операциональна и позволяет демонстрировать на классной доске ход исполнения алгоритмов, работающих с величинами.

На серийных КУВТ «Корвет», УКНЦ, «Ямаха» (и частично на КУВТ-86, КУВТ-87) четыре автоматических устройства, изучаемых в учебнике, промоделированы программно, с использованием графического экрана. Это позволяет составлять на школьном алгоритмическом языке и исполнять алгоритмы управления, например, «Роботом», наблюдая результаты работы на экране ЭВМ.

В перспективе можно ожидать появления набора серийных реальных автоматических устройств, подключаемых к школьному компьютеру. В этой ситуации алгоритмы управления, скажем, «Роботом» можно будет выполнить как в режиме моделирования — изображение «Робота» перемещается на экране компьютера, так и в режиме реального управления — «Робот» перемещается по столу или по классу.

Продолжение следует.

Н. ГЛАГОЛЕВА, Н. ЮНЕРМАН
Г. НОВОСИБИРСК

Основные понятия языка Рапира

Ветвления и циклы

С помощью одного и того же текста можно описать различную по содержанию информацию. Например, текст ЛБММРППП может описывать информацию о составе поезда, если символы Л, Б, М, Р и П обозначают названия вагонов (Локомотив, Багажный, Мягкий, Ресторан, Плацкартный), или о наборе игрушек в магазине, если эти же символы обозначают их названия (Лото, Барабан, Мяч, Рыбка, Пистолет).

Как определить число мягких вагонов в составе поезда, количество мячей в магазине? Фактически это одна и та же задача. Действительно, для решения каждой из них нужно посчитать, сколько раз в тексте встречается буква М.

В дальнейшем мы будем рассматривать различные задачи, связанные с обработкой текстов, не задумываясь о том, какая информация описывается с помощью данного текста.

Составим алгоритм для подсчета числа букв М в произвольном тексте.

1. Заведем счетчик — на чистом листе бумаги будем записывать число букв М, найденных к данному моменту. Сначала запишем число 0 (еще не найдено ни одной буквы М).

2. Возьмем первый символ текста.

3. Сравним его с буквой М. Если он является буквой М, то зачеркнем ранее написанное число и запишем рядом другое, полученное из предыдущего увеличением на 1, в противном случае этого делать не будем.

4. Перейдем к взятию следующего символа текста.

Будем повторять действия 3 и 4 до тех пор, пока не будут «обработаны» все символы текста. Число, записанное на листе бумаги последним (оставшееся незачеркнутым), является результатом решения нашей задачи. При подсчете количества букв М в тексте МАРМЕЛАД записи на листе бумаги после просмотра всего текста будут иметь вид:

Ø 1 2

Число 2 — искомый результат.

Мы хотим поручить машине выполнить этот алгоритм. Роль листа бумаги будет играть блок памяти, например, с именем СЧЕТЧИК. Его начальным значением является число 0. При обнаружении в тексте очередного символа М значение этого имени надо увеличить на 1. Это можно сделать с помощью предписания присваивания СЧЕТЧИК:=СЧЕТЧИК+1

Взять один символ из текста можно, как вы помните, с помощью операции выборки. А как сравнить выбранный символ с символом М и в зависимости от результата выполнить различные действия? Это можно сделать с помощью *условного предписания*. В Рапире условное предписание записывается так:

ЕСЛИ условие
  ТО предписания
  ИНАЧЕ предписания
ВСЕ

Условие — это два выражения, соединенных одним из знаков:

= (равно) /= (не равно)

> (больше) >= (больше или равно)

< (меньше) <= (меньше или равно)

Условие может быть истинным или ложным. Примеры истинных условий: 15>7*2, 18>=18, «РОГ»=«ПРОГРАММА» [2:4]. Ложными условиями являются 7=8, «СЛОН»=«СЛОГ», 20>197.

Если в условие входит имя, то от его значения зависит, будет условие истинным или ложным. Например, условие А>10 истинно, если значение А равно 15, и ложно, когда значение А равно —35.

Продолжение. Начало см.: Информатика и образование. 1989. № 1.

Тексты можно сравнивать только на совпадение или несовпадение.

Как исполняется условное предписание? Сначала проверяется условие. Если оно истинно, то выполняются предписания, записанные после слова ТО, в противном случае — предписания, записанные после слова ИНАЧЕ. Слово ИНАЧЕ и следующие за ним предписания могут отсутствовать. Тогда, если условие ложно, никакие действия не выполняются.

Допустим, что в блоках с именами А и В хранятся числа. Тогда после исполнения предписания

```
ЕСЛИ А>В
   ТО ВЫВОД:А
   ИНАЧЕ ВЫВОД:В
ВСЕ
```

будет выдано большее из этих двух чисел.

После слов ТО и ИНАЧЕ могут быть записаны любые предписания языка, в том числе и условные.

Задача 8. Выполните ручную прокрутку программ, если значением имени А является число —56, имени В — число 34, имени С — число 100.

```
а) ВЫВОД:"ДАНЫ ЧИСЛА:",А," И",В
   ЕСЛИ А>В
     ТО     МАКС:=А; МИН:=В
     ИНАЧЕ МАКС:=В; МИН:=А
   ВСЕ
   ВЫВОД:"МАКСИМАЛЬНОЕ ЧИСЛО=",МАКС
   ВЫВОД:"МИНИМАЛЬНОЕ ЧИСЛО=",МИН
б) ВЫВОД:"ДАНЫ ЧИСЛА:",А,",",В," И",С
   ЕСЛИ А>В
     ТО     ЕСЛИ А>С
              ТО     МАКС:=А
              ИНАЧЕ МАКС:=С
            ВСЕ
     ИНАЧЕ ЕСЛИ В>С
              ТО     МАКС:=В
              ИНАЧЕ МАКС:=С
            ВСЕ
   ВСЕ
   ВЫВОД:"МАКСИМАЛЬНОЕ ЧИСЛО=",МАКС
в) МОДУЛЬ:=А-В+С
   ЕСЛИ МОДУЛЬ<0
    ТО  МОДУЛЬ:=-МОДУЛЬ
   ВСЕ
   ВЫВОД:"МОДУЛЬ(А-В+С)=",МОДУЛЬ
```

Задача 9. Измените программу б) из задачи 8 так, чтобы с ее помощью можно было найти минимальное из трех чисел.

Вернемся к подсчету числа букв М в тексте. Обозначим рассматриваемый текст именем Т, а индекс выбираемого символа — НОМ. Тогда сравнение выбранного символа с М и, в случае их совпадения, увеличение счетчика на единицу можно выполнить с помощью предписания

```
ЕСЛИ Т[НОМ]=«М» ТО
   СЧЕТЧИК:=СЧЕТЧИК+1
ВСЕ
```

Чтобы завершить решение задачи, познакомимся с тем, как выполнить некоторую последовательность предписаний несколько раз.

Как правило, многократное выполнение последовательности предписаний требуется в следующих ситуациях:

известно, сколько раз надо выполнить данные действия (повторение);

известно условие, при соблюдении которого надо повторять выполнение данных действий (условие продолжения);

известна последовательность объектов (например, чисел), с которыми надо выполнить одни и те же действия (перечисление или перебор);

известно условие, при соблюдении которого выполнение надо закончить (условие окончания).

Такое многократное повторение записывается с помощью *предписания цикла*. В Рапире есть несколько вариантов этого предписания, которые позволяют запрограммировать каждую из перечисленных ситуаций. Какой из них лучше использовать в каждом конкретном случае, решает программист. Для решения одной и той же задачи могут быть использованы разные варианты предписания цикла.

Цикл ПОВТОР. Число повторений известно заранее.

```
ПОВТОР выражение
ЦИКЛ
   предписания
КЦ
```

В этом случае значением выражения, записанного после слова ПОВТОР, должно быть натуральное число или нуль, иначе будет выдано сообщение об ошибке. Как работает такое предписание? Сначала вычисляется значение выражения, а затем предписания, следующие между словами ЦИКЛ и КЦ, выполняются заданное число раз. Если значение выражения равно нулю, то предписания не выполнятся ни разу.

Например, после исполнения программы

```
Т1:="**********"
Т2:="*         *"
ВЫВОД:Т1
ПОВТОР  13
ЦИКЛ
  ВЫВОД:Т2
КЦ
ВЫВОД:Т1
```

будет напечатан квадрат из *.

Первая строка в предписании цикла называется заголовком цикла. Предписания, записанные между словами ЦИКЛ и КЦ (Конец Цикла), называются телом цикла. Тело цикла может содержать любые предписания, в том числе и предписания цикла.

Цикл ПОКА. Известно условие, при соблюдении которого надо выполнять тело цикла.


```
ПОКА условие
ЦИКЛ
    предписания
КЦ
```

Исполнение этого предписания начинается с проверки условия. Если оно истинно, то выполняется тело цикла и опять проверяется условие. Выполнение цикла прекращается, когда при очередной проверке условие окажется ложным. Если условие ложно при первой проверке, то цикл не выполняется ни разу! А если условие тождественно истинно, т. е. никогда не может стать ложным (например, 5=5), то произойдет зацикливание, т. е. тело цикла будет исполняться бесконечное число раз.

Следующая программа печатает все целочисленные решения уравнения $X**2-X-2=0$ на интервале [—1, 2].

```
НАЧ:=-1
КОН:=2
Х:=НАЧ
КОНТРОЛЬ:="НЕТ"                    \ *
ПОКА  Х<=КОН
ЦИКЛ
 ЕСЛИ  ХХХ2-Х-2=0
  ТО  ВЫВОД:"Х=",Х
      КОНТРОЛЬ:="ДА"               \ *
 ВСЕ
 Х:=Х+1
КЦ
ЕСЛИ-КОНТРОЛЬ="НЕТ"                \ *
 ТО  ВЫВОД:"НЕТ РЕШЕНИЙ"           \ *
 ИНАЧЕ ВЫВОД:"РЕШЕНИЯ НАЙДЕНЫ"     \ *
ВСЕ                                \ *
```

Обратите внимание, что хотя бы одна из переменных, входящих в условие цикла, должна изменять свое значение в теле цикла. Иначе произойдет зацикливание.

Переменная КОНТРОЛЬ «отвечает» за наличие корней уравнения на данном интервале. Если есть хотя бы один корень, то будет напечатано его значение и сообщение РЕШЕНИЯ НАЙДЕНЫ, а в случае отсутствия корней будет выдано сообщение НЕТ РЕШЕНИЙ. Если из программы удалить строки, в которых стоит комментарий *, то контроля за наличием корней не будет и в случае их отсутствия ничего не будет напечатано.

Задача 10. Выполните ручную прокрутку этой программы.

Задача 11. Измените эту программу так, чтобы с ее помощью можно было определить, существуют ли целочисленные решения уравнения $X**2+X-2=0$ на интервале [—2, 1]. Выполните ручную прокрутку полученной программы.

Цикл ДЛЯ. Выполнение действий с последовательностью чисел. Последовательность задается начальным и конечным значениями и величиной шага.

```
ДЛЯ имя ОТ нач ДО кон ШАГ величина
шага
ЦИКЛ
    предписания
КЦ
```

Имя, записанное после слова ДЛЯ, называется *переменной цикла.* Начальное и конечное значения (нач и кон) и величина шага — выражения. Выполнение этого предписания начинается с вычисления значений всех трех выражений.

Затем переменной цикла присваивается начальное значение и выполняется тело цикла. После этого переменной цикла присваивается следующее значение, которое получается из предыдущего увеличением его на величину шага (если величина шага — число отрицательное, то переменная цикла уменьшается). Далее проверяется, не выходит ли оно за пределы отрезка [нач, кон] (или [кон, нач] в случае отрицательного шага); если не выходит, снова выполняется тело цикла.

Если величина шага и разность между

конечным и начальным значениями имеют разные знаки, то тело цикла не выполняется ни разу (проверка знаков на совпадение производится компьютером сразу после вычисления начального и конечного значений и величины шага).

Внутри тела цикла запрещено присваивать новые значения переменной цикла!

Рассмотрим, что будет выдано после исполнения следующих циклов.

1)
```
ДЛЯ X ОТ 3 ДО 7 ШАГ 2
ЦИКЛ
   ВЫВОД: X
КЦ
```
Будет выдано:
3
5
7

2)
```
ДЛЯ X ОТ 7 ДО 3 ШАГ —2
ЦИКЛ
   ВЫВОД: X
КЦ
```
Будет выдано:
7
5
3

3)
```
ДЛЯ X ОТ 3 ДО 6 ШАГ 2
ЦИКЛ
   ВЫВОД: X
КЦ
```
Будет выдано:
3
5

4)
```
ДЛЯ X ОТ 3 ДО 3 ШАГ —2
ЦИКЛ
   ВЫВОД: X
КЦ
```
Будет выдано:
3

5)
```
ДЛЯ X ОТ 3 ДО 1 ШАГ 2
ЦИКЛ
   ВЫВОД: X
КЦ
```
В этом случае ничего не будет выдано.

С помощью следующей программы можно расположить символы текста в обратном порядке.

```
Т:="НАЛИМ"
ВЫВОД:"ДАНО СЛОВО: ",Т
РЕЗ:=""
ДЛЯ НОМ ОТ #Т ДО 1 ШАГ -1
ЦИКЛ
  РЕЗ:=РЕЗ+Т[НОМ]
КЦ
ВЫВОД:"ПОЛУЧИЛОСЬ СЛОВО: ",РЕЗ
```

Слово ШАГ и следующее за ним выражение могут отсутствовать. В этом случае величина шага считается равной единице.

Задача 12. Запишите решение задачи о расположении в обратном порядке символов текста, используя в цикле ДЛЯ положительное значение величины шага.

Еще с одним вариантом предписания цикла мы познакомим вас позже.

Запишем теперь решение задачи о подсчете числа букв М в тексте, используя различные варианты предписания цикла. Договоримся, что в этой и в следующих задачах этот текст хранится в блоке памяти с именем Т.

1. Цикл ПОВТОР. Действие сравнения символа из текста с символом М надо выполнить столько раз, сколько всего символов в тексте, т. е. #Т раз.

```
СЧЕТЧИК:=0
НОМ:=1
ПОВТОР  #Т
ЦИКЛ
  ЕСЛИ Т[НОМ]="М"
    ТО СЧЕТЧИК:=СЧЕТЧИК+1
  ВСЕ
  НОМ:=НОМ+1 \ к следующему символу
КЦ
ВЫВОД:"В ТЕКСТЕ",СЧЕТЧИК," БУКВ М"
```

2. Цикл ПОКА. Условие продолжения имеет вид НОМ<= #Т, где НОМ — индекс символа к тексте.

```
СЧЕТЧИК:=0
НОМ:=1
ПОКА НОМ<=#Т
ЦИКЛ
  ЕСЛИ Т[НОМ]="М"
    ТО СЧЕТЧИК:=СЧЕТЧИК+1
  ВСЕ
  НОМ:=НОМ+1 \ к следующему символу
КЦ
ВЫВОД:"В ТЕКСТЕ",СЧЕТЧИК," БУКВ М"
```

3. Цикл ДЛЯ. Последовательность чисел — индексы символов текста.

```
СЧЕТЧИК:=0
ДЛЯ НОМ ОТ 1 ДО #Т
ЦИКЛ
  ЕСЛИ Т[НОМ]="М" ТО
    СЧЕТЧИК:=СЧЕТЧИК+1
  ВСЕ
```

```
КЦ
ВЫВОД:"В ТЕКСТЕ",СЧЕТЧИК," БУКВ М"
```

Сравним три последние программы. Обратите внимание на то, что при использовании циклов ПОВТОР и ПОКА выполнено начальное присваивание переменной НОМ и тело цикла содержит предписание, изменяющее значение этой переменной. При использовании цикла ДЛЯ этого делать не нужно. Объясните почему.

Программа, использующая цикл ДЛЯ, является в данном случае наиболее компактной и наглядной.

Задача 13. Перепишите следующие предписания цикла с помощью цикла ПОКА.

а)
ПОВТОР а
ЦИКЛ
...
КЦ
б)
ДЛЯ х ОТ а ДО в
ЦИКЛ
...
КЦ

Задача 14. Составьте программу для нахождения суммы квадратов натуральных чисел от 1 до 25. Решите эту задачу тремя способами, используя различные варианты предписания цикла.

Составим программу, которая определяет, сколько в данном тексте слов и сколько предложений. Договоримся, что в тексте слова отделяются друг от друга лишь одним из символов: запятой, пробелом, двоеточием. А предложения — точкой, вопросительным или восклицательным знаком (без пробела после них).

Число предложений определяется по количеству точек, восклицательных и вопросительных знаков в тексте. А чтобы определить число слов, достаточно подсчитать количество разделителей между ними и добавить число знаков в конце предложений.

Можно, конечно, составить шесть очень похожих программ для определения числа точек, пробелов и т. д. Но тогда текст надо будет просматривать шесть раз! А нельзя ли за один просмотр текста выполнить все необходимые вычисления? Так как нам нужно знать только общее число точек, вопросительных и восклицательных знаков в тексте, то для их подсчета достаточно иметь один счетчик (СЧ__ПРЕДЛ). Другой счетчик потребуется для подсчета слов (СЧ__СЛОВА). Дальше можно поступить следующим образом. Каждый символ текста сначала сравнить с пробелом, если они совпадают, увеличить счетчик СЧ__СЛОВА на единицу, а если нет, то сравнить этот же символ с запятой, и т. д. Если символ совпал с одним из разделителей предложений, то увеличиваем другой счетчик (СЧ__ПРЕДЛ). Это можно записать с помощью условного предписания.

```
ЕСЛИ Т[НОМ]="␣"
 ТО СЧ_СЛОВА:=СЧ_СЛОВА+1
 ИНАЧЕ
  ЕСЛИ Т[НОМ]=","
   ТО СЧ_СЛОВА:=СЧ_СЛОВА+1
   ИНАЧЕ
    ЕСЛИ Т[НОМ]=":"
     ТО СЧ_СЛОВА:=СЧ_СЛОВА+1
     ИНАЧЕ
      ЕСЛИ Т[НОМ]="."
       ТО СЧ_ПРЕДЛ:=СЧ_ПРЕДЛ+1
       ИНАЧЕ
        ЕСЛИ Т[НОМ]="!"
         ТО СЧ_ПРЕДЛ:=СЧ_ПРЕДЛ+1
         ИНАЧЕ
          ЕСЛИ Т[НОМ]="?"
           ТО СЧ_ПРЕДЛ:=СЧ_ПРЕДЛ+1
          ВСЕ
        ВСЕ
      ВСЕ
    ВСЕ
  ВСЕ
ВСЕ
```

Запись такого условного предписания малонаглядна: трудно проследить, что должно выполняться при соблюдении каждого из условий. В языке есть специальное предписание ВЫБОР, более удобное в таких случаях. Оно записывается так:

ВЫБОР
ПРИ условие 1: предписания
ПРИ условие 2: предписания
...
ИНАЧЕ предписания
ВСЕ

Как исполняется такое предписание? Сначала проверяется условие 1; если оно истинно, то выполняются предписания, записанные после него, и исполнение предписания ВЫБОР завершается. Если условие 1 ложно, то прове-

ряется условие 2. Если оно истинно, то выполняются предписания, записанные после него, и исполнение предписания ВЫБОР завершается. Если условие 2 ложно, то проверяется условие 3, и т. д. Если ложны все условия, записанные после слов ПРИ, то выполняются предписания, указанные после слова ИНАЧЕ. Как и в условном предписании, слово ИНАЧЕ и следующие за ним предписания могут отсутствовать.

Число слов и предложений в тексте можно найти с помощью следующей программы.

```
СЧ_СЛОВА:=0; СЧ_ПРЕДЛ:=0
ДЛЯ НОМ ОТ 1 ДО 80
ЦИКЛ
 ВЫБОР
  ПРИ Т[НОМ]="␣": СЧ_СЛОВА:=СЧ_СЛОВА+1
  ПРИ Т[НОМ]=",": СЧ_СЛОВА:=СЧ_СЛОВА+1
  ПРИ Т[НОМ]=":": СЧ_СЛОВА:=СЧ_СЛОВА+1
  ПРИ Т[НОМ]="!": СЧ_ПРЕДЛ:=СЧ_ПРЕДЛ+1
  ПРИ Т[НОМ]=".": СЧ_ПРЕДЛ:=СЧ_ПРЕДЛ+1
  ПРИ Т[НОМ]="?": СЧ_ПРЕДЛ:=СЧ_ПРЕДЛ+1
 ВСЕ
КЦ
СЧ_СЛОВА:=СЧ_СЛОВА + СЧ_ПРЕДЛ
ВЫВОД:"В ТЕКСТЕ ЕСТЬ:"
ВЫВОД:"СЛОВ – ",СЧ_СЛОВА
ВЫВОД:"ПРЕДЛОЖЕНИЙ – ",СЧ_ПРЕДЛ
```

Обратите внимание на условия в предписании ВЫБОР, записанные после слов ПРИ. Выражения, стоящие слева от знака равенства, у них совпадают. В таких случаях можно использовать другой вариант предписания ВЫБОР, в котором выражение, являющееся общей частью всех условий, записывается в заголовке.

Теперь предписание ВЫБОР из последней программы можно записать так:

```
ВЫБОР Т[НОМ]
 ПРИ "␣": СЧ_СЛОВА:=СЧ_СЛОВА+1
 ПРИ ",": СЧ_СЛОВА:=СЧ_СЛОВА+1
 ПРИ ":": СЧ_СЛОВА:=СЧ_СЛОВА+1
 ПРИ ".": СЧ_ПРЕДЛ:=СЧ_ПРЕДЛ+1
 ПРИ "!": СЧ_ПРЕДЛ:=СЧ_ПРЕДЛ+1
 ПРИ "?": СЧ_ПРЕДЛ:=СЧ_ПРЕДЛ+1
ВСЕ
```

Случаи, когда при разных значениях выражения, записанного после слова ВЫБОР, надо выполнять одни и те же действия, можно объединить. Тогда все выражения записываются после слова ПРИ через запятую. Например:

```
ВЫБОР Т[НОМ]
 ПРИ "␣" , "," , ":" :
                СЧ_СЛОВА:=СЧ_СЛОВА+1
 ПРИ "." , "!" , "?"  :
                СЧ_ПРЕДЛ:=СЧ_ПРЕДЛ+1
ВСЕ
```

Решения задач

Задача 8.

а) Таблица значений имен

Имя	Значение
А	—56
В	34
МАКС	34
МИН	—56

Будет выдано:
ДАНЫ ЧИСЛА: —56 и 34
МАКСИМАЛЬНОЕ ЧИСЛО=34
МИНИМАЛЬНОЕ ЧИСЛО=—56

б) Таблица значений имен

Имя	Значение
А	—56
В	34
С	100
МАКС	100

Будет выдано:
ДАНЫ ЧИСЛА: —56, 34 и 100
МАКСИМАЛЬНОЕ ЧИСЛО= 100

в) Таблица значений имен

Имя	Значение
А	—56
В	34
С	100
МОДУЛЬ	10

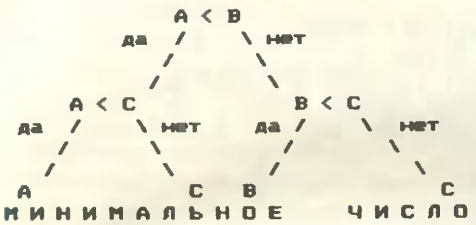
Будет выдано:
МОДУЛЬ (А—В+С)=10

Задача 9.

Возможное решение:

```
ВЫВОД:"ДАНЫ ЧИСЛА:",A,",",B," И",C
ЕСЛИ A<B
  ТО     ЕСЛИ A<C
           ТО     МИН:=A
           ИНАЧЕ МИН:=C
         ВСЕ
  ИНАЧЕ ЕСЛИ B<C
           ТО     МИН:=B
           ИНАЧЕ МИН:=C
         ВСЕ
ВСЕ
ВЫВОД:"МИНИМАЛЬНОЕ ЧИСЛО=",МИН
```

Значения имен A, B и C выбраны в условии задачи произвольно. Для ребят, не справившихся с этим заданием, можно предложить разобрать, например, такую схему:



Задача 10.

Таблица значений имен

Имя	Значение
НАЧ	−1
КОН	2
X	−1 0 1 2
КОНТРОЛЬ	НЕТ ДА

Будет выдано:
X=−1
X=2
РЕШЕНИЯ НАЙДЕНЫ

Задача 11.

```
НАЧ:=-2
КОН:=1
X:=НАЧ
КОНТРОЛЬ:="НЕТ"
ПОКА  X<=КОН
ЦИКЛ
  ЕСЛИ  X**2+X-2=0
    ТО  КОНТРОЛЬ:="ДА"
  ВСЕ
  X:=X+1
КЦ
ЕСЛИ КОНТРОЛЬ="НЕТ"
  ТО  ВЫВОД:"РЕШЕНИЙ НЕТ"
  ИНАЧЕ ВЫВОД:"РЕШЕНИЯ ЕСТЬ"
ВСЕ
```

Таблица значений имен

Имя	Значение
НАЧ	−2
КОН	1
X	−2 −1 0 1
КОНТРОЛЬ	НЕТ ДА

Будет выдано:
РЕШЕНИЯ ЕСТЬ

Задача 12.

```
T:="НАЛИМ"
ВЫВОД:"ДАНО СЛОВО: ",T
РЕЗ:=""
ДЛЯ НОМ ОТ 1 ДО 5T
ЦИКЛ
  РЕЗ:=T[НОМ] + РЕЗ
КЦ
ВЫВОД:"ПОЛУЧИЛОСЬ СЛОВО: ",РЕЗ
```

Задача 13.

а)
```
СЧЕТЧИК:=0
ПОКА СЧЕТЧИК<=A
ЦИКЛ
  ...
  СЧЕТЧИК:=СЧЕТЧИК+1
КЦ
```

б)
```
X:=A
ПОКА X<=B
ЦИКЛ
  ...
  X:=X+1
КЦ
```

Задача 14.

1) Цикл ДЛЯ

```
СУММА:=0
ДЛЯ N ОТ 1 ДО 25
ЦИКЛ
 СУММА:=СУММА + N**2
КЦ
ВЫВОД:"1**2 + 2**2 + ... + 25**2=",СУММА
```

2) Цикл ПОКА

```
СУММА:=0
N:=1
ПОКА  N<=25
ЦИКЛ
 СУММА:=СУММА + N**2
 N:=N + 1
КЦ
ВЫВОД:"1**2 + 2**2 + ... + 25**2=",СУММА
```

3) Цикл ПОВТОР

```
СУММА:=0
N:=1
ПОВТОР  25
ЦИКЛ
 СУММА:=СУММА + N**2
 N:=N + 1
КЦ
ВЫВОД:"1**2 + 2**2 + ...+ 25**2=",СУММА
```

Продолжение следует.

М. ЖИТОМИРСКИЙ, В. КЛАВДИЕВ

О методике преподавания информатики в средней школе

В предлагаемой статье сделана попытка оценить методики изложения школьного курса основ информатики и предлагается еще один подход, в основу которого положен опыт преподавания курса. Эффективность методик оценивается по времени, необходимому для достижения поставленных целей, информативной наполненности курса и логике построения изложения.

Целью введенного в школьную программу курса основ информатики является ликвидация компьютерной безграмотности. Однако не всегда правильное (возможно, излишне буквальное) понимание смысла компьютерной грамотности привело к некоторым, с нашей точки зрения, негативным результатам.

В качестве примера можно было бы взять студенческую аудиторию первых курсов институтов. Проведенные опросы показали неоднородность подготовки студентов и еще большую неоднородность оценки собственных знаний. Каждый из этих аспектов требует анализа, оба имеют один корень: разные условия, в которых находятся преподаватели (и учащиеся), и отсутствие такой методики преподавания предмета, которая легко бы адаптировалась к различным условиям, но давала одинаковые результаты.

Отметим два полярных подхода к организации преподавания.

Первый реализован в школах, имеющих широкий доступ к вычислительной технике и считающих необходимым использовать ее как можно раньше (лучше — сразу). Стремление к раннему использованию вычислительной техники сужает, с точки зрения авторов, информатику до изучения одного языка программирования, до овладения данным компьютером. А такие важные вопросы, как, например, формальная постановка задачи или разработка алгоритмов, остаются малоизученными. Ученику отводится роль не разработчика алгоритмов, а пользователя, или, скорее, оператора ЭВМ. У некоторых учащихся после первых успехов в общении с компьютером появляется ошибочное ощущение, что они знают информатику. Как следствие часто возникает некоторый внутренний протест при попытке расширить знания рассмотрением таких, например, вопросов, как отладка и тестирование алгоритмов, доказательство их правильности, применение различных формальных систем для записи алгоритмов и т. д.

Второй подход к преподаванию информатики воплощается в жизнь там, где вычислительной техники нет, и перспектива доступа к ней неопределенна. Такое положение вещей вызывает естественное неудовольствие и у преподавателей, и у учащихся. Однако мнение, что информатика в этом случае отодвигается на задний план, нельзя считать верным. Даже при «безмашинном» варианте обучения можно сообщить учащимся много полезных и интересных сведений и добиться определенных успехов в овладении основами предмета.

Такие точки зрения на преподавание информатики, как показал опрос первокурсников, настолько распространены, что может показаться, что «золотая середина» скорее является исключением.

Наша методика может быть реализована как в условиях достаточно широкого, так и ограниченного доступа к вычислительной технике (например, только во время летней практики). В последнем случае образующееся «лишнее» время может быть заполнено более подробным рассмотрением отдельных глав курса или изучением дополнительного (и, безусловно, не лишнего) материала, приведенного в конце данной работы.

Предлагаемая авторами методика основана на том, что информатика, как и любой другой учебный предмет, имеет своей целью знакомство учащихся с основами изучаемой науки, изложение тех

необходимых знаний, умений и навыков, которые могут быть применены в дальнейшей практической деятельности.

Поэтому при планировании и преподавании курса в первую очередь следует решить два вопроса: определить информатику как науку и очень подробно обсудить ее положение в системе других наук.

И если необходимость определения информатики не может вызвать никаких сомнений, то решение второго вопроса очертит нам круг решаемых информатикой задач, продемонстрирует ее прикладной характер, покажет связь информатики с другими науками. А это решит сразу две проблемы: обеспечит учителя бесконечным множеством задач и даст основу для разработки методов их решения, которые могут быть использованы другими школьными предметами. Например, применение начальных сведений из алгебры и логики позволит учащимся без большого труда усвоить такие понятия, как логические константы, составные условия и т. п.

Небесполезно продемонстрировать учащимся, какие практические результаты они смогли бы извлечь, применяя знания, полученные на уроках информатики, при решении задач из других школьных дисциплин. Можно, например, показать, что если при решении иррационального неравенства

$$\sqrt{f(x)} > \varphi(x)$$

записать (в любой доступной форме) алгоритм его решения то, во-первых, в случае действий в соответствии с предписаниями алгоритма может быть получен правильный ответ, а во-вторых, становится очевидной теорема равносильности:

$$\sqrt{f(x)} > \varphi(x) <=> \begin{cases} \varphi(x) \geqslant 0 \\ f(x) > \varphi^2(x) \end{cases} \quad \begin{cases} \varphi(x) < 0 \\ f(x) \geqslant 0 \end{cases}$$

После того как стало понятно место информатики в системе других наук, можно перейти к детальному ее изучению, начать которое следует с определения понятия задачи, предлагаемой информатике для решения.

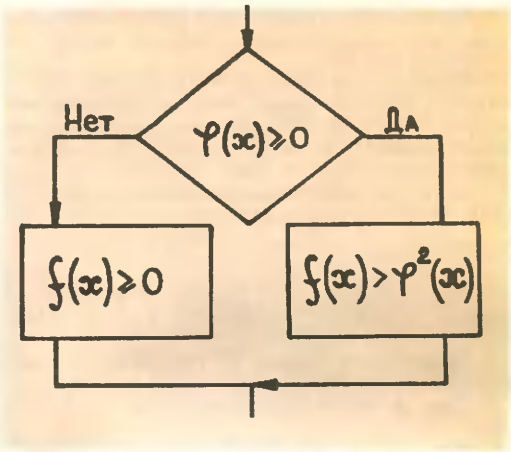
Необходимо подчеркнуть, что задача может иметь как формальную постановку (например, из 11 чисел выбрать наибольшее), так и неформальную (из 11 членов футбольной команды выбрать капитана). В случае, когда задача поставлена неформально, заботу о ее формализации берет на себя информатика. Так, для приведенного примера нужно показать, что задача о выборе капитана выглядит как задача поиска наибольшего из 11 элементов (выбор капитана происходит голосованием и подсчетом числа голосов, набранных игроками).

После того как вопрос о формальной постановке задачи обсужден, можно перейти к выбору метода ее решения, что зависит от удобства его использования и границ применения.

Именно здесь наступает время для введения понятия алгоритма. Но к разработке алгоритма, реализующего выбранный метод решения задачи и доводящего решение его до результата, можно приступать только после того, как задача формализована и выбран метод ее решения.

Следует отметить, что разработчик и исполнитель алгоритма не обязательно одно и то же лицо. А если так, то алгоритм обязательно должен быть понятен исполнителю, обладать свойством однозначности. Для этого алгоритм должен быть записан на каком-либо языке, обеспечивающем однозначность. Поскольку естественные языки, как правило, этим свойством не обладают, встает вопрос о выборе формального языка для записи алгоритмов решения задачи. Здесь полезно перечислить уже известные учащимся формальные системы — арифметику и алгебру, характеризуя при этом их выразительные возможности. Можно показать, что арифметика, например, удачно справляется с процессом сложения, но малопригодна для описания такого простого процесса, как покупка мороженого.

После того как учащиеся убедятся, что известные им формальные системы не всегда могут их удовлетворить, им может быть предложен новый способ записи алгоритмов. Нами выбран язык блок-схем как наиболее простой, достаточно выразительный и весьма распространенный. Кроме того, при использовании этого языка хорошо прослежи-



ваются такие свойства алгоритма, как дискретность, однозначность, результативность.

Язык блок-схем может быть изучен по достаточно традиционному плану (примерное планирование дано в приложении). Единственная рекомендация — ввести понятие операции присваивания уже при рассмотрении линейных алгоритмов. Это облегчит запись вычислительных алгоритмов и не будет усложнять восприятие последующего более трудного материала. Изучение любого нового элемента языка блок-схем обязательно должно сопровождаться новыми яркими иллюстративными примерами, как вычислительными, так и невычислительными.

Параллельно с изучением языка блок-схем до сведения учащихся доводится, что совсем не обязательно сразу детализировать алгоритм. Иногда достаточно построить его из довольно крупных блоков-подзадач. Если исполнителю понятна каждая из выделенных подзадач, то дальнейшую детализацию алгоритма можно не проводить.

Кроме того, необходимо обратить внимание на то, что процесс разработки алгоритма уже с начальной стадии должен сопровождаться проектированием системы его проверки, а каждый шаг детализации алгоритма — его тестированием и отладкой. При этом следует обязательно отметить, что задачей тестирования является поиск ошибки, а не доказательство ее отсутствия.

После изучения всех перечисленных вопросов — постановки задачи, выбора методов ее решения, разработки алгоритма, его тестирования и отладки, а также языка блок-схем возникает вполне естественное желание поручить исполнение алгоритма компьютеру. Несмотря на многие свои преимущества, язык блок-схем компьютеру непонятен. Поэтому появляется необходимость в перекодировании (!) уже разработанного нами алгоритма на какой-либо реализованный на имеющемся компьютере язык. Авторами в качестве такого языка был выбран Фортран-IV.

Изучение языка программирования происходит как подбор аналогов конструкций, известных из уже изученного языка программирования. Однако довольно много времени и внимания приходится уделять синтаксису изучаемого языка. Одновременно учащиеся знакомятся и с новыми командами — операторами, которых не было в языке блок-схем: DO, операторами описания, FORMAT и др. К этому моменту их необходимость и полезность становятся очевидными.

После ознакомления с основными операторами дальнейшее изучение языка программирования сопровождается лабораторными работами на имеющемся компьютере. Если есть возможность выбора, предпочтение отдается персональным ЭВМ. В школах, не имеющих такой возможности, обучение после языка блок-схем можно продолжить с использованием микрокалькуляторов с программным управлением, занимаясь фактически программированием в кодах. Здесь также удобно переводить разработанный алгоритм с языка блок-схем на язык микрокалькулятора.

С точки зрения авторов, все перечисленное составляет минимальный курс, с которым учащиеся должны быть ознакомлены в процессе изучения основ информатики.

В случае, если позволяют временные рамки, например для школ с углубленным изучением математики или если время высвобождается из-за ограниченных возможностей проведения лабораторных работ, учащиеся могут быть ознакомлены с перечисленными ниже важными разделами науки.

1. Операционные системы, история их развития, взаимное влияние развивающихся аппаратного и математического обеспечения вычислительных систем. Способ обработки программ пользователя в вычислительной системе [10].

2. Другие формальные системы, их отличие от изученных, при этом обращается внимание на их новые выразительные свойства. Могут быть выбраны, например, математическая логика, теория конечных автоматов [6, 7, 9].

3. Доказательство правильности программ [7], [8], при этом подчеркивается отсутствие общего определения правильной программы и обращается внимание на выбор формальной системы, подходящей для выполнения работы такого рода; это позволит изучить понятие математической модели и изоморфизма.

4. Способы перевода (трансляции) программ [6].

*

Информативность предлагаемого курса выше определенного школьной программой и реализуемого в большинстве школ. Поэтому изложение всего материала по предложенной методике требует несколько большего времени, чем отведенное для изучения информатики. Однако не следует забывать, что при такой логике изложения материала:

минимальный объем знаний, обусловленный школьной программой, может быть усвоен в более короткие сроки, чем это предусмотрено программой;

в курсе информатики рассматривается часть задач, обычно решаемых в курсе математики;

появляется возможность наполнить уроки информатики глубоким содержанием даже при безмашинном варианте работы.

Авторы могут предоставить всем интересующимся разработки уроков по описанной методике.

Список рекомендуемой литературы

Основная

1. Программа курса «Основы информатики и вычислительной техники» // Математика в школе. 1986. № 3.

2. Единая система программной документации. Государственный комитет СССР по стандартам. М., 1982.

3. *Ершов А. П., Монахов В. М.* Основы информатики и вычислительной техники. Ч. I, II. М.: Просвещение, 1986.

4. *Ершов А. П., Монахов В. М.* Основы информатики и вычислительной техники: Методическое пособие для учителей и преподавателей средних учебных заведений. Ч. I, II. М.: Просвещение, 1986.

5. *Кнут Д.* Искусство программирования для ЭВМ. В 3 т. Т. I, II, III. М.: Мир, 1977.

Дополнительная

6. *Ахо А., Ульман Дж.* Теория синтаксического анализа, перевода и компиляции. Т. I. М.: Мир, 1978.

7. *Гильберт Д., Бернайс П.* Основания математики. М.: Наука, 1982.

8. *Грис Д.* Наука программирования. М.: Мир, 1984.

9. *Дейкстра Э.* Дисциплина программирования. М.: Мир, 1977.

10. *Донован Дж.* Системное программирование. М.: Мир, 1975.

Приложение

Примерное планирование курса «Основы информатики и вычислительной техники»

Номера уроков	Содержание учебного материала
	Часть I—34 ч
1	Возникновение информатики. Ее место в системе других наук.
2	Задача информатики. Формализация задачи.
3	Понятие алгоритма. Свойства алгоритма.
4—5	Формальные системы, их необходимость. Язык блок-схем. Операция присваивания.
6	Ветвящиеся процессы.
7	Циклические процессы.
8	Решение задач.
9	Контрольная работа № 1.
10—11	Тестирование и отладка алгоритмов. Самостоятельная работа.
12—15	Логические операции. Модель. Самостоятельная работа.
16	Составные условия.
17—18	Решение задач.
19	Контрольная работа № 2.
20—23	Одномерные массивы. Поиск заданного элемента в массиве, нахождение наибольшего элемента. Вложенный цикл. Упорядочение массива по величине.
24	Самостоятельная работа.
25—26	Двумерные массивы.
27—29	Решение задач. Самостоятельная работа.
30	Вспомогательные алгоритмы.
31—32	Решение задач.
33	Контрольная работа № 3.
34	Итоговое занятие по первой части курса

Номера уроков	Содержание учебного материала
	Часть II—68 ч
1—2	Роль ЭВМ в современном обществе. ЭВМ как универсальный исполнитель алгоритмов.
3—4	История развития ЭВМ.
5—15	Устройство ЭВМ.
16	Контрольная работа № 1
17	Алфавит языка Фортран. Константы. Идентификаторы. Метки.
18	Арифметические и логические выражения.
19	Операторы END, CONTINUE, оператор присваивания.
20—21	Операторы GO TO и логический IF. Самостоятельная работа.
22	Арифметический IF.
23—24	Решение задач. Самостоятельная работа.
25—27	Оператор цикла. Решение задач. Самостоятельная работа.
28	Контрольная работа № 2.
29—30	Операторы описания.
31—34	Операторы READ, WRITE, FORMAT. Форматные коды. Самостоятельная работа.

Примерное планирование II части курса в случае использования программируемого микрокалькулятора (например, МК-56). 34 ч

Номера уроков	Содержание учебного материала
1—2	Роль ЭВМ в современном обществе. История развития ЭВМ.
3—4	Стековая память. Продвижение информации в стеке.
5—6	Знакомство с МК-56. Устройство. Клавиатура. Память.
7	Арифметические вычисления на МК. Самостоятельная работа.
8	Цикл. Составление простейших алгоритмов в режиме «АВТ».
9—10	Решение практических задач на МК: вычисление по схеме Горнера, решение систем двух линейных уравнений с двумя неизвестными методом Крамера.
11	Решение задач.
12	Контрольная работа № 1.
13—14	Общие сведения о работе МК в режиме «ПРГ». Клавиатура. Коды. Этапы вычисления по программе.
15—16	Ветвящиеся процессы. Самостоятельная работа.
17—20	Безусловный переход. Организация цикла. Самостоятельная работа.
21	Контрольная работа № 2.
22—25	Массивы. Решение задач.
26	Контрольная работа № 3.
27	Этап приближенного решения уравнений. Отделение и уточнение корней.
28—30	Решение нелинейных уравнений методами деления пополам, хорд и касательных.
31—32	Итерация. Решение нелинейных уравнений методом итерации.
33	Контрольная работа № 4.
34	Итоговое занятие. Перспективы развития ЭВМ
35—38	Запись. Признаки конца записи. Самостоятельная работа.
39—40	Контрольная работа № 3.
41—44	Операторная функция. Решение задач. Самостоятельная работа.
45—49	Подпрограмма FUNCTION. Самостоятельная работа.
50—54	Подпрограмма SUBROUTINE. Самостоятельная работа.
55—56	Контрольная работа № 4.
57—67	Стиль программирования. Документация. Оформление программ. Структура программ. Самостоятельное решение задач на компьютере.
68	Итоговое занятие. Перспективы развития ЭВМ

ЧТО МОЖЕТ ЭВМ?

Дешево и сердито

Психологи давно заметили: «На плохую память жалуются многие, на плохую голову — почти никто». Не будем обсуждать здесь, в чем причина такой «диспропорции», уясним лишь, что многим хотелось бы иметь хорошую память, но мало кто пытается улучшить способ управления ею.

Фирма «Ханиуэл» предлагает своим потребителям решение обеих проблем сразу. Система VLDS (Very Large Data Store) позволяет на обычной видеокассете VHS T-120 запоминать до 5,2 Гигабайт необходимой ее владельцу информации. Это значит, что вместо того, чтобы приобретать 30 десятидюймовых бобин с девятидорожечной магнитной лентой, или 5 тыс. 200 двусторонних пятидюймовых флоппи-дисков, или 52 пятидюймовых оптических диска WORM, вам будет достаточно купить одну стандартную видеокассету. Видеокассета обеспечит самую низкую удельную стоимость хранения информации. Накопитель VLDS обеспечивает почти безошибочное хранение записанных на ленту данных (единичная ошибка может произойти с вероятностью всего 10^{-12}) и легко подключается к наиболее распространенным компьютерам.

Г. ГАЗЕПОВ, В. РАСПОПОВ, Ю. СУЛЬДИН

Практикум по курсу основ информатики и вычислительной техники на ВЦ базовых предприятий

Еще далеко не во всех школах имеются персональные компьютеры и мини-ЭВМ. Поэтому там, где это возможно, практикум по курсу «Основы информатики и вычислительной техники» предлагается организовать на базе вычислительных центров предприятий. В статье обсуждается возможный вариант программы практикума для X класса, исходя из 68-часового объема учебной нагрузки (34 ч — теория, 34 ч — практические занятия).

Для реализации практикума по данной программе необходимо не менее 28 ч машинного времени, которые распределяются следующим образом. Для знакомства с работой стандартного программного обеспечения ЭВМ — 10 ч, для решения задач на ЭВМ средствами одного из языков программирования — 18 ч.

Целью всей системы учебно-воспитательных мероприятий, реализуемых по данной программе, является приобретение учащимися знаний, умений и навыков применения ЭВМ и ее программного обеспечения, а также языков программирования для решения учебных задач на ЭВМ.

Содержание программы реализует следующие принципы выбора учебных задач и средств программирования для изучения:

а) все задачи практикума должны выбираться из теоретической части курса информатики IX—X классов или составляться учителем на базе материала из учебника;

б) для практикума необходимо выделять типовое множество задач, содержание которых позволяет ознакомить учащихся с основными конструкциями языков программирования;

в) на теоретических занятиях должны быть глубоко раскрыты аналогии, касающиеся как структуры языков программирования, так и типов языковых средств. А именно программа на любом языке программирования есть последовательность невыполняемых операторов-описаний и выполняемых операторов-инструкций; оператор любого языка программирования представляет собой ключевое слово с информационной частью стандартного вида;

г) на теоретических занятиях следует выделять характерные для каждого языка типы объектов (величин), с которыми оперирует программа. Например, в языке программирования Фортран это простые переменные, массивы, а в языке PL/1 дополнительно введены структуры;

д) усложнение задач, включенных в практикум, должно производиться поэтапно.

При разработке программ на выбранном языке программирования рекомендуется использовать алгоритмы, изученные и описанные в IX классе средствами алгоритмического языка. Обязательным является обсуждение аналогий, касающихся типовых языковых конструкций.

С учетом 68-часового варианта утвержденной программы X класса предлагается следующее распределение часов:

а) на теоретические занятия с учащимися, согласно утвержденной программе,— 30 ч (4-часовое занятие по теме «Экскурсия на ВЦ» переносится в практикум);

б) на практические занятия с учащимися — 38 ч.

Как составление программ решения учебных задач практикума, так и изучение языковых конструкций следует проводить в теоретической части курса X класса, при изучении раздела «Знакомство с программированием». При этом новые языковые средства рекомендуется вводить по мере необходимости в процессе составления программ.

При подготовке к проведению теоретических и практических занятий, связанных с изучением выбранного язы-

ка программирования, учителя могут использовать рекомендуемую литературу и консультации со специалистами базовых вычислительных центров.

В качестве примера реализации данного подхода рассмотрим программу практикума, базирующегося на изучении языка программирования Фортран для ЕС ЭВМ и СМ ЭВМ. Аналогичным образом составляется программа изучения языка PL/1 и других языков программирования, входящих в состав операционной системы, эксплуатируемой на ВЦ базового предприятия.

Программа
вычислительного практикума на ЭВМ
(язык программирования Фортран)

Знакомство с вычислительной системой (16 ч) (соответствует разделу «Принципы устройства и работы ЭВМ» теоретической части курса X класса)

1. Знакомство учащихся с вычислительной системой Функциональный состав ЭВМ. Знакомство с ВЦ базового предприятия. Правила техники безопасности и внутреннего распорядка работы ВЦ — 2 ч

2. Понятие вычислительной системы как совокупности аппаратных средств и программного обеспечения. Операционная система: назначение и функции ОС. Минимальные языковые средства ОС, необходимые для реализации на ЭВМ программы пользователя, написанной на языке Фортран. Примеры составления задания для ОС по выполнению простейших программ — 4 ч

3. Работа пользователя с вычислительной системой на базовом ВЦ. Ввод заданий для ОС по выполнению простейших программ — 4 ч работы на ЭВМ

4. Диалоговый режим взаимодействия человека с вычислительной системой. Ввод и запуск прикладных программ, взаимодействие с игровыми программами, информационно-поисковыми системами, вывод информации на печать и т. п. — 6 ч работы на ЭВМ

Программирование на языке Фортран (18 ч) (соответствует разделу «Знакомство с программированием» теоретической части курса X класса)

Разработка и отладка программ линейного типа

5. Простейшие программы с использованием оператора присваивания и операторов ввода-вывода для обработки величин целого типа (описание простых переменных, оператор INTEGER, операторы READ, WRITE PRINT, оператор FORMAT со спецификацией, использование текстовых констант в операторе FORMAT, оператор присваивания «-», операторы STOP, END). Вариант задачи для разработки программы и отладки ее на ЭВМ: смотри раздел 1, § 1, п. 2 упр. 1а [1] — 2 ч работы на ЭВМ

6. Простейшие программы для обработки величин вещественного типа с элементами организации вывода (операторы REAL, FORMAT, со спецификациями F — для ввода данных, E — для вывода результата, X и /). Вариант задачи для разработки программы и отладки ее на ЭВМ: раздел 1, § 1, п. 2, упр. 1б [1] — 2 ч работы на ЭВМ

7. Использование подпрограмм и их организация (операторы CALL, SUBROUTINE, RETURN, — 2 ч работы на ЭВМ

библиотеки стандартных подпрограмм языка Фортран для вычисления значений элементарных функций $|x|$, $\sqrt{x}$, $\sin x$, $\cos x$ и т. п. Вариант задачи для разработки программы и отладки ее на ЭВМ: вычисление по формуле Герона площади треугольника, заданного на плоскости координатами своих вершин. Вычисление длины стороны организовать как подпрограмму.

Разработка и отладка программ с разветвлениями

8. Простейшие программы разветвляющейся структуры (понятие о логических величинах, отношения .EQ., .NE., .GT., .GE., .LT., .LE., логическое выражение, логический оператор IF) — 2 ч работы на ЭВМ

Оператор безусловного перехода GO TO. Вариант задачи для разработки программы и отладки ее на ЭВМ: решить квадратное уравнение. (Библиотека алгоритмов. [1]).

Разработка и отладка программ с циклами (повторениями)

9. Организация циклических программ при помощи оператора IF. — 2 ч работы на ЭВМ

Вариант задачи для разработки программы и отладки ее на ЭВМ: нахождение наибольшего общего делителя по алгоритму вклида (см. Библиотеку алгоритмов. [1]).

10. Использование оператора DO для организации циклических программ (операторы DO, CONTINUE). Вариант задачи для разработки программы и отладки ее на ЭВМ: вычислитель факториал натурального числа (см. раздел 1, § 3, п. 10, упр. 8 [1]; раздел 2, § 8, пример 5 [1]) — 2 ч работы на ЭВМ

11. Организация работы с массивами (табличными величинами): описание табличных величин, ввод, вывод и обработка табличных величин (операторы DIMENSION, READ, WRITE, PRINT). — 2 ч работы на ЭВМ

Вариант задачи для разработки программы и отладки ее на ЭВМ: подсчитать и отпечатать количество отрицательных, нулевых и положительных элементов одномерного двадцатиэлементного массива, значения которых до их обработки вводятся в ЭВМ

12. Разработка и отладка программ для решения задач из курса математики, физики. — 4 ч работы на ЭВМ

Вариант задачи для разработки программы и отладки ее на ЭВМ: вычислить и отпечатать значение площади криволинейной трапеции (см. Библиотеку алгоритмов. Алгоритм А10 [1]).

Роль ЭВМ в современном обществе. Перспективы развития вычислительной техники

13. Знакомство учащихся с содержанием задач, решаемых на ВЦ базового предприятия, режим работы пользователей ЭВМ. Характер труда специалистов, связанных с программированием и эксплуатацией, ремонтом ЭВМ. Перспективы развития вычислительного центра базового предприятия, вычислительной техники в связи с появлением персональных компьютеров и — 4 ч

супер-ЭВМ пятого поколения, сетей ЭВМ_____

Итого: 38 ч, из них необходимо машинного времени не менее 26 ч

Литература

1. Основы информатики и вычислительной техники:
Пробное учебное пособие для средних учебных заведений / Под ред. А. П. Ершова, В. М. Монахова. 4. 1, 2. М.: Просвещение, 1985, 1986.

2. *Первин Ю. А.* Основы Фортрана. М.: Наука, 1972.

3. *Салтыков А. И., Семашко Г. Л.* Программирование для всех. М.: Наука, 1986.

4. *Дрейфус М., Ганглоф К.* Практика программирования на Фортране. М.: Мир, 1978.

5. *Ламуатье Ж.-П.* Упражнения по программированию на Фортране IV. М.: Мир, 1978.

6. *Светозарова Г. И., Сигатов Е. В., Козловский А. В.* Практикум по программированию на алгоритмических языках. М.: Наука, 1980.

7. *Брич З. С., Капилевич Д. В., Котик С. Ю., Цагельский В. И.* Фортран ЕС. М.: Статистика, 1978.

8. Фортран / Под ред. Е. Л. Ющенко, Киев: Вища школа, 1980.

ПОСЛЕ КОНКУРСА

В. ФЕДОТОВ

Из рукописи конкурсного учебника

Первые вычислительные машины

Попытки создать механическую вычислительную машину предпринимались еще в средние века. Проект одной из таких машин принадлежит Леонардо да Винчи (1452—1519) — выдающемуся художнику и мыслителю эпохи итальянского Возрождения.

Первая счетная машина, о которой сохранились сведения, была построена в 1623 г. немцем Вильгельмом Шиккардом. Несколько больше известно о машине выдающегося французского математика Блеза Паскаля (1623—1662). Португалец Х. Р. Перейра в 1751 г. изготовил машину для обучения счету глухонемых.

Разнообразные клавишные счетные машины и арифмометры неоднократно изобретались и в XIX в. Например, глава петербургской математической школы П. Л. Чебышев (1821—1894) в 1878 г. собственноручно сконструировал арифмометр.

Решающий вклад в развитие вычислительной техники сделал английский математик Чарльз Бэббидж (1792—1871). Он впервые предложил устройство, которое выполняло бы действия автоматически, по заранее подготовленной человеком программе. В 1822 г. Бэббидж изготовил простейшую машину, способную вычислять значения некоторых элементарных функций.

В 1834 г. он изобрел аналитическую машину — прототип современных цифровых вычислительных машин. Над ее созданием Бэббидж работал около двадцати лет, сделав при этом более 200 чертежей различных узлов и изготовив многие устройства, но он так и не завершил свой труд.

Большую помощь Бэббиджу оказывала его ученица Ада Августа Лавлейс — дочь известного английского поэта Байрона. Если сам Бэббидж был занят в основном конструированием и изготовлением механических деталей машины, то леди Лавлейс по праву считается первой в истории программисткой. Она не только проводила вычисления на машине Бэббиджа, но и заложила основы теоретического программирования, написав первый учебник по этому предмету. Ей принадлежит изобретение оператора условного перехода, именно она ввела понятия рабочей ячейки и цикла.

Однако работа Бэббиджа и Лавлейс не получила непосредственного продолжения. Непрерывная история создания вычислительной техники начинается с имени американского инженера Германа Холлерита (1860—1927). Для обработки статистических данных он изобрел перфокарту, формат которой считается наиболее удобным до сих

Окончание. Начало см.: Информатика и образование. 1989. № 1.

пор. В 1884 г. Холлериту был выдан первый патент на машину для переписи населения. Позднее им были получены более 30 патентов на перфораторы, табуляторы и сортировальные машины.

Клавишный перфоратор Холлерита мог пробивать около 100 отверстий в минуту. Другой специальный перфоратор заносил повторяющуюся информацию сразу на несколько перфокарт. Сортировальная машина представляла собой несколько ящиков, каждый со своей крышкой. Карты продвигались между набором штырей, насаженных на пружины, и резервуаром со ртутью. Как только штырь попадал в отверстие, он касался ртути и замыкал электрическую цепь. При этом поднималась крышка нужного ящика, куда и попадала карта. Аналогично работал и табулятор, при этом электрический сигнал подавался на соответствующий счетчик, увеличивая его значение на единицу.

Система счетно-аналитических машин Холлерита позволила обработать результаты переписи населения США в 1890 г. точнее и в 3 раза быстрее, чем в 1880 г. Предприятие Холлерита со временем выросло в известную международную фирму IBM.

Электрические машины

Начало XX в. — время бурного развития электрической техники. Этому предшествовало интенсивное исследование электрических явлений и открытие в XIX в. их связи с магнитными. В конце XIX в. А. А. Яблочков впервые использовал электричество для освещения и создал обычную теперь электрическую лампочку. Другой выдающийся русский ученый — А. С. Попов изобрел в 1905 г. радио. Позднее принципы радиосвязи были использованы и в телевидении.

В 20-х гг. американец Ванневар Буш создал дифференциальный анализатор — первую аналоговую вычислительную машину. Сравнительно несложная электрическая схема позволила моделировать системы дифференциальных уравнений. Решение таких систем вручную требует огромного количества вспомогательных вычислений и дает только приближенный результат, тогда как применение дифференциального анализатора давало результат мгновенно и с более высокой точностью. Однако машина Буша не имела памяти: данные (напряжения, сопротивления, емкости и т. п.) вводились в нее непосредственно, как параметры электрической цепи, а результаты снимались электроизмерительными приборами.

Параллельно шло развитие цифровой техники. Первые машины, основанные на двоичной системе счисления, независимо друг от друга создали в начале 30-х гг. американский математик Джордж Штибитц и немецкий инженер Конрад Зюс. По своему устройству их машины были релейно-механическими, т. е. основанными на идеях Холлерита.

Графический экран*

В § 10 главы 3 вы познакомились с правилами изображения на экране дисплея текстовых строк. В частности, вы знаете различия между логической и физической строкой. Любую текстовую строку, записанную в тексте программы или вычисленную в процессе исполнения программы, можно вывести на экран дисплея, при этом если длина строки велика, то одна логическая строка займет несколько физических строк экрана.

Сходным образом единственному физическому экрану дисплея могут соответствовать несколько логических экранов. Каждый экран имеет свой порядковый номер и название. Количество логических экранов в разных машинах может быть различным. Мы будем говорить о двух: текстовом и графическом, их номера соответственно 1 и 2.

Для установки нужного экрана используется специальная команда ЭКРАН. Непосредственно после этого служебного слова указывается номер или название логического экрана, после которого может присутствовать дополнительная информация (о ней речь пойдет ниже). В одних машинах исполнение этой команды автоматически очищает экран, в других содержимое логического экрана при его смене другим запоминается и вновь восстанавливается при очередном вызове этого экрана, в этом случае для очистки применяется специальное служебное слово ОЧИСТИТЬ. Оно может входить в состав команды ЭКРАН либо быть самостоятельной командой. В режиме непосредственного диалога текстовый экран можно очистить, нажав на соответствующую клавишу.

Дисплей может быть двухцветным (черно-белым или черно-зеленым) или многоцветным. Независимо от этого вся выводимая на дисплей информация считается многоцветной и соответствующим образом кодируется. Разумеется, двухцветный экран не позволяет четко различить некоторые цвета. Здесь можно провести аналогию с телевидением, где черно-белый телевизор может принимать программу цветного телевидения,

---

* Звездочка означает, что содержание параграфа публикуется с сокращениями.— *Прим. ред.*

но изображение на его экране будет черно-белым.

Количество используемых цветов и их нумерация в разных машинах и отвечающих им версиях алгоритмических языков могут значительно отличаться друг от друга. Наиболее распространены заимствованные из цветного телевидения системы составления любого нужного цвета наложением трех основных цветов разной интенсивности. В качестве основных цветов могут выступать красный, зеленый и голубой, либо синий, желтый и красный, либо три оттенка, не имеющие в естественных языках точных названий, но особенно удобные для представления их комбинациями всех остальных цветов.

Нумерация цветов, как правило, соответствует разложению этих цветов на выбранные основные. Основным цветам при этом приписываются номера, равные степеням двойки, а составным — равные сумме номеров тех основных цветов, из которых этот цвет составляется.

Например, если желтому цвету приписан номер 1, синему — 2, а красному — 4, то зеленый цвет, полученный наложением желтого и синего, имеет номер $3=2+1$, фиолетовый — $6=4+2$ — наложение красного и синего, оранжевый — $5=4+1$ — наложение красного и желтого, черный — 0 — отсутствие всех основных цветов и белый — $7=4+2+1$ — наложение всех.

Описанная система включает $8=2^3$ цветов. В некоторых персональных ЭВМ используются системы, включающие до 256 цветов. Интересная 16-цветная система используется в японских ПЭВМ «Ямаха».

Команда ЦВЕТ или указание цвета в команде ЭКРАН позволяют изменить цвет текстового или графического экрана; причем одновременно могут быть изменены цвет основного фона, цвет изображения на этом фоне и цвет бордюра, ограничивающего фоновую зону текстового экрана. Еще одна особенность команды ЦВЕТ состоит в том, что в разных машинах она может относиться либо к физическому экрану, т. е. сразу ко всем логическим экранам одновременно, либо только к тому логическому экрану, который в данный момент вызван на дисплей.

Если на текстовом экране могут одновременно присутствовать только три различных цвета, то разные точки графического экрана могут быть независимо друг от друга закрашены в любые цвета. Для перекраски отдельной точки можно использовать команду ТОЧКА, аргументами которой служат координаты выбранной точки (обычно они заключаются в скобки) и цвет, в который перекрашивается выбранная точка. Если цвет не указан, то точка перекрашивается в последний ранее объявленный цвет изображения.

Особо следует сказать о системе координат графического экрана. Начало координат физического экрана размещается в его левом верхнем углу. Ось абсцисс совпадает с верхней границей физического экрана и направлена, как обычно, слева направо. А вот направление оси ординат непривычно, сверху вниз, и совпадает эта ось с левой границей экрана. Размеры экрана: 256 точек по горизонтали и 192 точки по вертикали, т. е. левый нижний угол имеет координаты (0; 191), правый верхний — (255; 0), а правый нижний — (255; 191).

Итак, абсцисса точки физического экрана является целым числом в пределах от 0 до 255, а ордината — от 0 до 191, причем последняя измеряется сверху вниз. Координаты же точек логического экрана обычно могут изменяться в значительно более широком диапазоне, например от —32767 до +32768. Это означает, что вы можете по своему усмотрению назначать цвета точек, находящихся за пределами физического экрана.

Файлы*

Сравнительно короткие тексты (до 250 символов) могут быть записаны в пределах программной строки. Для записи значительно более длинных текстов обычно организуются файлы. Файл представляет собой массив, как правило, однородной информации, при необходимости разбитый на более мелкие строки или записи произвольной, одной и той же или переменной длины, а иногда на блоки записей.

Файл может включать не только тексты в обычном смысле, но и числовые данные или тексты программ. Файл может быть записан на внешнем носителе (магнитной ленте или диске) либо размещен в буферном запоминающем устройстве. Как правило, размеры файла слишком велики для того, чтобы можно было держать его в оперативной памяти, и туда вызываются поочередно только отдельные записи или блоки, составляющие файл.

Каждому файлу присваивается имя. Имя файла подчиняется общим правилам составления имен, но оно должно быть сравнительно коротким. При работе программы с несколькими внешними устройствами во избежание путаницы имя файла обычно указывается вместе с именем того устройства или магнитного носителя, на котором находится нужный файл (можно сказать,

что к файлу обращаются по «имени», «отчеству» и «фамилии»).

Так как ошибки в работе с файлами приводят, как правило, к серьезным последствиям, то перед началом и в конце работы с файлом выполняются определенные операции, цель которых — уменьшить вероятность ошибок. Так, прежде чем выполнить какую-либо операцию с файлом, его нужно ОТКРЫТЬ соответствующим оператором, в котором указывается имя одного или нескольких открываемых файлов. Пока файл не открыт, попытка выполнить любую другую операцию с этим файлом приведет к сообщению об ошибке. Некоторые языки программирования требуют также точного указания в операторе ОТКРЫТЬ, с какой целью открывается файл — ДЛЯ ЗАПИСИ или ДЛЯ ЧТЕНИЯ, и последующая попытка выполнить с этим файлом иную операцию также считается ошибкой.

Передача по сети

В многопроцессорных вычислительных комплексах, комплектах учебной вычислительной техники и сетях ЭВМ более общего назначения возможна передача файлов с одного терминала на другой. В каждой конкретной сети имеются свои особенности и отличия в организации такой передачи.

В одних случаях передача информации возможна только с центрального терминала на остальные (например, с учительского на ученические в некоторых КУВТ) и обратно, в других — все терминалы связаны между собой попарно. Есть различия и в организации самой передачи.

Наиболее распространена передача по взаимному соглашению. Для передачи программы или файла данных с одного терминала на другой на принимающем терминале должна быть выбрана инструкция ПРИНЯТЬ (или ЗАГРУЗИТЬ, или ЗАПУСТИТЬ, или ПРИСОЕДИНИТЬ), а на передающем ПЕРЕДАТЬ (или СОХРАНИТЬ). Вместо имени файла или в его составе должно быть указано имя другого терминала, участвующего в этой передаче, или общее имя сети. В последнем случае возможна передача информации с одного терминала сразу на все, заказавшие непосредственно перед этим прием файла.

Возможна организация передачи по принципу подчинения терминалов. Например, с машины учителя можно запросить и принять любую информацию с любого ученического терминала, а также передать информацию, даже если ученик не только не предпринимает для этого никаких действий, но и ничего не знает о действиях учителя.

Такая организация сети позволяет продемонстрировать элементы параллельного программирования. Если программа требует для своей реализации большого объема оперативной памяти либо включает ряд программ, которые могут исполняться независимо друг от друга, то можно передать эти подпрограммы и части основной программы с учительской машины на ученические и организовать параллельные вычисления на ученических машинах.

Спрайты

Слово «спрайт» в переводе с английского означает «светлячок». Светлячки могут выводиться на графический экран и перемещаться. Это заметно обогащает изобразительные возможности ЭВМ, позволяет выделять нужную информацию на экране, программировать мультфильмы и т. д.

Спрайт обычно занимает небольшую часть экрана — квадрат 8×8 или 16×16 точек. Каждая точка этого квадрата либо прозрачна, либо закрашивается в определенный цвет. При кодировании спрайта закрашенным точкам соответствуют единицы, а прозрачным — нули. Это позволяет сопоставить изображению спрайта двоичное число. Так как это число слишком длинное — в нем 64 или 256 цифр, то для сокращения его записи каждая группа из 8 цифр, отвечающая строке или полустроке спрайта, заменяется символом, кодом которого является это восьмизначное двоичное число. Получается, что спрайт кодируется не длинным двоичным числом, а текстовой строкой из 8 или 32 символов.

Для облегчения кодирования спрайта используется функция ЗНАК, аргументом которой является число, а значением — текстовая строка, состоящая из одного символа, код которого равен числу — аргументу этой функции. Чтобы закодировать спрайт, вы можете ввести список нулей и единиц, отвечающих точкам квадрата этого спрайта, затем сгруппировать их по 8 и вычислять по схеме Горнера численные значения кодов строк или полустрок спрайта. Пользуясь функцией ЗНАК, вы можете составить текстовую строку, отвечающую этому спрайту. Когда спрайт сформирован, ему присваивается определенный номер, для чего используется оператор:

СПРАЙТ (НОМЕР):=СТРОКА

в котором СПРАЙТ — служебное слово (название оператора), НОМЕР — число или имя числовой величины (номер спрайта), а СТРОКА — текстовая строка или ее имя (символьный код спрайта).

Для вывода спрайта на графический экран

используется оператор ВЫВЕСТИ СПРАЙТ, в котором указываются номер спрайта, координаты точки экрана, в которой будет размещен левый верхний угол спрайта и номер плоскости. Номера плоскостей указывают на приоритет спрайтов при их наложении друг на друга: спрайт в плоскости с большим номером будет выглядеть находящимся впереди спрайта в плоскости с меньшим номером.

Текстовый редактор*

Для подготовки сравнительно длинных текстов удобно пользоваться различными программами для их редактирования. В принципе, составление программы текстового редактора вполне доступно и для вас самих.

Текст оформляется в виде файла, который затем может быть записан на магнитный диск, передан на другой терминал или печатающее устройство. Работа программы начинается с того, что с клавиатуры вводятся имя этого файла и ограничения для числа символов в каждой его строке, числа строк в каждой странице и числа страниц в самом файле. Выделяется место для файла на буферном устройстве и проверяется возможность последующей записи файла на магнитный диск, сообщение о чем выдается на экран дисплея.

Допускается использование имени одного из файлов, уже записанных на диск. Это будет означать, что файл не новый, т. е. вы будете редактировать (пополнять, изменять и т. д.) уже имеющийся текст. В этом случае содержимое нужного файла переписывается на буфер, как только вы подтвердите в ответ на запрос программы, что совпадение имен не является ошибкой.

Каждый введенный с клавиатуры символ, как и обычно, появляется на экране в позиции курсора и одновременно заносится в соответствующий байт буфера. Размеры экрана слишком малы для того, чтобы обозреть содержимое не только всего файла, но даже одной страницы текста. Поэтому экран используется как окно, позволяющее увидеть только часть какой-либо страницы, либо стык двух страниц. Это реализуется по-разному в текстовых редакторах и связано с клавишами управления движением курсора.

Прежде всего заметим, что во всех текстовых редакторах ведется учет текущей позиции курсора в файле, т. е. номер байта в строке, номера строки на странице и номера страницы файла, отвечающих положению курсора на экране. С вводом каждого очередного символа номер байта увеличивается на единицу до тех пор, пока не достигнет ограничения на число символов в строке. В этот момент производится переход на следующую строку. В частности, номер строки увеличивается на единицу, а номеру байта присваивается начальное значение (0 или 1). Аналогично происходит и переход на следующую страницу.

Основная сложность состоит в том, что строка текста, например предназначенного для последующей печати на стандартном листе бумаги, как правило, длиннее физической строки экрана. Способы преодоления этого затруднения и составляют основное отличие в работе различных текстовых редакторов.

Графический редактор

Операторы проведения линий на графическом экране, с которыми вы познакомились в § 1 главы 6, сравнительно сложны для использования их непрофессиональными программистами. Трудностей две — и в правилах составления самих операторов, и в расположении системы координат графического экрана. Специальная программа — графический редактор — позволяет обойти обе эти трудности и делает графический экран доступным для всех.

Работа программы начинается с появления на экране инструкции, объясняющей, что произойдет на экране при нажатии на ту или иную клавишу. В частности, сама эта подсказка может быть вновь в любой момент вызвана на экран нажатием на одну из функциональных клавиш.

После просмотра подсказки вы можете восстановить прежний рисунок, нажав на другую функциональную клавишу. На нее же нужно нажать и для начала работы. Экран при этом перекрашивается в белый цвет, что изображает чистый лист бумаги. В центре экрана вы видите курсор, имеющий форму небольшого крестика.

Еще одна из функциональных клавиш позволяет циклически изменять цвет этого крестика, играющего роль нашего карандаша. Стрелки управления курсором позволяют перевести крестик в новое положение, сохранив исходное. Затем, нажав на одну из буквенных клавиш, вы можете соединить старое и новое положение курсора отрезком или ломаной, провести окружность с центром в старом и проходящую через новое положение курсора, закрасить внутренность прямоугольника или круга. Можно просто снять предыдущее положение курсора, начав новое построение от его текущего положения, либо поменять ролями старое и новое положения курсора.

Все построения выполняются текущим цветом крестика-курсора. При его изменении цвет ранее проведенных линий не меняется. Это позволяет сделать многоцветный рисунок.

Одна из клавиш позволяет чертить на экране или убирать с него ось симметрии. При наличии оси каждое построение симметрично повторяется во второй половине экрана. Подчеркнем, что симметричные части чертежа сохраняются и при удалении оси симметрии, а несимметричные с ее появлением не дублируются. Это позволяет делать чертежи, содержащие как симметричные, так и несимметричные детали.

Нажатие на одну из управляющих клавиш переводит графический редактор в режим ввода символов. Курсор при этом из крестика превращается в обычный квадратик, но, для того чтобы белый курсор был виден на белом же фоне листа, он ограничивается черным контуром. При этом нажатие на символьные клавиши будет приводить к появлению соответствующих букв и цифр в позиции курсора до тех пор, пока вы не вернете редактор в основной режим его работы.

Когда чертеж закончен и сделаны все необходимые надписи, вы можете скопировать его в файл на магнитном диске или перенести на бумагу, используя печатающее устройство.

Редактор спрайтов

Спрайты (§ 6 главы 6) еще более трудны для непосредственного программирования, чем графические изображения. Для их редактирования имеется специальная программа.

Как и графический редактор, эта программа начинается с заставки с инструкцией. Вам предлагается выбрать один из нескольких режимов редактирования спрайтов.

В режиме подготовки спрайтов вы видите на экране два изображения спрайта: одно в натуральную величину и другое — значительно увеличенное, которое выглядит как шахматная доска. На одной из клеток этой доски стоит курсор. Стрелки управления курсором позволяют произвольно перемещать его в пределах доски. Если вы нажмете на клавишу «ВВОД», то в позиции курсора появляется крестик, а в изображении спрайта в натуральную величину загорается точка. Для удаления неудачно поставленной точки нужно накрыть курсором отвечающий ей крестик и нажать длинную клавишу ПРОБЕЛ.

Подготовив подобным образом изображения одного или нескольких спрайтов, вы можете перейти к любому из последующих этапов их редактирования. Редактор позволяет имитировать движение спрайтов, их столкновения и наложение друг на друга. Последовательное помещение в определенную область экрана то одного, то другого спрайта создает эффект мультипликации.

Закончив редактирование спрайтов, вы можете записать в файл на магнитном диске их символьные коды или соответствующие им фрагменты программ на алгоритмическом языке.

Музыкальный редактор

Если к вашему терминалу подключен громкоговоритель или наушники, то вы можете познакомиться и с программой, позволяющей редактировать звуковую информацию. Познакомившись с инструкцией, вы можете перейти к любому из режимов редактирования.

В одном из них вы видите на экране изображение клавиш одной октавы и условный номер октавы. Стрелки горизонтального перемещения курсора позволяют вам установить курсор на любой из нотных клавиш, а стрелки вертикального перемещения курсора меняют номер октавы на более высокий или низкий, в зависимости от направления стрелки. Нажав клавишу ВВОД, вы заносите выбранную ноту в очередную позицию записи музыкального текста.

Каждой ноте присваивается определенный номер — сравнительно небольшое целое число. Этот номер кодируется единственным байтом. Однако чаще его записывают не одним символом и не восемью двоичными цифрами (как, например, строку спрайта), а двумя шестнадцатеричными цифрами, первая из которых соответствует номеру октавы, а вторая — номеру звука в пределах октавы (вместе с полутонами октава включает 16 различных звуков). Итак, любой музыкальный текст представляется в машине как последовательность шестнадцатеричных цифр либо как последовательность байтовых символов (каждый из которых заменяет пару шестнадцатеричных цифр). Один из режимов редактирования позволяет увидеть ранее набранный текст именно в таком представлении и вносить любые изменения непосредственно в эту последовательность цифр.

Другой режим редактирования позволяет увидеть диаграммы высоты и силы звучания как функции от времени по временным промежуткам разной длительности. С помощью нескольких шаблонов вы можете выбрать линейный, пилообразный, синусо-

идальный или какой-либо иной закон изменения этих функций, что приводит к различным звуковым эффектам.

Наконец, в любом режиме редактирования вы можете прослушивать выбираемый звук, формируемый фрагмент музыкального текста или весь текст. По окончании сочинения музыки вы можете записать в файл на магнитном диске ее текст в шестнадцатеричном или символьном коде либо соответствующую последовательность команд алгоритмического языка.

Построение графика функции*

В программном обеспечении вашей ЭВМ наверняка есть какие-либо программы, позволяющие построить график функции. Полезно разобраться, как устроена такая программа, чтобы не допускать ошибок в ее использовании.

Рассмотрим задачу о построении графика функции на экране дисплея так, чтобы этот график был изображен как можно более крупно. При исследовании функциональных зависимостей в реальных задачах аргумент и значение функции обычно имеют различную природу и измеряются в разных единицах. Это позволяет независимо друг от друга выбирать и изменять масштабы на координатных осях. Поэтому наиболее подробным изображение графика будет в том случае, когда границы отрезка, на котором задана функция или строится ее график, совпадают с границами экрана по горизонтали, а максимум и минимум функции — с границами экрана по вертикали.

Из курса алгебры и начал анализа вы знаете способы построения графика, основанные на предварительном исследовании функции с помощью производных. Здесь мы воспользуемся более грубым методом построения — по точкам.

Через A и B обозначим наименьшее и наибольшее значения аргумента X, т. е. границы рассматриваемой области определения функции. Им отвечают 256 вертикальных точек экрана, пронумерованных от 0 до 255.

Границе A будет отвечать вертикаль экрана с номером 0, границе B — вертикаль 255, а в качестве промежуточных значений аргумента возьмем

X=(K·B+(255—K)·A)/255, где K=0, 1, 2, 3, ..., 254, 255. Заметьте, что K=0 означает X=A, а K=255 отвечает X=B.

Для этих значений аргумента не нужно составлять специальный список, так как они легко выражаются через K, которое может служить параметром цикла от 0 до 255.

Универсальная, т. е. пригодная для любой функции, программа построения графика должна обращаться к вычислению значений функции Y(X) как ко вспомогательному алгоритму — процедуре или подпрограмме.

Для поиска наибольшего и наименьшего значений из списка величин имеется немало различных алгоритмов. Можно воспользоваться любым из них, в том числе использовать в качестве процедуры или подпрограммы любой стандартный алгоритм из имеющегося программного обеспечения. В нашем случае лучше взять не самый быстрый, но весьма простой и удобный способ: вести последовательный учет наибольшего и наименьшего из ранее вычисленных значений одновременно с вычислением Y(X). Самое первое значение Y(A) объявляется одновременно наибольшим и наименьшим, а каждое последующее сравнивается с этими границами и записывается вместо той из них, за пределы которой оно выйдет.

Следующий этап программы — вычисление номеров горизонталей (вертикальных координат), отвечающих положениям контрольных точек на экране. В первом случае (когда экономится память) этот этап совмещается с последним: вслед за вычислением координат каждой точки она наносится на экран и соединяется отрезком с предыдущей точкой графика. Во втором случае сначала накапливается список номеров горизонталей, а затем сразу строится весь график. Номера горизонталей можно записывать и в тот же самый список, в котором ранее хранились значения функции, но лучше в другой, так как значения функций — вещественные числа, а номера горизонталей — целые.

Не забудьте, что горизонтали экрана нумеруются сверху вниз. Поэтому максимальное из контрольных значений функции будет лежать на нулевой горизонтали, а минимальному отвечает горизонталь с самым большим номером — 191. Произвольное значение функции Y будет лежать на горизонтали, номер которой равен 191(МАКС—Y)/(МАКС—МИН), где МАКС и МИН обозначают, соответственно, наибольшее и наименьшее из контрольных значений функции.

Итак, график построен. Он представляет собой ломаную, проходящую через контрольные точки. Так как шаг по горизонтали равен ширине элементарного точечного изображения на экране, то для «хороших» функций эта ломаная будет выглядеть как гладкая линия.

Для того чтобы график нес больше информации, целесообразно напечатать на экране максимальное и минимальное значения функции и аргумента. По возможности это нужно сделать на свободной от точек графика части экрана.

Для большей выразительности можно вывести на экран и оси координат. Для этого нужно подставить нулевые значения в формулы для вычисления отвечающих им номеров вертикали и горизонтали, а затем провести соответствующие отрезки от верхнего края экрана к нижнему и от правого к левому.

Программу построения графика следует оформить в виде достаточно замкнутого блока — процедуры или подпрограммы. Это позволит вам многократно пользоваться программой для построения графиков различных функций.

Несколько видоизменив программу, вы можете выводить на экран графики сразу двух или более функций, что даст возможность сравнивать их и графически решать уравнения.

Графический микроскоп*

Составленную в предыдущем параграфе программу построения графика функции удобно использовать для графического решения различных задач, связанных с исследованием функций, например для нахождения корней, точек экстремума и других характерных точек на графиках функций. Глядя на «хороший» график, эти точки легко увидеть. Зная значения аргумента, отвечающие границам отрезка области определения, для которого построен график, вы можете вычислить границы более короткого отрезка, содержащего интересующую вас точку, ввести их в текст программы в качестве A и B, после чего та же программа построит вам более подробный график вблизи интересующей вас точки.

А можно ли обойтись без грубых ручных вычислений (перед экраном дисплея ЭВМ!) и без поиска в тексте программы мест, куда нужно внести значения A и B? Разумеется, все это возможно, но потребует составления еще одной подпрограммы.

Пользуясь редактором спрайтов (§ 6, глава 7), вы можете сформировать и вывести на экран спрайт в форме крестика. Следующая наша цель — заставить этот спрайт двигаться, подчиняясь стрелкам управления курсором[1].

Если вы все сделаете правильно, то работа программы будет организована следующим образом. Сначала вычисляются контрольные точки и появляется на экране изображение исходного графика. Затем появляется крестик-курсор, перемещая который по экрану вы выбираете положение начала нового отрезка. Возможность перемещать курсор по вертикали позволяет совместить его непосредственно с точкой графика, однако при вводе учитывается только его абсцисса. После ввода первого крестика появляется второй, отвечающий второму концу отрезка. После ввода второго крестика программа зацикливается — на экране появляется второй график, затем вы вводите новые границы, появляется третий график, вводите очередные границы, появляется четвертый, и т. д.

Такой режим работы программы позволяет очень быстро и с высокой точностью находить корни и точки экстремумов непрерывных функций. Вы можете, глядя на график, подвести крестики почти вплотную к интересующей вас точке. Это позволит сократить длину рассматриваемого отрезка сразу в 40—130 раз. Когда же этот отрезок станет достаточно мал, общие первые цифры координат его начала и конца будут служить и цифрами абсцисс выбранной точки. При удачном управлении крестиками вы будете при каждом новом пересчете увеличивать число известных цифр абсциссы выбранной точки сразу на две.

При достижении нужной точности результата следует вывести программу из цикла. Для этого достаточно нажать на клавишу ПРЕРЫВАНИЕ.

Вы можете предусмотреть и выход из зацикливания по нажатию на одну из функциональных или символьных клавиш. Использование функциональных клавиш предоставляет и ряд других возможностей для усовершенствования программы. Например, вы можете запоминать все предыдущие значения границ отрезка и при нажатии на одну из клавиш вернуться к последней паре границ. Это дает две возможности. Во-первых, легко устраняются ошибки, связанные с ошибочными нажатиями при исполнении программы. Во-вторых, появляется возможность, не прерывая исполнение программы, последовательно и с высокой точностью найти несколько корней и/или точек экстремума исследуемой функции.

Синусоида издали и вблизи

Испытаем работу графического микроскопа на примере хорошо известной из курса алгебры и начал анализа функции — синуса. Прежде всего проверим правильность составления программы и ввода формулы для функции. С этой целью возьмем A=0 и B=10. На экране вы должны увидеть хорошо знакомую картину — чуть больше полутора периодов синусоиды: одну полную волну и верхнюю полуволну.

Теперь мы можем исследовать, как будет

[1] Далее в тексте следует описание программы «Графический микроскоп».— *Прим. ред.*

меняться вид графика при изменении границ отрезка. Заметим, что если длина отрезка превосходит период синуса, то верхний и нижний края экрана будут отвечать значениям ± 1, а потому изменения границ будут приводить, как правило, только к растяжению или сжатию графика по горизонтали, не меняя его вертикальный масштаб. Все отклонения от этого правила мы будем подчеркивать особо.

Сначала зафиксируем A=0 и будем изменять B в сравнительно небольшом диапазоне — порядка нескольких периодов синуса. «Красивые» синусоиды на экране наглядно демонстрируют все происходящие с графиком изменения и подтверждают, что экран дисплея достаточно подробно информирует нас о поведении функции.

Будем теперь уменьшать B, приближая его к нулю. Вот от волны осталась только выпуклая дуга, соединяющая левый нижний угол экрана с правым верхним. Минимальное значение синуса теперь равно нулю и отвечает началу отрезка A=0, а максимальное отвечает второму концу отрезка — B и уменьшается с его уменьшением. Программа устроена таким образом, что максимальное значение остается все время на верхней границе экрана, поэтому при дальнейшем уменьшении B в диапазоне меньше четверти периода синуса будет меняться и вертикальный масштаб графика, причем почти пропорционально горизонтальному.

С уменьшением B при приближении его к нулю дуга графика, соединяющая левый нижний угол экрана с правым верхним, все выпрямляется и выпрямляется, пока, наконец, не становится практически неотличимой от отрезка. Это наглядно подтверждает известное вам одно из основных положений математического анализа: в достаточно малой окрестности точки гладкая функция практически неотличима от линейной. Именно на этом принципе основывается понятие производной и все дифференциальное исчисление.

А теперь от привычных и легко объяснимых картинок перейдем к загадочным, по крайней мере на первый взгляд. Сохранив A=0, возьмем B=255· 3.14159.

Неужели это синусоида? На экране вы видите не кривую, а целую светящуюся область в форме угла, вершина которого помещается точно в середине левой границы экрана, а лучи сторон направлены, соответственно, в правый верхний и правый нижний углы экрана. Как же получилась такая картинка? Давайте разберемся.

3. 14159 — приближенное значение числа π — полупериода синуса. Другой множитель — 255 — как раз совпадает с числом промежутков между вертикалями экрана. Поэтому в качестве 256 контрольных точек выбраны значения, кратные 3.14159.

Если бы вы строили этот график вручную, то, считая π равным 3.14159, получили бы последовательность синусов от кратных значений аргумента. Вы знаете, что все они равны нулю. Поэтому все без исключения контрольные точки лежат на прямой Y=0. Исполняя формально построение синусоиды по точкам, при таком выборе контрольных точек вы обязаны сделать вывод, что графиком является прямая Y=0. Весьма поучительный пример того, как не следует строить график!

Однако машина хотя и пользуется приближенным значением для π, но более точным, чем 3.14159. Поэтому, применив формулы приведения, вы можете убедиться, что фактически построен график синуса от ... ошибок округления! О чем же он говорит?

Во-первых, о том, что, как правило, с ростом аргумента возрастают и ошибки его округления. Действительно, печать в нижней части экрана максимума и минимума из контрольных значений функции явно указывает, что все эти значения весьма малы. Предыдущий график с малым B наглядно показал вам, что при малых значениях аргумента синус почти ему пропорционален. Более того, из начал анализа вы знаете, что синус малого аргумента почти не отличается от значения самого аргумента. Уже это позволяет сказать о том, что на экране вы видите график не синуса, а самих ошибок округления. И вы видите, что абсолютная величина этих ошибок линейно возрастает с ростом аргумента.

Во-вторых, можно увидеть, что график на экране представляет собой пилообразную ломаную, звенья которой последовательно соединяют верхнюю сторону угла с нижней, а нижнюю с верхней. В итоге эти вертикальные отрезки и заполняют всю внутренность угла, что и создает эффект светящейся области вместо кривой. Нетрудно объяснить и это. Так как ошибки округления пропорциональны разности между 3.14159 и тем значением для π, которое использует машина, то отклонения от истинного корня функции носят систематический характер. Применение формул приведения показывает, что четность числа полупериодов синуса будет чередоваться при переходе от одной контрольной точки к соседней с ней. В конечном счете это приведет к чередованию знаков у последовательных контрольных значений, что и дает такое необычное изображение на экране.

Для подтверждения последнего вывода удвойте B, т. е. возьмите B=510· 3.14159

На экране вы увидете аккуратный отрезок из левого верхнего угла в правый нижний, так как на этот раз аргументы контрольных точек кратны целому периоду синуса и применение формул приведения даст один и тот же знак для всех контрольных точек. А так как выбранное значение 3.14159 меньше π, то этот знак — минус.

Неожиданный эффект дает B=512× ×3.14159. На экране вы увидите идеальное изображение одного целого периода синусоиды. Объяснение опять дают формулы приведения. Но здесь гораздо важнее не объяснение, а предостережение: и «идеальное» изображение графика на экране может ввести в заблуждение. Ведь, глядя на экран, можно было бы сделать ошибочный вывод, будто период синуса в 256 раз больше, чем на самом деле.

Не должна вас удивить и еще одна ложная синусоида — при B=85· 3.14159. На экране вы видите хотя и изломанную, но многократно повторяющуюся волну. Однако посмотрите на значения минимума и максимума — они сравнительно далеки от единиц. Для объяснения заметим, что 85=255/3. Поэтому на экране выданы значения синусов, аргументы которых кратны π/3. Постарайтесь подобрать и свои не менее интересные примеры.

А теперь вновь обратимся к построению графика на малом отрезке, но на этот раз — вблизи точки экстремума. Для этого нужно либо отказаться от условия A=0, либо заменить синус косинусом и, сохраняя A=0, брать B близким к нулю. Пойдем по второму пути. Когда B становится меньше, чем 10^{-5}, вы начинаете замечать, что дуга становится «шероховатой». Если уменьшать B дальше, то эти шероховатости начинают выделяться в самостоятельные всплески, плавная дуга превращается в ломаную и теряет свою монотонность. Этот эффект уже невозможно объяснить ни формулами приведения, ни другими свойствами функции, график которой выведен на экран.

Не следует думать также, что наш микроскоп дал столь сильное увеличение, что стали видны эффекты, относящиеся к «атомному строению» синусоиды. Синусоида не реальное тело, а математическая абстракция.

Причина — в дискретности цифровых машин. Вспомните: число записывается в ячейке с конечным числом двоичных разрядов. Отсюда следует, что машина способна различать не континуум вещественных чисел и даже не счетное множество рациональных, а только конечное число двоичных дробей с определенным числом разрядов. Так как это конечное число очень велико, то эффекты, связанные с дискретностью машинного представления, не сразу себя обнаруживают. Однако, когда интервал значений сужается настолько, что в его пределах остается сравнительно небольшое число различных машинных чисел, именно эти эффекты начинают играть решающую роль.

Дефекты в изображении графика начинают проявляться тогда, когда область значений функции сокращается до таких размеров, что в ее пределах остается около 192 или меньше различных машинных чисел, т. е. число горизонталей экрана начинает превосходить число значений, отвечающих этим горизонталям. В нашем примере максимальное значение косинуса равно 1, а остальные близки к нему. Если числовая ячейка двойной точности соответствует примерно четырнадцати десятичным разрядам, то, для того чтобы число различных машинных значений функции стало меньше числа горизонталей, минимум должен отличаться от максимума только в единицах двенадцатого-тринадцатого разрядов. А так как вблизи экстремума график является параболой, то порядок длины отрезка, на котором строится график, должен быть вдвое меньше. Поэтому наиболее четко проявляются эти эффекты, когда B близко к 10^{-6}.

Когда же B продолжает уменьшаться дальше, мелкие уступы на графике начинают превращаться в более заметные ступени. Одновременно увеличивается число беспорядочных отклонений от основной линии и амплитуда этих отклонений. Она как раз и равна разности между изображением на экране соседних машинных чисел. Наконец, когда B станет меньше 10^{-7}, эти беспорядочные скачки начинают преобладать настолько, что саму кривую становится сначала трудно, а затем и вовсе невозможно увидеть.

Обратите внимание и на то, что эффекты, связанные с дискретностью, не подчиняются формулам приведения. Если увеличить A и B одновременно на одну и ту же величину, кратную периоду, то изображение на экране изменится, вопреки свойствам функции. В частности, если эта добавка сравнительно велика либо по сравнению с ней относительно мала разность между B и A, то дискретность начнет проявляться не только по вертикали, но и по горизонтали.

Точность и достоверность вычисления*

Примеры предыдущего параграфа подводят нас вплотную к вопросу о точности и достоверности машинных вычислений. Наша цель — понять, по каким причинам происходит потеря точности вычислений, когда

ошибки накапливаются до такой степени, что теряется достоверность окончательных выводов, и как можно контролировать возникновение подобных ситуаций, избегать их и добиваться гарантированной точности вычислений.

Причины, приводящие к замене точного значения приближенным, могут быть самыми различными. Одна из них состоит в том, что существуют иррациональные числа, которые попросту невозможно записать конечным числом цифр или поместить в ячейку с конечным числом разрядов. В отличие от этой, внутриматематической причины, другая причина связана с характером приложения математических понятий. Никакое измерение не может быть абсолютно точным. Дело не только в несовершенстве инструментов, но и в непостоянстве измеряемых величин во времени, и в принципиальных препятствиях получения сверхвысокой точности. Например, молекулярное, атомное и внутриатомное строение вещества не позволяет достигнуть абсолютной точности измерения длин и расстояний. Даже некоторые целые величины, например численность населения нашей страны или всей планеты, невозможно узнать точно: они изменяются значительно быстрее, чем могут быть учтены.

Приближенным может быть не только число, но и формула или метод вычисления. Прилагательным «приближенный» фактически пользуются каждый раз, когда один объект — истинный, точный — в процессе вычисления заменяется другим — неточным, округленным, ошибочным. И хотя корень этого слова подразумевает близость точного значения и его приближения, степень этой близости никак не регламентируется.

Понятие погрешности позволяет количественно оценить степень близости точного и приближенного значений. Абсолютная величина их разности называется абсолютной погрешностью. Относительной погрешностью называется отношение абсолютной погрешности к точному значению. В ряде случаев (например, при оценке очень больших или близких к нулю величин) пользуются и другими понятиями погрешности. Подчеркнем, что о погрешности, за исключением чисто теоретических исследований, обычно говорят в том случае, когда точное значение неизвестно. Поэтому значения погрешности сами почти всегда только приближенны.

Из курса обществоведения вы знаете основные законы диалектики. В одном из них утверждается о переходе количественных изменений в качественные. Именно этот закон «отвечает» за то, что возрастание погрешностей исходных данных и накопление погрешностей в продолжительных (например, циклических) вычислениях нередко приводят к качественно неверному результату.

Алгоритм, исполнение которого над приближенными значениями исходных данных приводит к неконтролируемому росту погрешностей и качественно неверным результатам, называется некорректным. В приближенных вычислениях допустимо использование только корректных алгоритмов. С этой целью проводятся специальные исследования, проверяющие корректность наиболее часто используемых алгоритмов.

Так как, наряду с накоплением погрешностей элементарных операций, неверный результат может получиться и по причине погрешностей и ошибок в самом алгоритме, то каждый программист должен проводить испытания своих программ. Эти испытания должны начинаться с этапа отладки программы, т. е. проверки правильности отдельных ее подпрограмм и блоков. Затем должны быть проведены контрольные вычисления, для чего подбираются примеры, результат в которых либо заранее известен, либо может быть найден вручную, либо легко прогнозируется. Наконец, следует провести несколько серий вычислений, чтобы точно установить, корректен ли алгоритм, в каком диапазоне исходных данных он обеспечивает «хорошую» и какую именно точность результата, в каком диапазоне алгоритм дает качественно верный результат низкой точности, а в каких случаях обращение к этому алгоритму вообще нецелесообразно.

Особое внимание следует уделить аккуратному составлению инструкции по пользованию программой. Это необходимо потому, что вычисления стали коллективными: программы написанные одними программистами, включаются в качестве подпрограмм другими, а используются третьими. Если ничего не было известно о качестве подпрограмм, то ничего нельзя сказать и о достоверности результата.

Примером небрежно составленной инструкции может служить руководство по эксплуатации микрокалькулятора «Электроника МК-61». Там сказано, что при вычислении тангенса его аргумент не должен превосходить 10^{10}, что обеспечивает максимальную относительную погрешность $3 \cdot 10^{-7}$. Между тем индикация и размер ячейки этого микрокалькулятора позволяют хранить только 8 десятичных цифр мантиссы. Это означает, что у девятизначного целого числа (меньшего 10^{10}) будет проигнориро-

вана — заблокирована при вводе или отброшена при округлении результата вычисления — последняя цифра мантиссы, т. е. абсолютная погрешность аргумента составит ± 5. Период же тангенса, в пределах которого эта функция может принять любое вещественное значение, в три раза меньше. Поэтому для значений аргумента от 10^9 до 10^{10} микрокалькулятор не только не обеспечивает той относительной точности $3 \cdot 10^{-7}$, которая указана в руководстве, но вообще не может выдать достоверного значения тангенса.

Этот пример указывает на необходимость испытания и «чужих» программ, а также вычислительных устройств с целью проверки правильности составленных для них инструкций и руководств. Окончательным аргументом здесь является не печатное слово, а сама программа или машина.

Когда же программа испытана, следует предельно тщательно соблюдать составленную для нее инструкцию. В отличие от человека, ни машина, ни программа не обладают «здравым смыслом», а чисто формально исполняют те инструкции, которые в них введены.

Можно, однако, посмотреть на это и с другой стороны, относясь к машине как к наивному, но способному к обучению ребенку. «Обучение» состоит в том, что каждый раз, встретившись с неправильным использованием программы, вы вносите в нее дополнения и изменения, позволяющие впредь избежать подобной ошибки. Разумеется, «обучать» может только весьма квалифицированный «учитель», знающий до тонкостей и саму программу, и возможности машины.

ЧТО МОЖЕТ ЭВМ?

Опасности действительные и мнимые

Дисплею приписывают множество опасных свойств. В начале 80-х гг. по западным странам прокатилась волна паники: у женщин, много времени проводивших за дисплеями, обнаружились отклонения при родах. Виновником считали рентгеновское излучение кинескопа. Однако тщательные измерения показали, что оно ни при чем: его величина меньше, чем природное излучение в горных районах, да и та регистрируется лишь в нескольких миллиметрах от экрана.

А вот то, что экран излучает свет, действительно может быть опасным. Ведь при одновременной работе с дисплеем и текстом на бумаге глазам приходится то и дело перестраиваться с восприятия отраженного от бумаги света на восприятие света, излучаемого экраном. Если эргономика рабочего места продумана недостаточно, это может плохо сказаться на зрении.

Еще одна реальная опасность — пыль. При работе дисплея его экран электризуется, начинает притягивать пыль, и ее концентрация вблизи дисплея повышается. Это может осложнить жизнь людям, склонным к аллергии, а также привести к кожным заболеваниям.

И наконец, еще одно — аэроионы. Известно, что повышенное содержание в воздухе отрицательных ионов (как, например, в сосновом бору) оказывает благотворное воздействие на здоровье. Предполагается также, что положительные ионы могут ухудшать самочувствие. Статические электрические поля дисплея обогащают окрестности именно положительными ионами. На вопрос, не опасно ли это, врачам и физиологам еще предстоит ответить.

Компьютер учится читать

Когда с компьютером сталкиваются люди, которые раньше знали о нем только понаслышке, их часто постигает глубокое разочарование, особенно после того, как обнаружится, что этот страшно дорогой и умный прибор не только ничего не видит, он еще ничего не слышит, а без правильных команд ничего не говорит.

Чтобы хоть как-то помочь этим людям, а также огромной армии машинисток, журналистов и операторов фотонаборных машин, фирма «Сорикон» предлагает устройство для ввода текста. Небольшой пенал, легко умещающийся в руке,— вот и всё, что позволяет считывать с печатной страницы текст со скоростью до 170 слов в минуту и помещать его на диск. В верхней части пенала расположен источник света и фотоэлемент, распознающий находящийся под ним знак. Плавно проводя устройство вдоль строчек, оператор заставляет компьютер «увидеть» текст.

Разработчики считывателя обещают в 99 случаях из ста распознать предлагаемую их детищу букву, даже если у вас недостаточно твердая рука. В программу, обслуживающую считыватель, которая занимает в памяти компьютера совсем немного места, заложена возможность автоматической компенсации нестабильности скорости сканирования и непараллельности перемещения считывателя по строке.

Считыватель воспринимает любой машинописный или печатный текст, отпечатанный на лазерном или матричном принтере и вообще любых устройствах печати, независимо от формы или величины шрифта. Перед началом работы считыватель адаптируется к предлагаемой форме и контрастности текста.

Л. ГОРОДНЯЯ
г. Новосибирск

Сравнение учебных языков программирования Бейсик, Паскаль и Рапира

Естественность языковых средств программирования и удобство их применения определяют трудоемкость их изучения и использования. Поскольку допустимые трудозатраты всегда ограничены вполне реальными силами, ресурсами, временем, постольку принципиально ограничена сложность задач, решаемых с помощью определенных языков. Следовательно, выбирая производственный язык программирования, мы фактически выбираем и класс задач, в пределах которого сможем работать, и пространство решений, которые сумеем представить, точнее, построить из конструкций выбранного языка.

Каждый язык программирования формируется вокруг определенных целей, в конкретной обстановке, на существующем уровне понимания возможностей ЭВМ и содержания работы программиста, на фоне системы вкусов и предпочтений, сложившихся ко времени создания языка. Поэтому по мере расширения сферы применения ЭВМ возникает вполне закономерная потребность в построении новых языковых средств. Это не всегда означает, что прежние средства несостоятельны. Чаще это симптом несоответствия новым задачам традиционной реализации средств программирования и методики их применения. Но на практике язык программирования воспринимается неразрывно с системой программирования и привычной методикой работы. Следовательно, при выборе языка программирования для любой области применения необходимо проанализировать не только собственно язык (его определение и объем подготовленных на нем программ), но свойства доступной системы программирования и методику основной работы с системой и программами.

Паскаль создан для обучения основам программирования учащихся вузов. Вузовский цикл преподавания (лекция, семинар, практика) существенно повлиял на структуру языка и методику его преподавания. Основы программирования сводятся к задаче представления программ, техника представления оттачивается на стереотипном классе вынужденно решаемых при программировании на Паскале задач по обработке векторов и текстов, выработаны удобные для преподавателя приемы оформления программ, для самостоятельной работы требуется предварительное накопление некоторой «критической массы» знаний о языке и его программном окружении. Реализация Паскаля на «Ямахе» (Turbo-Pascal) является наиболее приспособленной к применению в учебном процессе из известных автору реализаций этого языка.

Бейсик возник как практическое средство для быстрого привлечения практикантов и лаборантов к работе на ЭВМ. Он ориентирован на доступ к основным возможностям оборудования при выполнении разовой работы, не рассчитанной на долгую жизнь и не требующей поэтому особой квалификации от исполнителя или качества результата. По этой причине типовая реализация Бейсика — небольшая операционная система, в которой Бейсик является и языком заданий, и языком программирования, часто единственным. Начальное знакомство с такой системой просто и приятно, но переход к систематической работе усложнен необходимостью наработки профессиональных приемов по организации программ (в случае Паскаля такие приемы предложены в языке: процедуры, функции, области действия имен, передача значений с помощью параметров, конструирование типов данных). Бейсик является самым легкодоступным на микропроцессорной технике языком. Обычно он встроен в ПЭВМ, поэтому начальное знакомство возможно при мини-

мальной конфигурации оборудования.

При разработке Рапиры предпринята попытка объединить достоинства ранее сложившихся учебных языков программирования, избежав их недостатков по отношению к задаче преподавания основ информатики учащимся средней школы. Язык Рапира сформировался в процессе факультативной работы со школьниками разных возрастов в условиях регулярного доступа к ЭВМ. Типовая реализация, как и в случае Бейсика, обеспечивает простоту начального знакомства. Языковые средства организации программ, как и в случае Паскаля, обеспечивают переход к систематической экспериментальной работе, но с помощью средств, более соответствующих возрастным особенностям обучения школьников. Рапира реализована в составе программной системы «Школьница» на ПЭВМ «Агат», разработаны системы программирования для Рапиры на ПЭВМ «Ямаха» (должна быть перенесена на «Корвет») и СМ-4 (перенесена на «Электронику УКНЦ»).

«Стартовый барьер» перехода к практике на ЭВМ

Для первой лабораторной работы по Бейсику достаточно познакомиться с простейшим вариантом оператора вывода и записью констант и операций. Это позволяет изучить особенности обработки значений и их границы. Причем возможно оперативное получение результатов каждого шага работы, что исключает случайное накопление ошибок в понимании.

```
PRINT 123,5+3,7—2,9*11,16/3
```

Аналогично в Рапире:

```
вывод: 123,5+3,7—2,9*11,16/3
```

Такая работа на Паскале требует знакомства с оформлением программы и понимания, что Паскаль-программа сначала транслируется, затем исполняется, для чего следует набирать соответствующие команды. (Даже если предусмотрена возможность задать одной командой трансляцию и исполнение, в дальнейшем, при переходе к работе с разными данными, придется трансляцию от счета отделить из соображений эффективности, чтобы не терять свое время на ожидание конца трансляции.)

```
program PROG;
  begin
    writeln (123,5+3,7—2,9*11,16/3)
  end.
```

Обнаружение ошибок

В случае Бейсика несоответствие программы определению изучаемого языка или невозможность исполнения показывается диагностическим сообщением с указанием номера строки, содержащей ошибку. Сообщение обычно содержит указание типа ошибки, естественно, на английском языке.

```
10 PRINT 2+3+
RUN
MISSING OPERAND IN 10
```

Здесь в 10-й строке пропущен операнд.

В случае Паскаля появится примерно такое же сообщение, только без номера строки. По нажатию клавиши «конец строки» курсор установится на строке, содержащей ошибку, в позиции, где она обнаружена.

```
Missing operand
program PROG;
  begin
    writeln (2+3± )
  end.
```

(Сообщение исчезает с экрана, когда появляется ошибочный текст. Его придется запомнить.)

В случае Рапиры появится сообщение на русском языке. На экране «Ямахи» оно высвечивается в особом окошке. Если правильно подобрать размеры окон для текстов программы и процедур, то можно сообщение и контекст ошибки увидеть на экране одновременно и, следовательно, не запоминать.

```
Требуется операнд
вывод: 2+3±
```

При желании и для Паскаля нетрудно обеспечить русскую диагностику. Достаточно всего лишь сменить файл с текстами сообщений, что некоторые сделали для себя, но не имеют возможности распространить свои результаты.

Коррекция текста программы

Чтобы исправить ошибку в тексте на Бейсике, надо с помощью команды LIST вывести на экран строки, содержащие корректируемый текст.

```
LIST 10—20
10 PRINT 2+3+
OK
```

Появился текст — надо найти в нем ошибку и после ее исправления (стереть лишний плюс или добавить еще одно число и, главное, не позабыть про «ритуальную» клавишу «конец строки», иначе вопреки изображению на экране исполнится прежний ошибочный текст) можно сразу повторно исполнить программу и заняться следующей ошибкой или увидеть результат работы программы.

```
10 PRINT 2+3+4
RUN
9
OK
```

В тексте на Паскале или на Рапире курсор автоматически устанавливается в позиции предполагаемой ошибки. После исправления текста на Рапире можно сразу исполнить программу, а на Паскале — оттранслировать и затем исполнить.

Организация программы

В Бейсике программа — набор пронумерованных операторов, взаимосвязанных с помощью переходов по номерам. Незанумерованные операторы исполняются немедленно в диалоге. Набор номера является сигналом, что предстоит сохранить оператор в памяти ЭВМ. Чтобы исполнить программу, достаточно набрать команду RUN.

В Паскале программа представляет собой целостную систему вложенных блоков с выделенной головной программой. Программа транслируется и исполняется с помощью особых команд.

Рапира определяет программу как одноуровневый набор процедур и функций, каждая из которых пишется как бы на отдельной странице. Страницу можно найти по имени записанной на ней процедуры. Исполнение осуществляется непосредственно в диалоге или из других процедур предписанием вызова (аналогично для функций, только их вызовы выглядят как формулы в выражениях).

Элементарные данные и операции

Элементарные типы данных и набор операций над ними задают в языке программирования основную область приложения языковых средств. Формально Бейсик, Паскаль и Рапира обладают общей областью приложения — это обработка целых и вещественных чисел и текстов с помощью общего набора операций и встроенных функций. Но типовые системы программирования для этих языков имеют заметными различиями, существенными для учебного процесса.

В Паскале принято целые и вещественные числа рассматривать как объекты принципиально различной природы, что требует специальных пояснений, трудно воспринимаемых учащимися. Кроме того, в учебных экспериментах приходится сразу знакомиться с особенностями работы на границах диапазона целых, так как он невелик, что методически не обусловлено для учащихся средней школы, является преждевременным усложнением.

Бейсик допускает числа в более широком диапазоне, причем строгого разделения на целые и вещественные не предусмотрено, поэтому длинные целые автоматически переводятся в форму с плавающей запятой, что расширяет допустимый диапазон целых чисел в программах. Значения параметров цикла FOR особо ограничены.

Рапира также обеспечивает обработку чисел без строгого разделения целых и вещественных. В некоторых реализациях снято фиксированное ограничение на длину представления чисел. Естественным ограничением является общий объем памяти, необходимой для работы программы. Поэтому допускается свободный учебный эксперимент, выполнимый без знания технических деталей организации процесса обработки данных на ЭВМ. При вводе данных в диалоге с программой возможны некоторые ограничения длины набора строкой экрана (на СМ-4 и «Электронике УКНЦ»), форматом экрана («Школьница» на «Агате»), общим наличием памяти (на «Ямахе»).

Работа с текстовыми строками в этих языках различается незначительно (особенностями в выборе подстрок).

Идентификация

Обычно язык программирования позволяет обозначать значения и функции с помощью имен, выбираемых программистом. Свобода в выборе таких имен, возможность систематизировать обозначения, связывать их с мнемоникой естественного языка существенно повышают удобство работы преподавателя с программой и упрощают долговременную экспериментальную работу.

В Бейсике имена различаются по малому числу литер (часто по двум первым) и для текстовых данных содержат специальный символ-флаг, отличающий их от имен чисел. Это упрощает Бейсик-интерпретатор, но требует специальных пояснений при изучении, а также специальных навыков по обозначению данных небольшим набором различных имен. Обычно допустимы только латинские буквы.

N1 ST ST$ последнее — имя строки
LETTER LENGTN LE неразличимые имена

Паскаль предлагает активное использование длинных имен, почти фраз, из латинских букв без пробелов между словами (Turbo-Pascal допускает подчеркивания вместо пробелов), что удобно преподавателю при чтении ученических программ. Но во многих реализациях Паскаля это удобство ослаблено тем, что имена фактически различаются по первым 6—8 символам.

NAMEOFVERYLONGSTRING имя очень длинной строки

NAME _ OF _ VERY _ LONG _ STRING можно в некоторых реализациях.

В Рапире также активно используются длинные имена, но из русских и латинских

букв. Определение языка требует различения имен по полному набору образующих их символов.

ИМЯ_ОЧЕНЬ_ДЛИННОЙ_СТРОКИ можно русские фразы HARDWARE_BOOK можно и английские.

Области действия имен

Имена в программе на Байсике глобальны, поэтому в случае сколько-нибудь больших программ приходится согласовывать их использование, что требует специальных навыков.

В Паскале имена локализуются по статически вложенным блокам, благодаря чему возможна согласованная разработка независимых частей программы. Согласование необходимо для совместно используемых имен в статически объемлющих блоках.

В Рапире вложенных блоков нет, но автоматически локализуются имена параметров функций (процедур) в тексте функций (процедур). Локализацией других имен можно управлять с помощью специальных предписаний, задающих способ поиска значения имени в зависимости от цепочки ранее вызванных незавершенных функций и процедур.

Описание типов данных

Переменной в Паскале присваиваются значения одного заранее определенного типа, описываемого при объявлении переменной в спецификации блока. Это позволяет на этапе трансляции осуществлять контроль типов данных при присваиваниях до исполнения программы, что в случае больших программ и/или данных экономит машинное время и упрощает отладку.

```
program PR;
  var N, L, M: integer; (*имена целых*)
      C, H: char; (*имена литер*)
      A: array [1..10] of integer;
        * имя вектора из целых чисел*)
      S: array string [20];
        (*имя текста, т. е. вектора из
        литер*)
  begin
...
    N:=1;
    C:='X';
    A [3]:=120;      (*присваивание числа
        компоненту вектора*)
    S:='string'; (*присваивание строки целиком*)
...
  end.
```

Переменные, обозначающие элементарные данные, в Бейсике можно не описывать. Но к имени переменной, обозначающей строку, приписывается литера $, чтобы отличать их от переменных, обозначающих числа. Обязательно объявляются заранее переменные, обозначающие массивы. Контроль типов данных при присваивании выполняется при использовании программы.

```
10 C$='H'      'присваивание литеры
20 S$='string' 'присваивание строки
30 S=123.45    'присваивание числа
40 DIM A (5)   'объявление массива
50 A (3)=120'присваивание числа компоненту массива
```

Имена в Рапире могут обозначать значения любого типа без предварительного объявления, т. е. вводятся по мере необходимости.

```
СИМ:=«X»
N:=1
НОМЕР:=124
ПИ:=3.14
строка: = «string»
ВЕКТОР:=⟨2, 4, 6, 9⟩\присваивание кортежа
ВЕКТОР [1]:=8      \присваивание компоненту кортежа
A:=ВЕКТОР          \пересылка кортежа
A [3]:=120
```

Распределение памяти

В Паскале распределение памяти выполняется на этапе трансляции. При недостатке можно организовать в программе перераспределение памяти с помощью встроенных процедур отказа от определенных ячеек памяти (dispose) и повторного их использования (new).

В Бейсике память под данные отводится при обработке объявления массивов (DIM) и по мере появления имен переменных. Перераспределение памяти выполняется как отказ от ненужных данных (ERASE).

В Рапире память под данные отводится по мере формирования значений, в том числе и составных (кортежей). Освобождающаяся при этом память автоматически используется повторно.

Структуры данных

Паскалем предусмотрена статическая, т. е. заранее продумываемая, прогнозируемая организация структур данных с помощью векторов, множеств и записей из определенного числа однотипных элементов (для записей — определенного типа). Динамические, т. е. зависящие от текущей ситуации, структуры данных или «накрываются» статическими структурами, или моделируются в программе с помощью встроенных процедур, выполняющих выделение и освобождение памяти (new, dispose).

Бейсик чуть проще благодаря динамиче-

ской, диалоговой схеме обработки программ и ориентации на один тип структур данных — массивы из однотипных элементов.

Рапира обеспечивает динамическое конструирование и представление любых иерархически организованных структур данных — кортежей — без ограничений на типы элементов, число компонентов и глубину их вложенности, не требуя предварительного объявления. Данные реорганизуются по ходу работы.

Исходные данные

В Бейсике для задания исходных данных необходимо перечислить входящие в них элементарные данные в специальном операторе DATA, а в программе организовать цикл, выполняющий запись этих данных в нужные компоненты структур данных.

```
10 DATA 11, 22, 33, 44, 55, 66
20 DIM A (5)
30 FOR I=0 TO 5
40 READ A (I)' запись очередного данного
                в массив
50 NEXT I
```

В стандартном Паскале для этого придется трудолюбиво выписать 10 присваиваний (или в программе организовать чтение данных из файла или клавиатуры).

```
program PR2;
  var A: array [1..6] of integer;
  begin
    A[1]=11;
    A[2]=22;
    A[3]=33;
    A[4]=44;
    A[5]=55;
    A[6]=66;
...
  end.
```

Turbo-Pascal предоставляет более щадящее решение:

```
const A: array [1..6] of integer=
      (11, 22, 33, 44, 55, 66);
```

Но выполнить это можно только при описании переменных.

В Рапире достаточно по мере необходимости выполнять присваивание кортежа переменной:

```
A:=⟨11, 22, 33, 44, 55, 66, 77, 88, 99, 0⟩
```

Вывод структур данных

Чтобы посмотреть результаты, накопленные в массиве, на Бейсике требуется написать цикл, организующий покомпонентный вывод полученных данных.

```
10 DIM A (10)
...
200 FOR I=0 TO 10
210 PRINT A (I)
220 NEXT I
```

Аналогичная работа типична и для Паскаля при обычном выводе на экран.

```
program PR3;
  var A: array [1..10] of integer;
  begin
...
    for i:=1 to 10
      write (A[i]);
...
  end.
```

На Рапире независимо от структуры данного достаточно одного оператора.

```
вывод: A
```

В случае ввода данных и обмена с внешней памятью картина такая же.

Слияние структур данных

При необходимости совместной обработки раздельно полученных данных на Бейсике (и на Паскале) необходимо заранее предусмотреть массив достаточной для этого размерности и написать программу переписи этих данных в общий массив.

```
10 DIM  A(10), B(20), C(30)
...
200 FOR I=1 TO 10
210 C(I)=A(I)
220 NEXT I
230 FOR I=1 TO 20
240 C(I-10)=B(I)
250 NEXT I
```

На Рапире это выполняется операцией склеивания кортежей, результат которой присваивается переменной.

```
C:=A+B
```

Для текстовых строк и в Бейсике, и в Паскале обеспечен такой же способ склеивания данных.

Передача параметров

Параметризация формул является мощным средством унификации записи программ, а главное, методом принципиальной экономии трудозатрат на отладку с обеспечением надежности функционирования программ и, естественно, с разделением труда.

Средства выделения подпрограмм в Бейсике не поддерживают никакого механизма параметризации (кроме функций). Поэтому передачу параметров приходится организовывать с помощью присваиваний переменным и циклов в случае пересылки массивов.

В Паскале процедуры и функции обеспечены передачей параметров при совпадении типов параметра и передаваемого данного.

В Рапире выполняется передача параметром любого данного, включая процедуры и функции. То обстоятельство, что возможное несоответствие типа значения типу операции

обнаружится лишь при исполнении, мало сказывается на удобстве учебного диалога, в котором разница между моментом передачи параметра и моментом его применения мало существенна (в отличие от производственного применения программ, когда необходимо не допустить исполнения сколько-нибудь сомнительного процесса). При начальном знакомстве это освобождает от принудительного планирования структуры данных и ее представления.

Модификация программы

На начальных этапах отладки программ на Бейсике можно просто наслаждаться удобством вставки новых команд. Стоит лишь набрать строку с новым номером, как она вставится в нужное место даже без подвода курсора. Но вскоре исчерпывается резерв свободных номеров строк, и возникает необходимость перенумерации. Однако при программировании сколько-нибудь большой программы обычно выполняют распределение и учет диапазонов номеров строк подпрограмм и имен переменных и функций. При модификации программ приходится внимательно отслеживать взаимосвязи подпрограмм по переходам. После перенумерации программы это приводит к трудноуловимым дефектам, потому что номера подсознательно связываются с отдельными осмысленными частями программы. При абсолютной уверенности в правильности программист обращается к таким частям по устаревшим номерам. Искать такие ошибки очень трудно. Это провоцирует воздержание от перенумерации построением длинных строк, содержащих много команд, что быстро портит обозримость текста и усложняет работу по его коррекции и модификации вплоть до несостоятельности автора в поддержании работоспособности программы. (Можно рекомендовать специальную методику работы с Бейсиком на основе систематической перенумерации текста и наглядных комментариев, обеспечивающих профилактику подсознательного связывания смысла с номерами строк. Но кто может утверждать, что умеет прививать школьникам склонность к добровольной дисциплине, связанной с непонятно когда окупающимися накладными расходами? Одни из них не верят в ограниченность своих способностей, другие затрудняются набирать лишние символы, не нужные машине. Неприятности они видят сейчас, а мифический выигрыш ждет потом.)

В Паскале при модификации программы необходимо отслеживать взаимодействие с именами в объемлющих блоках. Они всегда расположены выше по тексту программы вне блоков равного ранга. Поэтому естественная схема развития Паскаль-программ — «сверху вниз», что подходит для хорошо отработанных, стабильных постановок задач.

В Рапире при модификации процедуры необходимо понимание обстановки, в которой предстоит исполнять эту процедуру, что требует учета смысла имен процедур и внешних имен значений, используемых динамическими предшественниками. В не слишком больших программах такая работа сводится к просмотру страниц, содержащих упоминание модифицируемой процедуры. Достаточно естественно выполняется как нисходящее, так и восходящее развитие программы, что пригодно и для экспериментального решения слабоизученных задач.

Трудоемкость программирования

Трудоемкость программирования можно измерять в соответствии с тремя градациями.

1. Выбор обозначений в программе не требует никаких искусственных приемов, и при работе программы не возникает никаких проблем ни с объемом памяти, ни со скоростью исполнения. При таких условиях программирование не вызывает особых затруднений, но выход из таких границ зависит от особенностей реализации языковых конструкций.

2. Затруднения в выборе обозначений могут быть разрешены систематизацией и учетом, недостаток памяти нейтрализуется своевременным отказом от ненужных данных, а скорость становится приемлемой в результате рациональной организации циклов. Работа на этом уровне сложности требует некоторых профессиональных приемов, до которых надо додуматься или которым надо у кого-то научиться. Во всяком случае реорганизация циклов, систематизация, учет и отказ от значений — неизбежные накладные расходы.

3. Трудности с обеспечением приемлемого решения задачи столь существенны, что приходится тщательно организовывать переобозначение данных и частей программы, обмен с внешней памятью, реализовывать специальные способы представления и обработки информации, позволяющие исполнить программу на грани возможностей имеющихся технических средств. Независимо от языковых средств работа на этом уровне сложности принципиально очень трудоемка даже при высокой профессиональной квалификации. Приближение к границам возможностей и трудоемкость достижения предельной производительности зависят от приспособленности используемых средств к выражению эффективных решений, что требует

от программиста и знаний, и навыков, и опыта.

На Бейсике первый уровень сложности исчерпывается очень быстро на сравнительно малых программах и данных из-за малой мощности набора различных идентификаторов и применения номеров строк в качестве обозначения частей программы. Второй уровень сложности имеет высокую трудоемкость из-за неудобства модификации программ. До необходимости работать на третьем уровне сложности дело доходит крайне редко. Проще изучить еще один язык, более приспособленный для сложной работы.

На Паскале первый уровень сложности связан с умением использовать английские обозначения, но благодаря трансляционной схеме обработки программ довольно некритичен в отношении памяти для программы и данных. Программисту, владеющему иностранным языком или привыкшему пользоваться транслитерацией, первый уровень сложности позволяет решать довольно широкий класс задач. Второй уровень сложности возникает при больших объемах данных и при организации динамических структур. Необходимость работы на третьем уровне сложности возникает не скоро, так как определение языка и его типовая реализация приспособлены к достаточно эффективной работе.

На Рапире первый уровень сложности более прост по системе обозначений — русские и латинские литеры, но более ограничен по объему памяти для программы и данных. Теоретически возможна реализация, поддерживающая такую работу на более широком классе задач, но она пока не выполнена. Из-за недостатка памяти на имеющихся школьных ПЭВМ вскоре возникает необходимость экономии, и работа приобретает сложность второго уровня. Переход к работе на третьем уровне в Рапире не поддержан; возможно, в этом нет необходимости, так как для чисто производственного применения язык не предназначен. Но навыки, приобретенные при работе с Рапирой на втором и третьем уровнях сложности, пригодятся в работе на производственных языках.

Ю. КУЗЬМИН, И. ГВАРДИНА
Латвийский государственный университет

Система «Рига» для КУВТ-86 с Бейсиком

КУВТ-86 с транслятором языка Бейсик отличается от КУВТ-86 с Фокалом не только транслятором, но и программами обслуживания информационной сети. Если Фокал-вариант реализует инициативу обмена файлами со стороны ДВК-2М, то Бейсик-вариант — со стороны БК-0010. Такое принципиальное различие решений потребовало реализовать систему «Рига» в двух вариантах.

Внешне основное их различие в том, что в новой реализации перед работой сети учитель должен вызвать программу инициализации каждого БК-0010. Эту процедуру нужно повторять при каждом сбое сети КУВТ.

Инициализация БК-0010. В связи с тем что передача файлов от ДВК-2М возможна только после запроса от БК-0010, после запуска системы «Рига» нужно вызвать режим 8 (Выход на Бейсик-сеть, разработанную в ВЦ Вильнюсского университета), а с БК-0010 запросить BLOAD "TT:R".

Произойдет передача программы R, которая позволит в дальнейшем проводить обмены файлами по инициативе ДВК-2М. После передачи R на экране БК-0010 появится сообщение «ЖДУ». Теперь можно передать в БК-0010 файл системы «Рига». Для этого нужно нажать на дисплее ДВК-2М клавиши **СУ** и **ЦС** одновременно, чтобы выйти из режима инициализации.

Программа R позволяет выполнять все работы, описанные в инструкции по работе в сети КУВТ с Бейсиком.

Передача файла в БК-0010. Для передачи файла в БК-0010 из ДВК-2М нужно в основном кадре задать режим 1 (ПЕРЕДАЧА В БК), после чего ЭВМ запросит имя файла. Если в ответ нажать **ВК**, то будет выдан каталог файлов. Если же ввести имя файла, то ЭВМ вызовет его и запросит ввод номеров БК-0010 для передачи файла.

Процесс передачи файла занимает около 20 с. При этом на экране дисплея образуется двоичный образ файла, показывающий процесс приема файла в БК-0010. После успешной передачи файла он запускается автоматически.

Прием файла из БК-0010. Информация,

введенная с магнитофона или с пульта в БК-0010, может быть записана на дискету ДВК-2М. Ввод программ на Бейсике производится в режиме 8 (ВЫХОД НА БЕЙСИК-СЕТЬ) по правилам, приведенным в инструкциях по работе с КУВТ (поставляются заводом).

Ввод других файлов производится в режиме 2 (ПРИЕМ ИЗ БК). При записи файлов, не имеющих расширения RIG или RED, программа запрашивает адрес начала файла в памяти БК-0010 и длину файла — все вводится восьмеричными числами.

Печать файлов. Файлы можно распечатать, если они написаны на Т-языке или подготовлены редактором на БК-0010 (расширения RIG и RED соответственно).

Каталог. Большинство дискет системы «Рига» содержат файл CAT, в котором приведены имена файлов и краткое их содержание на русском языке. Такой файл можно вызвать в режиме 4. Пример каталога приведен ниже.


```
КАТАЛОГ        ДИСКЕТА ПРИМЕНЕНИЙ ЭВМ

Р43      —  ПРАВОПИСАНИЕ АНГЛИЙСКИХ ГЛАГОЛОВ (ОБУЧАЮЩИЙ ТЕКСТ)
Р45      —  ТРЕНАЖЕР ПО ЧАСТЫМ АНГЛИЙСКИМ СЛОВАМ
...         ...
```

Электронная почта. Возможности электронной почты нового варианта существенно расширены. Введены новые запросы:

справка по директивам;

регистрация учеников;

обмен сообщениями между БК-0010;

передача файлов по запросам от БК-0010.

После входа в электронную почту нужно указать активные БК-0010, участвующие в обменах (пробел означает активность всех ЭВМ). Затем вводится директива С (по ней выдается краткая справка по директивам) или номера БК-0010 — им будет передано сообщение с ДВК-2М или с них будут приняты сообщения в ДВК-2М (пробел означает активность всех БК-0010). Затем можно вводить сообщение — до 24 символов при передаче из ДВК-2М в БК-0010. Если вместо ввода сообщения нажать ВК, то в ДВК-2М будут собраны сообщения из тех БК-0010, которые были объявлены активными.

```
ЭЛЕКТРОННАЯ ПОЧТА
ВВЕДИТЕ НОМЕРА АКТИВНЫХ БК 3, 4, 5, 12
ТЕКУЩАЯ СВЯЗЬ: 4, 5, 12
ВВЕДИТЕ НОМЕРА БК ИЛИ КОМАНДУ (С-СПРАВКА)
ДВК: НАБЕРИТЕ СВОИ КОДЫ
  4:00004:10A15
  5:00005:10A02
 12:00012:10A23
НЕ ГОТОВ: 3
```

В примере работы почты первая строка содержит просьбу учителя, а остальные — ответы учеников. За пультом № 4, код регистрации которого был 00004, введен новый код регистрации 10А15, (10А класс, ученик, номер которого по классному журналу равен 15). Если теперь в ДВК-2М ввести символ Р (РЕГИСТРАЦИЯ), то все последующие сообщения от этого ученика (и результаты тестов) будут сопровождаться кодом 10А15.

Другие команды почты:

П — печать сообщений учеников;

Ф — занесение сообщений учеников в электронный журнал на дискете ДВК-2М;

В — однократное выполнение запросов учеников;

А — автоматическое (многократное) выполнение запросов учеников.

Запросом ученика может быть, например, запрос на получение в БК-0010 файлов из ДВК-2М или на передачу сообщений от одного БК-0010 другому.

Внимание! При запросе нельзя нажимать на БК-0010 клавишу **ВВОД** до отработки запроса.

Электронный журнал. ЭВМ позволяет ввести в учебный процесс новый способ регистрации данных. Система «Рига» сохраняет данные в специальном файле — R.PRO. При этом каждое сообщение начинается кодом ученика, по которому при анализе можно установить принадлежность данных. В файл R.PRO заносятся и данные об ошибках учеников в тестах знаний.

Структура электронного журнала следующая:

Признак	Код	Имя файла	Сообщение (параметры)
2 байта	5 байтов	6 байтов	64 байта (52 байта)

Здесь:

признак равен нулю в конце журнала; положителен для активных данных; отрицателен для пассивных (временно исключенных) данных;

код — код ученика;

имя файла — файл, в котором произошло

обращение к электронному журналу;

сообщение — сообщение с БК-0010 или с ДВК-2М. Последнее всегда следует перед сообщениями с БК-0010, в нем указываются данные о классе, дата и т. п.;

параметры — 26 параметров, собранных с БК-0010 во время тестирования знаний учеников.

Данные, собранные в файле R.PRO, анализируются программой RIGA2. Она находится в стадии развития, так как нет достаточного опыта работы с электронным журналом. Учитель может создать свой вариант RIGA2. Выход на эту программу производится вызовом режима 7 в основном кадре системы «Рига».

Вот типичные вопросы, на которые отвечает программа:

какой тест был самым трудным для IX Б класса;

какой ученик сделал меньше всего ошибок в тесте Т13;

сколько ошибок в Т21 сделал X А класс;

какой вопрос был самым трудным в тесте Т34;

назвать учеников X Б класса в порядке убывания числа ошибок, сделанных в тесте Т12;

сколько ошибок сделал ученик 10Б28 во всех тестах;

выдать все ответы ЭВМ ученика 09А16;

выдать матрицу ошибок IX В класса в тесте Т27.

Анализ электронного журнала позволит учителю получать полезную информацию о ходе учебного процесса в классах и у отдельных учеников.

И. РОБЕРТ
канд. пед. наук, НИИ ШОТСО АПН СССР

Средства новых информационных технологий — школе

Ускорение научно-технического прогресса поставило перед советской общеобразовательной школой серьезную задачу — активно включиться в качественно новый этап развития, связанный с информатизацией современного общества.

Решение этой задачи (осуществление социального заказа) зависит как от технической оснащенности школ учебной вычислительной техникой с соответствующим периферийным оборудованием, учебным демонстрационным оборудованием, так и от обеспечения психолого-педагогическими и методическими разработками, направленными на выявление оптимальных условий использования средств новых информационных технологий (СНИТ) в практике обучения. Становится также актуальной разработка определенных методических подходов к использованию средств новых информационных технологий для реализации идей развивающего обучения.

Однако анализ отечественного и зарубежного опыта использования компьютера в обучении и, в частности, компьютерных программ в качестве средства обучения констатировал неоправданные ожидания и в ряде случаев серьезные разочарования со стороны педагогов-предметников.

На наш взгляд, это вызвано тем, что на данном этапе компьютер используется, во-первых, для «латания прорех» традиционной методики. При этом педагогическая целесообразность разработанных и используемых программ остается «на совести» авторов, так как единой концепции, предлагающей критерии оценки качества педагогических программных средств, выбора их тематики, не существует; хотя надо отметить и то, что отдельные подходы к этим проблемам имеются [1, 2]. Во-вторых, пользование компьютером в основном происходит вне учебной среды, которая, с одной стороны, должна служить поддержкой процесса преподавания учебного предмета, с другой стороны, амортизировать или компенсировать негативные последствия общения обучаемого с компьютером. Последнее констатируется в ряде исследований психологов и методистов [3, 4, 5, 6], в которых, в частности, отмечается, что при работе с компьютерной программой и в процессе так называемого экранного творчества ученик подменяет объекты реаль-

Материалы статьи можно рекомендовать как дополнение к Концепции информатизации образования при раскрытии методических аспектов использования новых информационных технологий в учебно-воспитательном процессе общеобразовательной школы.— *Прим. автора.*

ного мира либо моделями, изображениями этих объектов, либо символами, обозначающими объекты и отношения между ними. При этом восприятие реального мира подменяется опосредованным восприятием последнего, что может привести к утрате предметности деятельности. Кроме того, как известно, работа с компьютером связана с высоким эмоциональным напряжением, которое не всегда и не каждому может быть полезно.

Учитывая вышеизложенное, компьютер следует рассматривать как компонент системы средств обучения, в состав которой должны входить средства обучения, компенсирующие отсутствие предметной среды, обеспечивающие предметность деятельности и ее практическую направленность (например, учебные роботы, управляемые ЭВМ, электронные конструкторы, средства пространственного ввода и манипулирования текстовой и графической информацией и т. д.), и, кроме того, «традиционные» средства обучения, обеспечивающие поддержку преподавания данного учебного предмета.

Функционирование учебной среды обеспечивается в условиях работы кабинета вычислительной техники, в котором организуется общение обучаемого со СНИТ.

Первые попытки разработать рекомендации по созданию учебной среды для преподавания ОИВТ были предприняты еще в 1985 г. [7, 8]. Несмотря на то что «Перечни» составлялись с позиций комплексного подхода к изучению ОИВТ с использованием системы средств обучения, в них недостаточно учитывалась специфика процесса изучения нового курса с использованием ЭВМ (так называемый машинный вариант) и, кроме того, совершенно не учитывались возможности других информационных технологий. В связи с этим приходится признать необходимость новых концептуальных подходов в этому вопросу.

Учебно-методический комплекс, использующий средства новых информационных технологий

Включение компьютера в учебный процесс изменяет роль средств обучения, используемых в процессе преподавания того или иного предмета, а использование средств новых информационных технологий изменяет учебную среду, в которой происходит процесс обучения.

Реализуя возможности современных ЭВМ в области управления различными устройствами и механизмами, можно создавать качественно новые средства обучения, имитирующие разнообразные промышленные механизмы и приспособления, управляемые ЭВМ.

Примером может служить использование учебных роботов, управляемых ЭВМ (например, робот-подъемник, имитирующий промышленные механизмы, осуществляющие погрузочно-разгрузочные работы, или робот-тележка, имитирующий управление движущимися объектами).

Цель использования учебных роботов — демонстрация возможностей современных ЭВМ в сфере управления объектами реальной действительности, обучение составлению программ для управления, профориентация подрастающего поколения.

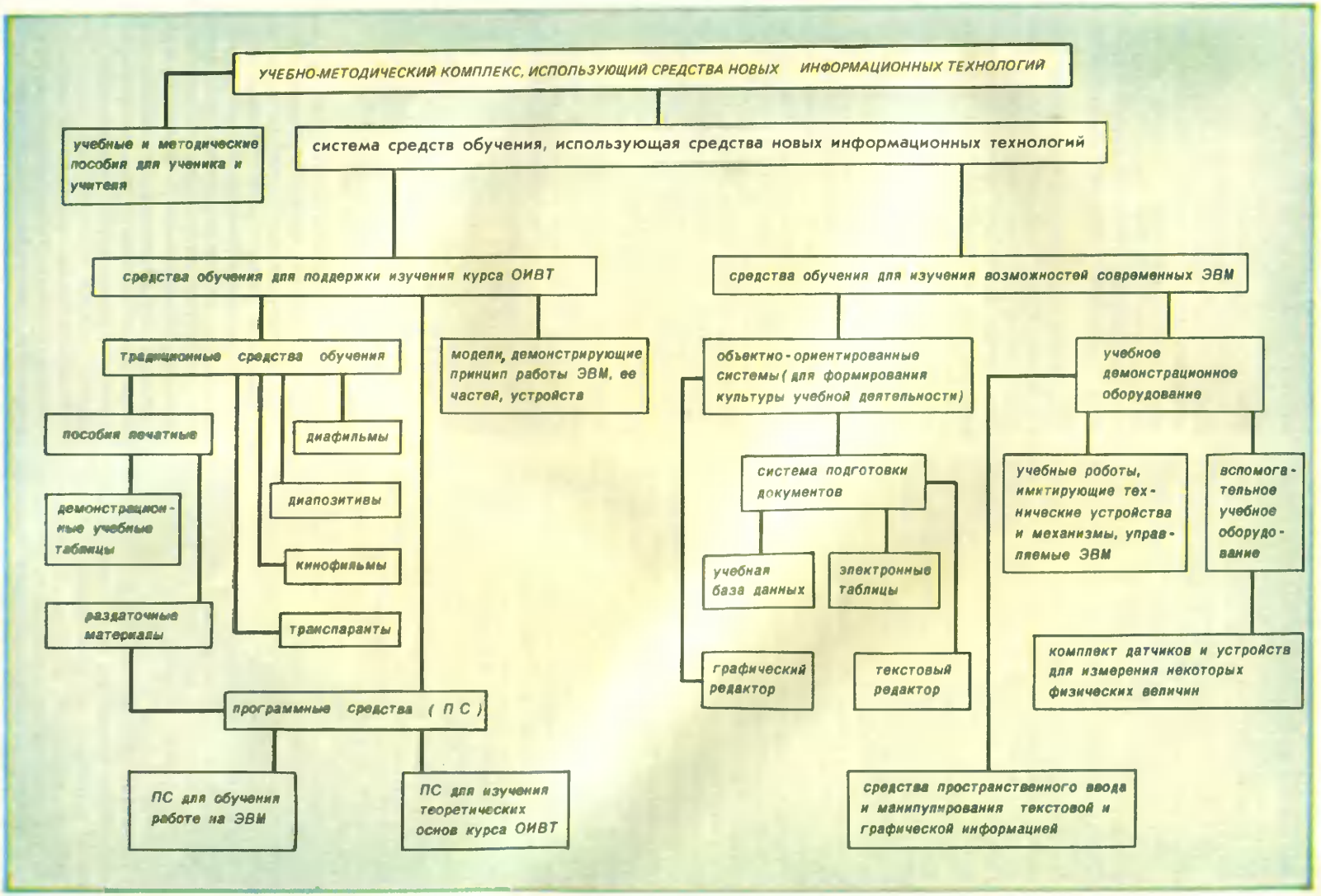
Принципиально новым компонентом учебной деятельности становится работа со средствами пространственного ввода и манипулирования текстовой и графической информацией. К ним можно отнести манипуляторы типа «мышь», «граф-пад», «световое перо». Цель использования этих средств — демонстрация возможностей компьютера в сфере обработки и передачи информации.

Новое направление использования компьютера в учебном процессе открывает применение устройств, подсоединяемых к компьютеру, обеспечивающих ввод и вывод аналоговых и дискретных сигналов, а также датчиков и устройств для измерения некоторых физических величин (например, величины светового потока, температуры, кислотности среды). С помощью таких устройств, подключаемых к ЭВМ, или оборудования на их базе (условно назовем его вспомогательным учебным оборудованием, подключаемым к ЭВМ) становится возможна визуализация различных физических закономерностей в виде графиков, динамически изменяющихся в зависимости от изменения входных параметров.

Применение вспомогательного оборудования позволит учащимся создавать модели изучаемых процессов, проигрывать поведение, развитие модели при различных условиях; прогнозировать развитие процессов и осуществлять с помощью компьютера проверку достоверности прогноза. Становится возможна автоматизация школьного эксперимента; проведение на исследовательском уровне лабораторных и демонстрационных экспериментов по основам наук; изучение развития процессов, протекающих в реальной жизни.

Это позволит перевести процесс обучения с уровня «сообщение суммы знаний — усвоение суммы знаний» на уровень «исследовательский подход и прогнозирование», что позволит обучать «открытию» закономерностей изучаемой науки, развивать мышление, актуализировать процесс усвоения нового учебного материала.

Таким образом, с помощью СНИТ стано-


УЧЕБНО-МЕТОДИЧЕСКИЙ КОМПЛЕКС, ИСПОЛЬЗУЮЩИЙ СРЕДСТВА НОВЫХ ИНФОРМАЦИОННЫХ ТЕХНОЛОГИЙ
учебные и методические пособия для ученика и учителя
система средств обучения, использующая средства новых информационных технологий
средства обучения для поддержки изучения курса ОИВТ
средства обучения для изучения возможностей современных ЭВМ
традиционные средства обучения
модели, демонстрирующие принцип работы ЭВМ, ее частей, устройств
пособия печатные
диафильмы
диапозитивы
кинофильмы
транспаранты
демонстрационные учебные таблицы
раздаточные материалы
программные средства (П С)
ПС для обучения работе на ЭВМ
ПС для изучения теоретических основ курса ОИВТ
объектно-ориентированные системы (для формирования культуры учебной деятельности)
система подготовки документов
учебная база данных
электронные таблицы
графический редактор
текстовый редактор
учебное демонстрационное оборудование
учебные роботы, имитирующие технические устройства и механизмы, управляемые ЭВМ
вспомогательное учебное оборудование
комплект датчиков и устройств для измерения некоторых физических величин
средства пространственного ввода и манипулирования текстовой и графической информацией

вится реальным введение в процесс обучения принципиально нового учебного демонстрационного и вспомогательного оборудования, предоставляющего учащимся следующие возможности:

управлять с помощью ЭВМ объектами реальной действительности;

визуализировать физические закономерности на экране ЭВМ, используя специальные датчики, подключаемыми к ЭВМ (например, акустический или фотометрический датчик);

демонстрировать графические возможности современных ЭВМ, конструируя на экране разнообразные графические формы (например, с помощью графического планшета).

Итак, вырисовывается определенная система средств обучения, ориентированная на использование средств новых информационных технологий, применение которой создает условия обучения тому или иному учебному предмету в среде, обеспечивающей психолого-педагогические и научные требования к оптимальному педагогическому процессу.

Такая система средств обучения совместно с учебно-методической литературой и программным обеспечением учебного курса составит учебно-методический комплекс (УМК), использующий СНИТ.

Определение соотношения составляющих компонентов УМК — дело будущего, как и уточнение его состава (ориентировочный состав УМК 1-го поколения представлен на таблице). Варьируя состав и комплектность, им можно пользоваться в процессе преподавания информатики, физики, химии, биологии, а также интегрированных курсов.

Исходя из вышеизложенного можно предложить систему средств обучения для изучения курса информатики и вычислительной техники, котсрая, с одной стороны, должна демонстрировать возможности современных ЭВМ, с другой стороны, осуществлять поддержку процесса преподавания курса.

Эта система средств обучения совместно с учебно-методической литературой и программным обеспечением составит учебно-методический комплекс (УМК), использующий СНИТ, для изучения курса информатики и вычислительной техники.

Программное обеспечение курса информатики, со своей стороны, должно быть ориентировано, во-первых, на поддержку изучения курса (изучение теоретических вопросов, выработка умений и навыков общения с ЭВМ), во-вторых, на обеспечение управления учебным процессом, автоматизацию контроля, в-третьих, на формирование специфических умений и навыков, повышающих культуру учебной деятельности, способствующих общему развитию учащихся (умение работать с текстовым редактором, информационно-поисковыми системами, учебными электронными таблицами, различными графическими редакторами).

К учебному демонстрационному оборудованию, осуществляющему поддержку изучения курса ОИВТ, можно отнести также модели, демонстрирующие принцип работы ЭВМ, ее устройств. Это облегчает учащимся усвоение разделов ОИВТ, связанных с изучением структуры ЭВМ и принципов ее работы.

Не обойтись на уроках информатики и без учебно-наглядных пособий — демонстрационных таблиц (например, серия таблиц, представляющая известных ученых в области информатики и вычислительной техники, позволит пополнить знания учащихся историческими сведениями), диапозитивов, диафильмов (например, «Применение новых информационных технологий в народном хозяйстве»), транспарантов.

Необходимость использования так называемых традиционных средств обучения обусловлена их специфическими функциями, которые передать компьютеру либо невозможно, либо нецелесообразно с педагогической или гигиенической точки зрения. Например, демонстрацию статической информации, представляемой учащимся для запоминания теоретических положений, а также систематизированные сведения, справочные данные, которые ученик должен запомнить, следует предъявлять в виде учебных таблиц, схем, которые являются демонстрационными средствами обучения. Систематически, из урока в урок наблюдая демонстрируемый таблицей материал, ученик непроизвольно заучивает его, не тратя на это специального времени. Естественно, что компьютер в этом случае неприемлем. Если же справочный материал не подлежит запоминанию и нужен для кратковременного использования, его целесообразно вызывать на экран с помощью специальной программы или пользоваться информационно-поисковой системой.

Учитывая то обстоятельство, что выход учащихся на ЭВМ следует предварительно подготавливать, чтобы машинное время не тратилось на разного рода прикидки, целесообразно пользоваться специально разработанными для этой цели сериями транспарантов для графопроектора, диапозитивов (например, серия транспарантов «Работа на ЭВМ с графической информацией» подготавливает учащихся к работе над программами, строящими простейшие геометрические фигуры).

Подготавливая программное обеспечение и средства обучения для каждого урока или темы, необходимо стремиться к тому, чтобы

ученик выполнял на ЭВМ ту работу, которую с помощью других средств обучения он выполнить не сможет. Поэтому целесообразно разрабатывать комплексы, обеспечивающие каждый урок или тему всеми необходимыми средствами обучения.

Перспективы использования средств новых информационных технологий в учебном процессе

Перспективным направлением использования средств новых информационных технологий в целях обучения является интеграция возможностей компьютера и различных средств передачи аудиовизуальной информации. Реализуется это, в частности, в так называемых интерактивных видеосистемах, позволяющих обогатить учебный процесс следующими возможностями:

обеспечение разнообразных путей доступа к библиотеке движущихся и неподвижных изображений со звуковым сопровождением или без него;

выбор в любой последовательности из базы данных необходимой на данном этапе аудиовизуальной информации;

использование библиотеки программ;

контаминация (смешение, перетасовка) информации, включающей текстовую, графическую, подвижные диаграммы, мультипликацию со звуковым сопровождением или без него;

«оконное» представление информации.

Все это многообразие возможностей способствует развитию у обучаемых творческой инициативы в процессе приобретения новых знаний, развивает умение извлекать необходимую на данном этапе информацию, манипулировать ею, анализировать и систематизировать ее. При таких подходах ознакомление учащихся с новой для них информацией, формирование новых представлений и понятий происходит на совершенно ином уровне, как мотивационном, так и развивающем.

Естественно, что использование интерактивных видеосистем предполагает принципиально новый уровень организации учебного процесса в учебной среде, обеспечивающей применение широкого спектра средств новых информационных технологий.

Рабочее место ученика и учителя, оснащенное интерактивной видеосистемой, должно обеспечивать возможность доступа к различным источникам аудиовизуальной информации. Объединение рабочих мест учащихся в сеть должно обеспечивать возможность обработки результатов обучения и тестирования (с целью объективного видения самим учеником результатов продвижения в учении), проведения индивидуальной, парной, групповой и коллективной работы.

Сам по себе процесс внедрения средств новых информационных технологий немыслим без средств телекоммуникаций на уровне синтеза компьютерных сетей и средств телефонной, телевизионной, спутниковой связи. Такие комплексы образуют системы передачи/приема информации региональных масштабов [9]. При этом общение через компьютерные сети (локальные и глобальные) позволяет производить обмен символьной, графической информацией в виде запросов пользователем и получения им ответов из информационного банка центральной ЭВМ. Реализуемая таким образом непрерывность общения пользователя с информационным банком центральной ЭВМ способствует оптимальному использованию информации, включающей и учебную, которая по желанию разработчиков может быть представлена в виде обучающих систем.

Телекоммуникационные связи могут осуществляться как в реальном времени, по телефонной сети (так называемая синхронная телекоммуникационная связь), так и с задержкой по времени с помощью электронной почты (асинхронная телекоммуникационная связь). Общение с помощью телекоммуникационных сетей позволяет организовать пользование совместными информационными ресурсами, способствует развитию у учащихся умений в сжатой форме выражать свои мысли, отсортировывать по определенным признакам необходимую информацию и, что самое главное, вырабатывает у учащихся способность к коммуникативным действиям.

Таким образом, использование телекоммуникационных сетей позволяет в кратчайшие сроки тиражировать передовые педагогические технологии и вместе с тем способствует общему развитию обучаемого.

Внедрение средств новых информационных технологий в учебный процесс неизбежно влечет за собой введение новой педагогической технологии обучения. Как и в случае любого иного изменения устоявшейся технологии, этот процесс требует определенного «адаптационного» периода для ознакомления с новыми возможностями и психологической «притирки». Важно, однако, не растягивать на годы этот процесс, а планомерно вводить новую педагогическую технологию обучения, использующую разнообразные возможности средств новых информационных технологий.

Литература

1. *Мирская А., Сергеева Т.* Обучающие программы оценивает практика // Информатика и образование. 1987. № 6.

2. *Роберт И.* Какой должна быть обучающая программа? // Информатика и образование. 1986. № 2.

3. *Зинченко В. П.* Эргономика и информатика // Вопросы философии. 1986. № 7.

4. *Роберт И. В.* Интегрированная система средств обучения курсу «Основы информатики и вычислительной техники» // Сб. материалов IV Всесоюзного семинара «Разработка и применение программных средств ПЭВМ в учебном процессе». М., Ротапринт ИПИ АН СССР. 1988.

5. Социальные и методологические проблемы информатики, вычислительной техники и средств автоматизации: Материалы «круглого стола». Ч. 1 // Вопросы философии. 1986. № 9.

6. *Тихомиров О. К.* Информатика и новые проблемы психологической науки // Вопросы философии. 1986. № 7.

7. Кабинет вычислительной техники всех типов средних учебных заведений (на базе персональных микро-ЭВМ): Методические рекомендации. М., Ротапринт НИИ ШОТСО АПН СССР. 1986.

8. Перечни технических средств, учебно-наглядных пособий и мебели для кабинетов вычислительной техники всех типов средних учебных заведений. М.: Просвещение, 1985.

9. *Полат Е. С.* Проблемы использования компьютеров в системе образования развитых капиталистических стран // Информатика и образование. 1987. № 3.

Г. АРУТЮНЯН, В. ЛАШУК, А. НОВОДВОРСКИЙ, А. СМИРНОВ, А. ФОКИН

Система «Лектор»

Существующие практика использования КУВТ и программная поддержка локальной сети ориентированы в основном на самостоятельную работу обучающихся. Преподаватель в этой среде зачастую превращается в «заспинного» консультанта. Напрягая голосовые связки, он пытается привлечь к себе внимание, чтобы направить ход учебного процесса в должное русло, указать типичные ошибки и т. п. Затем он совершает «забег» по классу с целью реализовать свой план для всех вместе и каждого в отдельности.

По всем меркам, включая опыт педагогической практики и эргономические нормы, это шаг назад от доски с мелом (которые отсутствуют в классах с КУВТ по известным причинам). Такие занятия не укладываются в традиции отечественной педагогики, в соответствии с которыми преподаватель — ведущее звено учебного процесса и не может быть заменен даже самой совершенной техникой.

Сложившееся положение послужило толчком для создания такого программного обеспечения локальной сети КУВТ, которое поставило бы учебный процесс «с головы на ноги», вернуло центральное место в педагогическом процессе преподавателю.

Локальная сеть КУВТ должна позволять полностью управлять ходом занятия с центрального компьютера. Для этого, с нашей точки зрения, соответствующие программные средства должны обеспечить:

регистрацию на рабочих местах;

поддержку сетевых протоколов контролирующих программ;

возможность получения твердой копии протокола занятия (идентификатор занятия, список группы, оценки и баллы контролирующих программ и т. д.);

обмен между компьютерами учеников (только с разрешения преподавателя);

возможность просмотра содержимого экрана ученика на рабочем месте преподавателя с внесением необходимых коррективов;

передачу содержимого экрана учителя либо файла с графическим содержанием на экраны учеников в текстовом и графических режимах;

возможность блокировки клавиатуры учеников при необходимости (например, чтобы сделать организационное сообщение) и т. д.

Полная реализация перечисленных функций в одном мониторе была бы излишне громоздкой. Да и использование такой универсальной системы представляло бы сложность для рядового преподавателя-предметника. А хотелось бы уже сейчас видеть компьютер не только как предмет изучения, но как мощное техническое средство обучения в преподавании самого широкого спектра предметов.

Поэтому для реализации нами было избрано функционально узконаправленное подмножество указанных выше требований. Было решено реализовать монитор, функционально поддерживающий занятия типа лекции.

Итак, «Лектор».

Его опорная идея — возможность чередования изложения предмета, иллюстрируемого наглядным графическим и текстовым материалом, с короткими, 2—3-минутными, тренажерными вставками.

Полифункциональность такого чередования очевидна. Это и закрепление материала, и активное сотворчество в образовании, и способ сменить форму деятельности — стряхнуть привычную (увы) лекционную дремоту аудитории.

Что касается эргономичности, то захватить на лекцию пару десятков «плакатов» на дискете гораздо проще, чем тащить в аудиторию груду истрепанного ватмана. Подготовить графический материал поможет наличие графических редакторов на всех типах КУВТ. А 2—3-минутные тренажеры доставят радость творчества любому предметнику, владеющему азами программирования в среде одного из доступных интерпретаторов — Бейсика или Фокала.

Высказанные выше идеи были реализованы на КУВТ «Ямаха» и КУВТ-86 (Фокал).

Каждый из классов имеет свои особенности аппаратного и программного плана, однако общая идейная основа обусловила функциональную близость обоих пакетов. В число их возможностей входят:

демонстрация на мониторах обучаемых учебного материала, оформленного в виде плакатов с использованием текста, графики и цвета;

использование электронной указки при комментировании плакатов;

динамическое (по ходу комментариев) дополнение «плаката» необходимой информацией — «метафора доски»;

динамическая смена демонстрации «плакатов» работой с обучающими программами, тренажерами и т. д.;

возможность блокирования работы компьютеров с целью обратить внимание обучаемых на узловые моменты изучаемого материала, типичные ошибки и т. д.

Как показала опытная эксплуатация в ЦИУУ, Московском ГИУУ и Владимирском областном ИУУ, «Лектор» весьма прост в обращении, значительно увеличивает эффективность работы преподавателя по дидактическим, эргономическим и временны́м критериям.

Пакет заблаговременно приготовленных плакатов и тренажеров, объединенных тематикой лекции, помещается на диск. Последовательный список имен файлов с плакатами и тренажерами заносится в служебный файл-«ролик» системы «Лектор». Это позволяет переходить от плаката к плакату, от плаката к тренажеру простым нажатием клавиши преподавательского компьютера.

Впрочем, от услуг файла-«ролика» можно отказаться. В этом случае выбор последовательности подачи материала становится гибким, но имя каждого следующего файла преподавателю придется набирать на клавиатуре.

В целом система «Лектор» поддерживает с пользователем дружественный интерфейс. Используется система подсказок для функций «Лектора», а протокол работы с учебным материалом выводится на монитор преподавателя. В то же время алгоритм системы и достаточно сложный сетевой протокол остаются совершенно прозрачными для пользователя.

Есть в разработке и удачные программные находки. Так, «Лектор» для КУВТ «Ямаха» содержит систему автоматического исправления сетевых сбоев, а «Лектор» для КУВТ-86 выгодно отличается от существующих передатчиков возможностью запуска программ на Фокале на РМУ, что снимает досадные задержки и неувязки учебного процесса, связанные с временны́ми потерями.

В скором времени обе версии системы «Лектор» будут распространяться централизованно.

В целом система «Лектор» является мощным инструментальным средством педагога, позволяющим поддерживать занятия разного предметного наполнения. Предлагая разработку вниманию читателей ИНФО, мы рассчитываем найти единомышленников среди преподавателей самых различных дисциплин.

Как и для любого инструмента, эффективность «Лектора» прежде всего определяется педагогическим мастерством его пользователя. В этой области, на наш взгляд, «Лектор» открывает широкие перспективы.

Авторы будут благодарны как за конструктивные предложения, так и за конструктивную критику.

Уважаемые читатели!
Конкурс программ по игре КТОР,
объявленный
в шестом номере нашего журнала за 1988 г.,
продлевается до января 1990 г.
Заявки принимаются до 1 ноября 1989 г.
ленинградским КТОР-клубом.

А. ФОМЕНКО, М. ГУЛЯКОВИЧ, В. МАЛАНИНА

Обучающая система «Страус»

В Московском институте электронной техники разработана обучающая система «Страус». Она позволяет использовать компьютер в качестве помощника учителю для закрепления знаний у школьников при изучении различных предметов школьной программы.

Основные принципы, которые заложены в алгоритм системы,— это простота работы за дисплеем ЭВМ и наглядность информации. Даже ученики младших классов сумеют выполнять директивы компьютера, на которые указывает маркер (светящийся квадратик) на экране дисплея.

В систему «Страус» входят следующие режимы:

«Теория»,
«Тренажер»,
«Экзаменатор»,
«Сводка о работавших»,
«Статистика ответов».

Директива компьютера о запросе работы в этих режимах отображается на экране дисплея в виде кадра 1.

Программно-методическое обеспечение «Страус» как обучающей системы имеет два основных аспекта: это обучение, которое обеспечивается режимами «Теория» и «Тренажер», и самоконтроль обучаемого или контроль состояния знаний учащегося со стороны учителя, который ведется в режиме «Экзаменатор».

Режим «Теория» помогает напомнить обучаемому правила с экрана дисплея, на которые будут предложены упражнения либо в режиме «Тренажер», либо в режиме «Экзаменатор». Переход от правила к правилу ученик осуществляет нажатием клавиши ввода, тем самым он регулирует время чтения каждого правила по своему усмотрению: ускоряя предъявление ему следующего текста или замедляя смену кадра.

МИЭТ — НИИ ШКОЛ
(предмет, класс, наименование темы) «СТРАУС»

∅ — ТЕОРИЯ
1 — ТРЕНАЖЕР
2 — СВОДКА О РАБОТАВШИХ
3 — СТАТИСТИКА ОТВЕТОВ
4 — ЭКЗАМЕНАТОР

УКАЖИТЕ НОМЕР РЕЖИМА РАБОТЫ

Кадр 1 ⊞ маркер

Режим «Тренажер» обеспечивает закрепление знаний учащихся, полученных на уроках, и самостоятельное изучение некоторых тем. Упражнения на экране дисплея выдаются по датчику случайного числа с выбыванием их из банка заданий при правильном ответе. При неправильном ответе учащегося система выдает верный ответ и соответствующее правило, что позволяет лучше закрепить данный материал. Работа на ЭВМ сопровождается регистрацией верных и неверных ответов. По окончании работы система «Страус» выставляет отметку и высвечивает упражнения, которые были выполнены неверно, с правильными ответами, чтобы ученик еще раз мог закрепить данный материал.

Работа в режиме «Экзаменатор» аналогична работе в режиме «Тренажер» с той лишь разницей, что в данном режиме после ввода ответа учеником компьютер не выдает информацию о верности ответа, а фиксирует ее только в своей памяти.

Все обучающие курсы в данных режимах построены по следующим алгоритмам:
зрительного восприятия;
статически конструируемого ответа;
динамически конструируемого ответа;
работы с пиктограммами.

Суть алгоритма зрительного восприятия заключается в том, что учащийся, увидев появившуюся на экране дисплея информацию для запоминания, должен зафиксировать ее в своей памяти и после исчезновения ее с экрана повторить эту информацию, набирая на клавиатуре дисплея.

Суть алгоритма статически конструируемого ответа состоит в том, что учащийся строит ответ в естественном виде на том месте, где стоит маркер.

При динамически конструируемом ответе учащийся перемещает маркер в требуемое место и затем вводит ответ в естественном виде.

Суть алгоритма пиктограмм сводится к тому, что учащийся для ввода ответа в машину указывает номера пиктограмм, нарисованный на экране дисплея. В качестве пиктограммы могут выступать графические символы составления блок-схем программ согласно Единой Системе Программной Документации (ЕСПД) или графические обозначения членов предложения при их синтаксическом разборе и т. д. и т. п.

Описанные три режима работы системы «Страус» предназначены для учащихся. Сле-

дующие режимы из меню системы предназначены для учителя.

Работая в режиме «Сводка о работавших», учитель может получить информацию о работавших учениках как на уроке, так и на внеурочных занятиях. Сводка включает следующие сведения: фамилию ученика и класс, режим, в котором работал ученик, номер урока или четверти, количество предъявленных ему упражнений и количество, на которое он верно ответил, отметку (кадр 2).

Режим «Статистика ответов» предназначен для выявления пробелов в знании учащихся при работе в режимах «Тренажер» или «Экзаменатор». Система «Страус» в своем банке заданий имеет до 100 упражнений по каждой теме. Каждое упражнение имеет свой порядковый номер. При работе в данном режиме на экране появляется таблица (кадр 3). На пересечении строк и столбцов, которые соответствуют номеру упражнения, напечатаны сведения о том, сколько раз данное упражнение было выдано ученикам для выполнения (верхняя строчка) и сколько раз ученики ответили неправильно (нижняя строчка). Учитель, проанализировав таблицу, сможет выявить, на какие правила допущено большое число ошибок, и на уроке, методически более доходчиво, еще раз разобрать данное упражнение.

```
ТИТОВ Х — А
РЕЖИМ 1, УРОК 2, ПРИМЕРОВ 16, ИЗ НИХ
ВЕРНЫХ 15, ОТМЕТКА 4
ИВАНОВ Х — Б
РЕЖИМ 1, УРОК 1, ПРИМЕРОВ 10, ИЗ НИХ
ВЕРНЫХ 10, ОТМЕТКА 5
ЕГОРОВА IV — В
РЕЖИМ 0, УРОК 3, ПРИМЕРОВ 0, ИЗ НИХ
ВЕРНЫХ 0, ОТМЕТКА 0

НАЖМИТЕ ВК
```

Кадр 2

АНАЛИЗ ОТВЕТОВ ПО ТЕМЕ:
ТРУДНОЗАПОМИНАЕМЫЕ СЛОВА

СЛОВА В СЛОВАРЕ		0	1	2	3	4	5	6	7	8	9
А	00	0	1	0	0	3	0	0	0	0	0
В		0	0	0	0	0	0	0	0	0	0
А	10	1	6	0	0	0	0	0	1	0	0
В		0	4	0	0	0	0	0	1	0	0
А	20	0	0	0	0	0	0	0	0	0	0
В		0	0	0	0	0	0	0	0	0	0
А	30	0	0	2	0	0	1	0	0	0	0
В		0	0	1	0	0	0	0	0	0	0
А	40	0	0	0	1	0	0	7	0	0	0
В		0	0	0	0	0	0	1	0	0	0

СТРОКА А — ЧИСЛО ОБРАЩЕНИЙ К СЛОВУ; В — ЧИСЛО НЕПРАВИЛЬНЫХ ОТВЕТОВ
КОНЕЦ РАБОТЫ — ГТ ПРОДОЛЖЕНИЕ РАБОТЫ — ВК

Кадр 3

Учебно-методические материалы для создания компьютерных уроков на базе описываемой обучающей системы разработаны совместно в сотрудниками НИИ школ МНО РСФСР С. Грицаевой, М. Рыбаковым и Г. Стебеневой.

Таким образом, разработанная система позволяет:

ознакомить учащихся с возможностями вычислительной техники;

сформировать важнейшие навыки работы с ЭВМ;

преодолеть психологический барьер в общении с компьютером;

закрепить знания, полученные на уроке, тем самым повысить эффективность урока;

освободить учащихся от домашних работ;

апробировать методику изложения материала учителем.

Данная система передана в школы, оснащенные КУВТ на базе ДВК или БК-0010Ш.

Информация: поводырь или трамплин?

В 70-е — начале 80-х гг. США было все труднее противостоять наступлению японских фирм. Причиной этого было не научное или технологическое отставание, а «всего лишь» менее эффективная организация управления.

Естественно, американцы попытались устранить разрыв. Проблему исследовали многие ученые и практики, на эту тему были написаны десятки книг и сотни статей. Бестселлером стала работа Т. Питерса и Р. Уотермена «В поисках совершенства» (на русском языке издана в 1986 г. издательством «Прогресс» под названием «В поисках эффективного управления»).

Основная идея этой книги такова: чтобы работать успешно, недостаточно своевременно реагировать на изменение условий функционирования. Настоящего успеха может достичь лишь предприятие, само инициирующее эти перемены, не боящееся отказаться от только что достигнутого хорошего ради лучшего.

При чем же здесь информатика?

Чтобы своевременно реагировать на изменения, нужна свежая и полная информация. Но чтобы идти вперед изменений, нужно использовать информацию сверхсвежую и сверхполную! Выводы очевидны.

Е. РАНЬКОВ, А. АБДУКАДЫРОВ

Решение задач с комментариями

Предлагаемая статья построена как методический разбор некоторых задач, включенных в раздел «Упражнения для повторения» второй части курса ОИВТ [4], которые охватывают весь курс и позволяют на этапе повторения строить альтернативные варианты решений с использованием конструкций разной сложности. При таком подходе приводимая методика решения задач не утратит актуальности, как бы ни изменились впоследствии школьные учебники ОИВТ (даже если в них существенно трансформируется изучаемая модель алгоритмического языка).

Следует особо подчеркнуть, что в изучении любого алгоритмического языка в курсе средней школы мы прежде всего видим мощное средство дисциплинирования мышления, одновременно вводящее учащихся в современный мир программирования.

Процесс решения задачи складывается из трех этапов: нахождение способа решения (алгоритма), его изложение и аргументация мотивов выбора конкретной формы реализации решения.

Первый этап — нахождение алгоритма — представляет для учащихся наибольшую трудность, но, к сожалению, в настоящее время он не нашел достаточного отражения в курсе ОИВТ (это и главная задача школьного обучения в целом — научить мышлению, т. е. умению думать). **На втором этапе** решения задачи — изложение алгоритма — алгоритмический язык действительно необходим: его использование требует от учащихся четкого понимания задачи, умения обобщить и формализовать выбранный способ решения, полностью исключив из него неоднозначности и неопределенности. Аргументация мотивов решения **(третий этап)** при использовании алгоритмического языка может быть осуществлена посредством *комментария*, сопровождающего алгоритм (если включить в текст комментария обоснование выбранного способа решения, или же доказательство правильности выбранного алгоритма, или отражение цепочек логических умозаключений учащегося при построении данного алгоритма и т. д.). Кроме того, комментарий обеспечивает возможность легкого «прочтения» алгоритма, без дополнительных разъяснений.

Так, наш **первый принцип:** алгоритмы должны сопровождаться комментарием, построчным или типа резюме, предваряющим или заключающим текст алгоритма. Комментарий должен давать ключ к пониманию алгоритма и того, как можно было до него додуматься.

Наш **второй принцип** состоит в том, что алгоритмы, даже будучи учебными, должны в полной мере отвечать основным требованиям к алгоритмам [5].

Третий принцип: полезность методических разработок заключается не столько в изложении конкретных решений, сколько в анализе процесса их получения, в сопоставлении различных вариантов решения и вскрытии их сравнительных достоинств и недостатков. Рекомендуемый в конечном счете алгоритм должен быть правильным (не содержать ошибок), отвечать всем изложенным выше принципам, быть наглядным и эффективным, а решаемая задача должна быть разумной и выполнимой.

Прежде чем перейти к рассмотрению конкретных задач, остановимся еще на одном частном, но чрезвычайно важном для любого алгоритмического языка вопросе: способах организации циклов или командах повторения.

В учебнике [1, 2] предусмотрены две команды: «пока...» и «для...».

Команда повторения «пока...» вводится в [1] при первом знакомстве учащихся с циклами. Это удобная и простая для восприятия конструкция, в которой в заголовке цикла указываются лишь условия его повторения, а обновление всех параметров цикла (т. е. величин, меняющихся перед каждым его повторением) производится в теле цикла специально предусмотренными для этого командами.

Команда повторения с параметрами «для... от... до...» вводится позднее, в [2], для циклов с одним параметром, меняющимся к тому же лишь по арифметической прогрессии.

В целом конструкция «для...», с одной стороны, беднее, но с другой — нагляднее (все значения параметров четко указаны в заголовке цикла) и в большей мере автоматизирована (не нужно самим заботиться о присвоении параметру очередного значения). Поэтому при наличии параметров, меняющихся регулярно, по арифметической прогрессии, предпочтительнее команда «для...».

Однако на практике часто бывает необходимо организовать цикл с параметром (например, для просмотра таблиц), но прервать его при выполнении некоторого условия (например, если встретилось заданное число). В таких случаях методическое пособие рекомендует использовать конструкцию «пока...» с двумя и более условиями. На наш взгляд, здесь следует сохранить более наглядную и удобную конструкцию «для...», но в теле цикла при выполнении заданного условия присвоить параметру цикла его предельное значение. Это обеспечит «искусственное насыщение» параметра цикла и, как следствие, принудительное прерывание цикла по заданному условию. Данный прием достаточно прост для восприятия учащимися и существенно расширяет их представления об организации циклов. По-видимому, в курсе ОИВТ целесообразно познакомить учащихся (в дополнение к учебнику) с двумя этими приемами на материале первых упражнений (3, 7, 12 и др.) [4], а в дальнейшем следовать выработанным рекомендациям.

Перейдем непосредственно к рассмотрению упражнений. Учитывая ограничения на объем статьи, из них выбраны наиболее интересные. Условия задач приводятся в соответствии с [4]. Следуя изложенным выше принципам, для каждого упражнения принята следующая структура.

Алгоритм МЕТПОС — решение, приведенное в методическом пособии [4], но дополненное нашим построчным комментарием.

КОММЕНТАРИЙ — описание алгоритма, развернутое в меру необходимости, разъясняющее назначение введенных переменных и суть решения.

ЗАМЕЧАНИЯ авторов настоящей работы к соответствующему алгоритму, если таковые имелись.

РЕКОМЕНДАЦИЯ — алгоритм, рекомендуемый авторами в дополнение к МЕТПОС или взамен него.

В отдельных случаях дополнительно к построчному комментарию помещены ПРИМЕЧАНИЯ, в которых подчеркнуты особенности рекомендуемых алгоритмов.

Авторами в соответствии с приведенной методикой рассмотрены все упражнения [1; 2]. Этот материал записан на диск КУВТ «Ямаха-1». Желающие переписать себе эту информацию или получить распечатки могут обратиться по адресу: 700064, Ташкент-64, Педагогическая ул., д. 103. Ташкентский государственный педагогический институт им. Низами, кафедра информатики и вычислительной техники. Абдукадырову А. А., Ранькову Е. А.

Упражнение 9. Дана целочисленная таблица A [1:100]. Подсчитайте, сколько раз встречается в этой таблице максимальное по величине число.

```
алг  метпос_9(цел таб A[1:100],цел количество)
       арг  A
       рез  количество
нач цел i,наибольшее
    наибольшее:=A[1]    'начальное приближение (по A[1])'
    для i от 2 до 100
    нц   'цикл для поиска наибольшего'
      если  A[i]>наибольшее
          то  наибольшее:=A[i]    'корректировка текущего max'
      все
    кц
```

```
    количество:=0     'чистка сумматора'
    для i от 1 до 100
    нц   'цикл для сравнения с наибольшим (поиск равных)'
      если  A[i]=наибольшее
          то  количество:=количество+1
      все
    кц
кон
```

Рекомендация

```
алг  упр.9(цел таб A[1:100],цел количество)
      арг  A
      рез  количество
нач цел i,max
    max:=A[1],количество:=1   'начальное приближение (по A[1])'
    для i от 2 до 100
    нц   'просмотр таблицы с A[2]'
      выбор
            при  A[i]>max : 'найден новый текущий максимум'
                 max:=A[i] ; количество:=1
            при  A[i]=max :  'найден повтор текущего максимума'
                 количество:=количество+1
      все
    кц
кон
```


Примечания.

1. Используется конструкция **выбор**, так как анализируются 2 исхода из 3 возможных; при 2 из 2 была бы эффективнее конструкция **если**.

2. При обнаружении нового *max* отсчет повторов начинается снова (количественно: = 1).

Упражнение 15. Дана целочисленная таблица A[1:1000]. Подсчитайте наибольшее число одинаковых, следующих подряд элементов.

```
алг  метпос_15(цел таб A[1:1000],цел количество
      арг  A
      рез  количество
нач цел i,k     'k-длина текущей серии'
    k:=1;количество:=1     'начальная оценка,учитывающая A[1]'
    для i от 2 до 1000
    нц   'просмотр таблицы,начиная с A[2]'
      если  A[i]<>A[i-1]
          то   'элемент A[i] открывает новую серию'
            если  k>количество
                 то   'завершившаяся серия длиннее прошлых'
                      количество:=k   'корректировка max'
            все
               k:=1    'в новой (текущей) серии пока только A[i]'
          иначе    'элемент A[i] продолжает серию'
            k:=k+1    'текущая серия удлинилась на A[i]'
      все
    кц
кон
```

Комментарий. Назовем последовательность одинаковых элементов таблицы A серией, а число элементов в ней длиной серии K (естественно, что $K_{min}=1$ и $K_{max}=1000$). Задача состоит в определении K_{max} для данной таблицы A. Алгоритм, приведенный выше, за один просмотр таблицы фиксирует начало каждой новой серии ($A[i] \neq A[i-1]$) и в этот момент корректирует значение K_{max} с учетом длины последней (закончившейся) серии. Чтобы начать отсчет длины новой серии, в этот же момент восстанавливается K: = 1.

Замечания.

1. В алгоритме есть ошибка: не учитывается длина последней серии.

2. Алгоритм станет нагляднее, если ветви A [i]$\neq$A [i—1] и A [i]=A [i—1] поменять местами.

Рекомендация

```
алг  упр.15(цел таб A[1:1000],цел количество)
      арг  A
      рез  количество
нач цел i,k
    k:=1;количество:=1
    для i от 2 до 1000
    нц
      если  A[i]=A[i-1]
          то  k:=k+1    'серия продолжается'
          иначе  'серия кончилась и началась новая'
                 если  k>количество
                     то  количество:=k  'корректировка max'
                 все
                  k:=1    'начальная длина новой серии'
      все
    кц
    если  k>количество  'учет последней серии'
        то  количество:=k
    все
кон
```


Упражнение 18. Даны целочисленные таблицы A [1:100], B [1:100], причем A [1] $\leqslant$ $\leqslant$ A[2] $\leqslant$...$\leqslant$ A[100] и B[1] $\leqslant$ B[2] $\leqslant$...$\leqslant$ B[100]. Построить таблицу C [1:200], содержащую все элементы таблиц A и B, в которой C[1] $\leqslant$ C[2] $\leqslant$...$\leqslant$ C[200].

```
алг  метпос_18(цел таб A[1:100],B[1:100],C[1:200])
      арг  A,B
      рез  C
нач цел i,j
-------------------------------------------------------------
    i:=1;j:=1  'начальные значения i,j для просмотра A,B'
    пока i<=100 и j<=100
    нц   'пересылка из A,B в C до завершения хотя бы одной из A,B'
      если  A[i]<B[j]
          то   'пересылается меньшее,т.е.A[i]'
            C[i+j-1]:=A[i]   'пересылка-см.комментарий'
            i:=i+1    'номер следующего элемента из A'
          иначе    'пересылается меньшее,т.е.B[j]'
            C[i+j-1]:=B[j]   'пересылка-см.комментарий'
            j:=j+1    'номер следующего элемента из B'
      все
    кц   'конец пока при (i=101,j<=100) или (i<=100,j=101)'
-------------------------------------------------------------
    пока i<=100     'таблица B исчерпана в первом цикле'
    нц    'досылка остатка A,если он есть'
      C[100+i]:=A[i]    'пересылка-см.комментарий'
      i:=i+1   'наращивание i для следующего элемента A'
    кц
-------------------------------------------------------------
    пока j<=100     'таблица A исчерпана в первом цикле'
    нц    'досылка остатка B,если он есть'
      C[100+j]:=B[j]    'пересылка-см.комментарий'
      j:=j+1   'наращивание j для следующего элемента B'
    кц
-------------------------------------------------------------
кон
```

Комментарий. Алгоритм реализован в виде трех последовательных циклов, осуществляющих пересылку в С попеременно из обеих таблиц А и В с предварительным сравнением значений их текущих элементов (первый цикл) и вслед за ней — досылку остатка либо из таблицы А (второй цикл), либо из таблицы В (третий цикл). На практике всегда работают только два из этих циклов: первый и второй (когда $A[100] \geqslant B[100]$ или первый и третий (когда $A[100] < B[100]$).

В общем случае можно считать, что пересылается один из элементов $A[i]$ либо $B[j]$. К этому моменту в таблицу С пересланы $(i-1)$ элементов А и $(j-1)$ элементов В, т. е. всего $(i+j-2)$ значений. Поэтому очередное значение должно засылаться в $C[i+j-1]$. В первом цикле этот «адрес» присутствует в явном виде, а во втором и третьем он трансформировался в $C[100+i]$ либо $C[100+j]$ соответственно, так как для второго цикла $(i+j-1)=(i+101-1)=(i+100)$, а для третьего $(i+j-1)=(101+j-1)=(100+j)$.

Все циклы организованы с помощью команды **пока**. Изменение (наращивание) индексов для выборки данных из таблиц А, В реализовано в явном виде: как только какое-либо данное $A[i]$ или $B[j]$ пересылается в С, значение соответствующего индекса увеличивается на 1 ($i:=i+1$ или $j:=j+1$ соответственно). Значение второго индекса при этом остается без изменения, так что в следующем такте цикла одно из чисел будет анализироваться повторно (в первом цикле).

Замечания.

1. Желательно, чтобы значения индексов в процессе работы алгоритма (программы) означали реальные номера уже обработанных элементов (или пересылаемых в данный момент). Тогда адрес пересылаемого значения в таблице С всегда был бы $C[i+j]$.

2. Алгоритм существенно проигрывает из-за необходимости включения в него второго и третьего циклов. Этого легко избежать, если в каждую из входных таблиц добавить по одному элементу, взяв в качестве него число заведомо большее всех данных А, В. Тогда достаточно одного первого цикла (непересланными останутся добавленные два элемента).

3. Цикл для пересылки (с учетом п. 2) лучше организовать без команды пока, взяв в качестве параметра цикла индекс «заполняемого» элемента таблицы С,— такая запись более наглядна и натуральна.

Рекомендация

```
алг  упр.18(цел таб A[1:101],B[1:101],C[1:200])
       арг  A,B
       рез  C
нач цел i,j,k
    если  A[100]<B[100]
        то    'максимум из A,B равен B[100]'
          A[101]:=B[100]+1
        иначе     'максимум из A,B равен A[100]'
          B[101]:=A[100]+1
    все
      i:=0; j:=0     'счетчики элементов,выбранных из A,B'
    для k от 1 до 200
    нц     'цикл для поэлементного заполнения таблицы C'
        если  A[i+1]<B[j+1]     'сравнение 'претендентов''
            то    'пересылается A[i+1]'
              i:=i+1 'наращивание индекса'
              C[k]:=A[i]    'пересылка'
            иначе     'пересылается B[j+1]'
              j:=j+1  'наращивание индекса'
              C[k]:=B[j]    'пересылка'
        все     'востановилось соотношение:k=i+j'
    кц     'при выходе из цикла i=100,j=100,k=201'
кон
```

Примечание. Первый оператор: если $A[100]<B[100]$..., содержащий два условных присваивания, эквивалентен двум безусловным операторам присваивания:

$A[101]:=B[100]+1$

$B[101]:=A[100]+1$

Упражнение 24. Напечатать в порядке возрастания первые 1000 чисел, которые не имеют простых делителей, кроме 2, 3 и 5.

```
алг  метпос_24
нач цел i,k,количество
     печать(1)    'число 1 автоматически включено в искомый список'
     количество:=1    'текущая длина списка (не более 1000)'
     i:=2     'текущее целое,подлежащее анализу'
     пока количество<=1000
     нц  'обработка числа i:проверка делителей и,возможно,печать'
       k:=i    'рабочая копия числа'
       пока k/2=целч(k/2)
       нц  'исключение делителя 2,если таковой есть'
         k:=k/2
       кц
       пока k/3=целч(k/3)
       нц  'исключение делителя 3,если таковой есть'
         k:=k/3
       кц
       пока k/5=целч(k/5)
       нц  'исключение делителя 5,если таковой есть'
         k:=k/5
       кц
       если  k=1
         то   'других простых делителей нет,т.е.i-искомое'
           печать(i)
           количество:=количество+1    'список удлинился на 1'
       все
         i:=i+1    'наращивание анализируемого целого'
     кц   'завершение цикла при условии:количество=1001'
кон
```

Комментарий. Алгоритм построен в виде трех последовательно «работающих» однотипных циклов, использующих конструкцию пока. В каждом из этих циклов повторяется деление нацело (на 2, 3 или 5 соответственно), пока оно возможно.

По завершении алгоритма переменная i содержит целое (натуральное) число, следующее за последним элементом напечатанного списка; переменная «количество» содержит число 1001, а переменная $k=1$.

Замечания.

1. Используется функция целч, отсутствующая в библиотеке алгоритмов (выделение целой части).
2. Использование трех совершенно однотипных конструкций в алгоритме не оправданно (делает его громоздким). Это еще более наглядно, если представить, что список «разрешенных» простых делителей существенно длиннее.
3. Число «1» здесь обрабатывается вне цикла, хотя в этом нет необходимости.

Рекомендация.

1. Предлагается подойти к решению так, чтобы избежать трех однотипных конструкций (см. упр. 24—1) и при этом чтобы перечень «разрешенных» делителей был сколь угодно длинным (см. упр. 24—2).
2. Вместо функции целч целесообразно использовать функцию остаток .
3. Обработку числа «1» лучше внести в общий цикл (алгоритм станет «стройнее»).
4. Наращивание числа i целесообразнее проводить в начале цикла, а не в конце его (тогда при выходе из цикла в i сохранится последнее проанализированное число, что бывает существенно в более сложных алгоритмах).

```
алг  упр.24-1.
нач цел i,k,j,количество,цел таб D[1:3]

     D[1]:=2;D[2]:=3;D[3]:=5    'разрешенные делители
                                 организованы в массив D[1:3]'
```

```
    количество:=0;i:=0     'подготовка к просмотру ряда
                           натуральных чисел'
    пока количество<1000
    нц  'анализ очередного натурального числа'
      i:=i+1    'переход к очередному члену натурального ряда'
      k:=i      'рабочая копия числа i'
      для j от 1 до 3
      нц    'проверка очередного делителя'
        пока остаток(k,D[j])=0
        нц
          k:=k/D[j]
        кц
      кц    'выход из цикла при j=4'
      если  k=1
          то  печать(i)  'число i-искомое'
            количество:=количество+1
      все
    кц    'выход из цикла при значениях переменных
          количество=1000,k=1,i=последнему напечатанному'
кон


алг  упр.24-2(цел n,цел тав D[1:n],A[1:1000])

    арг n,D 'n-число разрешенных делителей;D-их список'
    рез A 'список искомых натуральных чисел (вместо печати)'
нач цел i,k,j,количество
    количество:=0;i:=0     'подготовка к просмотру ряда
                           натуральных чисел'
    пока количество<1000
    нц  'анализ очередного натурального числа
         (следующего за i)'
      i:=i+1    'переход к очередному члену натурального ряда'
      k:=i      'рабочая копия числа i'
      для j от 1 до n
      нц   'цикл для последовательного исключения
           всех разрешенных делителей'
        пока остаток(k,D[j])=0
        нц   'многократное исключение D[j]'
          k:=k/D[j]
        кц
      кц   'выход из цикла после анализа всех
           n делителей,при j=n+1'
      если  k=1
          то  'число i-искомое'
            количество:=количество+1   'порядковый номер
                                        найденого искомого'
            A[количество]:=i  'занесение в выходной список'
      все
    кц
кон
```

Примечание. Задача упр. 24—2 состоит в том, чтобы сформировать возрастающую последовательность первых 1000 натуральных чисел, не имеющих простых делителей, кроме перечисленных в таблице D[1:*n*].

Упражнение 31. Даны две литерные величины A и B. Проверьте, можно ли из букв, входящих в A, составить B (буквы можно переставлять, но каждую букву можно использовать не более одного раза).

```
алг  метпос_31(лит A,B,ответ)
арг  A,B
рез  ответ
нач цел i,j,лит T,признак
  T:=A   'создание рабочей копии слова A'
  ответ:='да';i:=1     'подготовка цикла для просмотра B'
  пока i<=длин(B) и ответ='да'
  нц  'анализ i-ой буквы слова B:есть ли она в T?'
    признак:='нет';j:=1  'подготовка к просмотру T'
    пока j<=длин(T) и признак='нет'
    нц  'проверка j-ой буквы T-слова'
      если T[j:j]=B[i:i]
          то  'нужная буква в T есть'
            T:=T[1:j-1]+T[j+1:длин(T)]  'исключение j-ой буквы'
            признак:='есть'  'дальнейший просмотр T не нужен'
      все
        j:=j+1   'наращивание j для анализа следующей буквы в T'
    кц   'поиск i буквы B-слова в T завершен'
    если признак='нет'
        то  'i-ая буква B-слова в T не найдена'
          ответ:='нет'  'дальнейший анализ B не нужен'
    все
      i:=i+1   'наращивание i для перехода к следующей букве'

  кц
кон
```

Комментарий. Идея алгоритма проста: очередная буква B-слова отыскивается в A: если поиск успешен, переходим к следующей букве в B; если нет — решение задачи прекращается с отрицательным ответом (остальные буквы B анализировать не имеет смысла). Чтобы исключить возможность повторного использования букв из A, каждая найденная буква тотчас выбрасывается из слова A, так чтобы в следующем поиске она не участвовала. Поскольку эта операция трансформирует аргумент A (невосстанавливаемое изменение), в начале алгоритма формируется рабочая копия A в T и именно T впоследствии используется для анализа.

Замечания.

1. При исключении *j*-й буквы из T отсутствует проверка, не является ли она крайней (при $j=1$ отсутствует T[1:$j-1$], а при j=длин(T) нет T[$j+1$:длин(T)], что может привести к ошибке при выполнении указанного оператора. То же относится к случаю 'длин(T)=1'.
2. Конструкция пока с двумя условиями излишне громоздка. Лучше использовать команду повторения с параметром.
3. Наращивание индексов *j* и *i* лучше выполнять при входе в цикл, а не при выходе из него, чтобы не создавать «лишних» операций (например, уже сформирован признак «есть», а *j* все равно увеличится на 1) и, с другой стороны, чтобы значения *j* и *i* соответствовали реально проанализированным буквам обрабатываемых слов.
4. В ситуациях, когда слово A короче B, поиск общих букв вовсе не нужен.

Рекомендация 1 (учтены все замечания).

```
алг  Упр_31-1(лит A,B,ответ)
арг  A,B
рез  ответ
нач цел i,j,лит T
  ответ:='нельзя'  'заготовка ответа для негативных исходов'
  если длин(B)<=длин(A)  'попытка составить B из букв A
                          не бессмысленна'
      то T:=A  'создание рабочей копии A'
```

```
        для i от 1 до длин(B)
        нц  'анализ i-ой буквы B-слова:есть ли она в T?'
          для j от 1 до длин(T)
          нц  'поиск нужной буквы в T'
            если T[j:j]=B[i:i]
                 то  'нужная для B буква найдена'
                   если длин(T)>1  'исключение буквы из
                                    дальнейшего анализа'
                     то выбор
                              при j=1 :T:=T[2:длин(T)]
                              при j=длин(T) : T:=T[1:длин(T)-1]
                              иначе T:=T[1:j-1]+T[j+1:длин(T)]
                        все
                   все
                     j:=длин(T)+1  'искусственное прерывание
                                    просмотра T для B[i:i]'
            все
          кц  'просмотр T завершен с j=длин(T)+1,если нужная буква
              не найдена,и с j=длин(T)+2,если таковая есть'
          если j=длин(T)+1
               то  'B[i:i] в T не была найдена'
                 i:=длин(B)+1  'дальнейший анализ B не нужен'
          все
        кц   'просмотр B завершен успешно с i=длин(B)+1
              и прерван,если i=длин(B)+2'
        если i=длин(B)+1
            то  'все буквы B были найдены в T'
              ответ:='можно'
        все
  все   'завершен анализ,если B не длиннее A'
кон
```

78

Рекомендация 2. Вместо использования рабочей копии T=A и исключения из нее использованных букв можно ввести маркировку этих букв. При этом возможность использования букв из A можно маркировать числовым (0 — да, 1 — нет) или литерным значением.

```
алг  Упр_31-2(лит A,B,ответ)
арг  A,B
рез  ответ
нач цел i,j,лит T
  ответ:='нельзя'  'заготовка ответа для негативных исходов'
  если длин(B)<=длин(A)  'весь анализ имеет смысл лишь при
                          условии что слово A не короче B'
       то
         для i от 1 до длин(A)
         нц  'заготовка маркировочной шкалы,в которой всем буквам
             из A соответствует код Y,т.е.буква свободна'
           T[i:i]:='Y'
         кц
         для i от 1 до длин(B)
         нц  'анализ очередной буквы B[i:i]:есть ли она в A?'
           для j от 1 до длин(A)
           нц  'просмотр букв в A,еще не использованных'
             если T[j:j]='Y' и A[j:j]=B[i:i]
                  то  'нужная буква в A найдена'
                    T[j:j]:='N'  'маркировка найденной буквы'
```

```
                    j:=длин(A)+1   'искусственное насыщение j
                                   для прерывания цикла'
              все
            кц  'просмотр A завершен успешно при j=длин(A)+2'
            если j=длин(A)+1
                 то  'просмотр A был завершен безрезультатно'
                   i:=длин(B)+1  'искусственное завершение
                                 цикла для просмотра B'
              все
          кц  'просмотр B завершен успешно при i=длин(B)+1 и прер-
               ван из-за отсутствия нужных букв при i=длин(B)+2'
          если i=длин(B)+1
             то  'все буквы в B можно получить из A'
               ответ:='можно'
          все
  все     'при A не короче B анализ закончен;иначе он не имеет смысла'
кон
```

Упражнение 33. Перестановкой К чисел называется последовательность A[1], A[2]..., A[K], в которой встречаются все числа от 1 до К. Постройте алгоритм, печатающий все перестановки 100 чисел. Используйте в качестве вспомогательного алгоритм 'Печать (цел таб A[1:100])', печатающий элементы таблицы А.

```
алг  метпос_33
нач цел i,цел таб A[1:100],C[1:100],лит признак
  для i от 1 до 100     'Чистка (нулем)управляющей таблицы C:
                         если C[i]=0,число i в перестановку не
                         вошло,если C[i]=1-уже включено'

  нц
    C[i]:=0
  кц
    A[1]:=0;i:=1  'подготовка параметров цикла'
    признак:='продолжить'
  пока признак='продолжить'
  нц  'i-номер первого изменяемого элемента перестановки'
    A[i]:=A[i]+1  'первый изменяемый элемент в перестановке'
    если A[i]<=100
         то если C[A[i]]=0   'i-ый элемент еще не превысил
                              100 и свободен'
               то   'число A[i] включить в перестановку
                     i-тым элементом'
                 C[A[i]]:=1   'признак включения A[i]'
                 если i<100
                      то   'перейти к формированию i-го элемента'
                        A[i+1]:=0   'подготовка параметров цикла'
                        i:=i+1
                      иначе   'i=100,т.е. перестановка сформирована'
                        печать(A)
                        C[A[i]]:=0   'освободить A[100] как
                                      подлежащее замене'
               все
      все
    иначе   'число A[i]>100 и,следовательно,
               не может быть использовано'
            если i>1
                 то   'вернуться к предыдущему элементу и
                       вывести его из перестановки'
                   i:=i-1 ; C[A[i]]:=0
```

```
                иначе    'первый элемент достиг максимума
                          т.е.все перестановки испытаны'
                  признак:='закончить'
              все
        все
     кц
кон
```

Комментарий. В данном алгоритме в качестве первой перестановки формируется натуральная последовательность (1, 2,..., 100). Далее состав чисел меняется, начиная с последних составляющих, т. е. с $i=100,99$ и т. д. После того как A[i] достигает 100 (т. е. наращивать дальше его уже нельзя), алгоритм возвращается к A[$i-1$], наращивает его на 1 и затем корректирует все последующие компоненты перестановки A[i], A[$i+1$],..., A[100].

Основным средством управления составом перестановки является справочная (маркировочная) таблица C[1:100], в которой единицами маркируются числа, уже включенные в формируемую перестановку, нулями — еще не использованные в ней.

В частности, очередное значение A[i] включается в перестановку при условии, что ему соответствует нулевой маркер (C[A[i]]=0).

Каждый из последующих компонентов (A[$i+1$],..., A[100]) формируется аналогично, начиная с нулевого значения: наращивается на 1 (т. е. 1, 2,...,) до тех пор, пока впервые не осуществится нулевой маркер (т. е. C(j)=0, тогда как для всех K<j, C(K)=1)

Замечания.

1. Алгоритм недостаточно нагляден и трудно интерпретируем.
2. В алгоритме очень много прогонов цикла, не завершающихся формированием перестановки. Это и не эффективно, и мешает «прочтению» алгоритма. В первую очередь это относится к многошаговому наращиванию каждого отдельного члена перестановки, не говоря о возвратах к предыдущему члену, и т. д.
3. Серьезное нарекание вызывает само условие задачи. Во-первых, допущена неточность в определении перестановок: почему указаны конкретно числа от 1 до K, когда речь идет просто о «перестановке K чисел», т. е. необязательно K первых натуральных чисел? Во-вторых, культура программирования требует от программиста хотя бы ориентировочного оценивания требований к компьютеру и к техническим средствам для выполнения разработанного алгоритма. По оценке Д. Кнута («Искусство программирования», ч. 1, с. 78), расчет всех перестановок только из 10 чисел должен занять около 1 ч компьютерного времени. Что же говорить о перестановках из 100 (???) чисел, если к тому же их нужно не только сформировать, но и напечатать?!!

Рекомендация. Учтены все замечания. В условии задачи 100 чисел заменены на N чисел и оператор печати перестановок предложено заменить оператором: буфер (A), т. е. пересылкой в буфер, или в двумерную таблицу P[1:M, 1:N], где M — число всех перестановок.

Во избежание в алгоритме «пустых» прогонов и для придания ему большей направленности примем в качестве исходной перестановки возрастающую последовательность (1, 2,..., N), а в качестве конечной — обратную перестановку (N, N—1,..., 1), в которой все числа следуют в порядке убывания. Нарушение этого порядка, т. е. случай A[i]<A[$i+1$], будем для удобства называть «инверсией», хотя обычно, наоборот, под инверсией понимают A[i]>A[$i+1$], имея в виду перестановку (1, 2,..., N).

В таком случае алгоритм получения новой (очередной) перестановки состоит в том, чтобы отследить новую (справа) инверсию и начиная с нее заменить весь «хвост» перестановки, пользуясь только свободными (еще не задействованными, судя по маркировочной таблице) числами.

```
алг  Упр.33(цел N, цел таб P[1:M,1:N])
     арг\ N
     рез \P
нач цел i,j,K, цел таб A[1:N],C[1:N]
  K:=0      'Счетчик готовых перестановок в P'
  для i от 1 до N          'Заготовка начальной перестановки
                            A=(1,2,...,N) и маркировочной таблицы
```

```
                                C=(1,1,...,1)'
  нц
    A[i]:=i;C[i]:=1
  кц
    i:=-1  'признак продолжения основного цикла'
  пока i<0
  нц  'основной цикл:один такт - перестановка'
    k:=k+1
      для j от 1 до N  'засылка готовой перестановки в результ.
                        таблицу P;мог бы быть оператор: буфер (A)'
      нц
        P[k,j]:=A[j]
      кц
    C[A[N]]:=0    'младший член сразу исключается из A'
    i:=N-1    'просмотр начнется с A[N-1]'
    пока i>0
    нц    'цикл для поиска первой инверсии'
      C[A[i]]:=0   'независимо от результата A[i] исключить из A'
      если A[i]<A[i+1]
          то  'инверсия обнаружена;замене подлежат
               все члены,начиная с A[i]'
            j:=A[i]+1    'ищется минимальное j,большее A[i],
                          с нулевым маркером,т.е. с C[j]=0'
            пока C[j]=1
            нц
              j:=j+1
            кц
             A[i]:=j;C[j]:=1  'засылка j;изменение маркера'
            для j от 1 до N  'ищется очередное (по возрастанию)
                              число с нулевым маркером взамен
                              каждого из последующих A[i+1],...,A[N]'
            нц
              если C[j]=0
                  то i:=i+1  'текущий номер A'
                     A[i]:=j;C[j]:=1   'корректировка A,C'
              все
            кц
              'признак для возвращения в основной цикл'
              i:=-1  'перестановка сформирована'
          иначе   'продолжить поиск инверсии'
            i:=i-1
      все
    кц  'конец поиска инверсии с i=-1,если поиск был успешен и
         сформирована новая перестановка,и с i=0,если инверсий нет,
         т.е. предыдущая перестановка последняя:(N,N-1,...,1)'
  кц  'конец основного цикла при i=0'
кон
```


Литература

1. Основы информатики и вычислительной техники: Ч. 1 / *А. П. Ершов, В. М. Монахов.* М.: Просвещение, 1986.

2. Основы информатики и вычислительной техники: Ч. 2 / *А. П. Ершов, В. М. Монахов.* М.: Просвещение, 1986.

3. Изучение основ информатики и вычислительной техники: Метод. пособие для учителей и преподавателей сред. учеб. заведений: В 2 ч. Ч. 1 / *А. П. Ершов, В. М. Монахов, М. В. Витиньш* и др.; Под ред. А. П. Ершова, В. М. Монахова. М.: Просвещение, 1986.

4. Изучение основ информатики и вычислительной техники: Метод. пособие для учителей и преподавателей сред. учеб. заведений: В 2 ч. Ч. 2 / *А. П. Ершов, В. М. Монахов, М. В. Витиньш* и др.; Под ред. А. П. Ершова, В. М. Монахова. М.: Просвещение, 1986.

5. *Кнут Д.* Искусство программирования. М.: Мир, 1977.

От редакции. Несмотря на довольно обширную литературу по программируемым микрокалькуляторам, выпущенную к настоящему времени, редакционная почта свидетельствует о значительном интересе к этому виду ВТ в учительской среде. Он объясняется не только недостаточными тиражами изданий по этой тематике, которые, как утверждают авторы писем, практически не доходят до «глубинки», но и тем, что подобная литература не носит методического характера. Начиная публикацию статей, отражающих один из подходов к построению уроков с ПМК, мы надеемся, что они будут полезны учителям школ, преподавателям техникумов и профтехучилищ.

Л. ШТЕРНБЕРГ,
канд. физ.-мат. наук

Уроки с программируемыми микрокалькуляторами

Наиболее распространенным средством вычислительной техники, которым оснащены многие школы, является программируемый микрокалькулятор. Его вычислительные возможности невелики, но вполне достаточны для решения задач, возникающих в школьном курсе.

Рассмотрим методику проведения уроков с программируемыми микрокалькуляторами. Для школ лучше всего подходят модели БЗ-34, МК-54, МК-56, имеющие одинаковую систему команд, которая более проста и логична, чем у моделей БЗ-21, МК-46. Впрочем, методика проведения урока сравнительно мало зависит от типа калькулятора. Мы будем использовать названия клавиш модели МК-56 и МК-54 (у БЗ-34 несколько клавиш отличаются по надписям на них).

Для проведения урока необходимо иметь не менее одного калькулятора на парту (т. е. на 2 человека) и один — для учителя. Для надежного проведения урока желательно работать с машинками, включенными в сеть.

Возможно два варианта проведения уроков. Первый — по мере изучения алгоритмического языка: изучили линейные программы — изучаем их реализацию на калькуляторе, изучили ветвления — рассматриваем, как это сделать на калькуляторе, и т. д. Но можно принять и другой вариант: сначала изучаем алгоритмический язык, затем проводим уроки по программированию рассмотренных алгоритмов на калькуляторах.

Тема «Структура ЭВМ на примере программируемого микрокалькулятора»

Эта тема занимает два урока. Желательно иметь плакаты «Лицевая панель калькулятора» и «Внутреннее устройство калькулятора» (с минимальным количеством деталей: операционные и адресуемые регистры, программная память, счетчик адреса, клавиатура, индикатор, процессор в виде блока без детализации).

Первый урок. В течение 10 мин, используя плакаты, надо описать основные блоки микрокалькулятора: процессор (мы его не видим — он внутри), память для программы и данных, устройство ввода (клавиатура), устройство вывода (индикатор). Надо отметить, что исполнитель «Калькулятор» понимает «язык», состоящий из нажатий клавиш.

Далее целесообразно сразу же переходить к работе на калькуляторах. Поскольку в школе нет возможности делить класс для этих занятий на подгруппы, то учитель вынужден проводить уроки сразу с 25—30 учениками. Если сначала объяснить им, как работать на калькуляторах, а затем предложить работать самостоятельно, то учитель не будет успевать отвечать на возникающие вопросы. Одним из возможных вариантов является проведение первых уроков с калькуляторами в «командном режиме», когда весь класс синхронно по команде учителя выполняет одинаковые действия. Учитель параллельно объясняет, что при этом происходит. Если у кого-то из-за ошибочного действия получается не то, что

надо, учитель не отвлекается на частные ошибки (не всегда можно быстро понять, что именно неправильно сделал ученик, тем более что не всякую ошибку можно быстро нейтрализовать), а предлагает перейти к работе на одном калькуляторе с соседом (при работе парами на одном калькуляторе пара распределяется по одному человеку к двум другим парам). Опыт показывает, что за урок происходит порядка 5 таких случаев.

Итак, учитель объясняет, что клавиши надо нажимать только по его команде. На вопрос: «У всех на индикаторе то-то?» — должен следовать *молчаливый* кивок ученика, если все в порядке, или же ученик должен поднять руку (и то, и другое легко фиксируется учителем). На первом уроке достаточно объяснить работу в режиме вычислений с использованием только регистров X и Y (без работы со стеком) и адресуемых регистров. При этом учитель может произнести монолог, примерно такого содержания:

— В верхнем левом углу панели вы видите рычажок включения. Включить калькулятор! На индикаторе загорается ноль. У всех есть эта индикация? (Взгляд на класс.) В микрокалькулятор числа вводятся последовательным нажатием клавиш: введем число 25 — нажмем «2», на индикаторе «2», нажимаем «5» — на индикаторе «25». (Далее следует вопрос: «У всех получилось?» — в тексте опускается, но звучать он должен постоянно.) Число 25 записано в регистре X. Клавишей «B↑» число 25 засылается в регистр Y и остается в регистре X — содержимое Y мы не видим. Нажмем «3», на индикаторе — «3», теперь в регистре Y «25», а в регистре X — «3». Можем в этом убедиться: клавиша «↔» (целесообразно параллельно показывать эти клавиши на плакатах) обменивает содержимое регистров X и Y. Нажимаем ее и на индикаторе — «25».

Теперь выполним, например, сложение: нажмите «+» — на индикаторе «28». Это в X оказалась сумма X и Y. Прочие двуместные операции выполняются аналогично.

Далее можно предложить тем, кто перешел к калькуляторам соседа, вернуться к своим машинкам — это удобный момент для возврата. Учитель объясняет назначение переключателя «радианы — градусы», одноместные операции, ввод дробных и отрицательных чисел, чисел с порядком (эти вопросы трудностей не вызывают: с ними ученики знакомы по инженерным калькуляторам), а также исправление ошибочно набранного числа.

Затем переходим к работе с адресуемыми регистрами.

— Наберем, например, число 3 — нажмем «3» — оно загорелось на индикаторе. Теперь нажмем клавишу «x→П» — число на индикаторе мигнуло — и клавишу с номером регистра, например «0»,— число на индикаторе еще раз мигнуло — теперь оно записано в регистр 0 и осталось в регистре X...

Аналогично заносим число 4 в P1, 5 — в P2, затем вызываем их опять в регистр X, иллюстрируя работу команд «П→x» и «x→П».

— Теперь проведем небольшой расчет. Например, зная длины сторон треугольника вычислим косинус угла, лежащего против стороны, указанной первой. (Нужно провести расчет по одной из линейных программ из предыдущих уроков.) Допустим, длины сторон a, b, c равны соответственно 3, 4, 5. При этом число 3 находится в P0, 4 в P1, 5 в P2.

Расчет проводим по формуле:

$$\cos A = (b^2 + c^2 - a^2) / (2 \cdot b \cdot c)$$

— Нажмем «П→x» «1» — на индикаторе «4» — это на регистр X вызвано значение b. Возводим его в квадрат: «F» «x^2» — на индикаторе «16». Вызываем в X значение c «П→x» «2» — на индикаторе «5»; калькулятор устроен так, что при вызове нового числа или его наборе результат предыдущей операции автоматически уходит в регистр Y. Сейчас у нас в X значение c, а в Y — значение b^2. Возводим c в квадрат...

В таком стиле объясняем дальнейшие вычисления. Поскольку пока пользуемся регистрами X и Y, то значение числителя придется записать в адресуемый регистр, а затем вызвать из него. Для объяснения происходящего удобно иметь таблицу следующего вида:

Клавиши	Индикация	Комментарий
П→x 1	4	$b \to X$
F x^2	16	$X^2 \to X$; теперь $X = b^2$
П→x 2	5	$X \to Y$; $c \to X$: $X = c$; $Y = b^2$
. . . . .		

Желательно иметь такую таблицу в виде плаката (наилучший вариант — в виде раздаточного материала), но можно рисовать и на доске.

В качестве домашнего задания можно предложить сделать аналогичную таблицу для одной из задач (например, решение квадратного уравнения при заведомо неотрицательном дискриминанте). Таблица должна отражать последовательность нажатий клавиш, ожидаемые на индикаторе результаты для каких-то простых данных, комментарий происходящих действий.

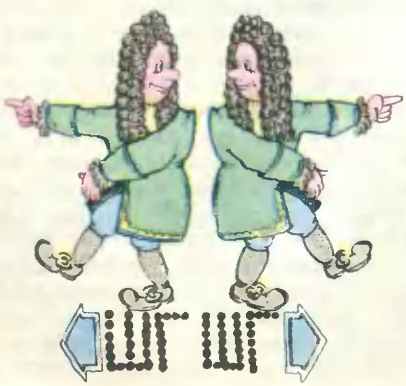
Второй урок. Тема этого урока — запись программы в память калькулятора и ее

выполнение. На уроке необходимо иметь плакаты: лицевая панель калькулятора, таблица кодов команд калькулятора (ее хорошо иметь и в виде раздаточного материала), таблица «клавиши — индикация — комментарий» из предыдущего урока, дополненная слева столбцами «адреса» и «коды» (на первом уроке эти столбцы можно закрыть или просто не рассматривать).

Этот урок целесообразно провести в «командном» режиме с контролем «у всех получилось?» на каждом шаге и описанием расположения новых клавиш. Для этого удобно использовать ту же задачу, которая была просчитана в режиме вычислений на прошлом уроке. Для напоминания техники работы в режиме вычислений занесем значения 3, 4, 5 в P0, P1, P2 соответственно — они нам пригодятся. Дальнейший рассказ учителя может выглядеть, например, так:

— Переводим калькулятор в режим программирования: нажмем «F» и «ПРГ» — на индикаторе видим два нуля на месте порядка. Теперь калькулятор будет **не исполнять** подаваемые нажатием клавиш команды, а **запоминать их** в программной памяти в закодированном виде. Коды команд вы видите в таблице (указание на плакат). На индикаторе справа вы видите с ч е т ч и к а д р е с а: он показывает, что очередной код будет запомнен в ячейке с адресом ноль. Теперь будем нажимать те же клавиши в том же порядке (этот момент надо подчеркнуть). Нажимаем «П→x» «1» — на индикаторе появился код «61» — по таблице видим, что это код команды «П→x» «1», он запомнен в ячейке 00, а в счетчике адреса горит 01: следующая команда будет запомнена в ячейке 01...

Таким образом вводим всю программу, добавив в конце «С/П», назначение которой объясняется позже. В процессе ввода необходимо сделать 4 ошибки, чтобы показать, как их исправлять. Первая исправляется сразу:



— Теперь давайте ошибемся: допустим при вводе команды «П→x» «0» мы плохо нажали клавишу «П→x» — вы ее не нажимайте вообще — нажимаем «0» — вместо кода 60 появился код 00...

Исправляем эту ошибку с помощью клавиши «ШГ». Следующую ошибку «замечаем» не сразу, а после набора еще двух команд и показываем ее исправление с помощью клавиши «ШГ» с пропуском уже набранных после нее команд клавишей «ШГ». Тем самым показывается, как клавишами «ШГ» и «ШГ» двигается «окошко», сквозь которое видны три ячейки памяти. Еще две ошибки «оставляем незамеченными».

Далее выполняется проверка программы. Нажимаем клавиши «F» «АВТ», «В/О», «F» «ПРГ» и читаем программу с помощью клавиши «ШГ». На этом этапе надо заметить и исправить одну из сделанных ошибок (вторую пока «не замечаем»).

Теперь выполняем программу по шагам: «F» «АВТ», «В/О» и многократно нажимаем клавишу «ПП». На этом этапе следует обратить внимание на то, что на каждом шаге выполняется одна очередная команда из памяти. Называем эту команду и показываем, что на индикаторе получается то же самое, что было при работе в режиме вычислений. На этом этапе у нас проявится последняя ошибка: для ее обнаружения надо после появления на индикаторе ошибочных показаний нажать «F» «ПРГ», и в левом углу индикатора мы увидим код ошибочной команды, т. е. команды, вызвавшей неправильный результат. Исправление программы выполняется обычным способом, после чего программа проверяется сначала.

После проверки программы можно ввести понятие **автоматического** режима работы калькулятора, когда он выполняет команды одну за другой, не дожидаясь нашей команды (нажатия клавиши). Здесь объясняется назначение команды «С/П» в программе: выход из автоматического режима и клавиши «С/П» — запуск работы в автоматическом режиме. Подготовив программу к работе (занеся ее начальный адрес в счетчик клавишей «В/О»), выполним программу, нажав «С/П». Здесь следует отметить, что подготовка программы, как мы видели, достаточно трудоемка, но теперь когда мы имеем готовую программу в калькуляторе, можно легко проводить вычисления для разных исходных данных. Здесь следует ввести в P0, P1, P2 другие значения длин сторон треугольника и получить результат для них. Таким образом следует просчитать несколько вариантов при

различных исходных данных.

Домашнее задание: при возможности организовать работу на калькуляторах во внеурочное время можно предложить провести всю работу по вводу и отладке программы из домашнего задания прошлого урока; если такой возможности нет, то выписать на листе бумаги всю последовательность нажатий клавиш при вводе программы (без умышленного введения ошибок) и показания индикатора.

Основные методические моменты. Наиболее целесообразно провести эти уроки сразу после введения понятия величины и рассмотрения первых алгоритмов работы с величинами, т. е. в начале 2-й четверти.

В заключение этих уроков полезно сделать обзор выполненной работы и отметить следующие моменты.

Калькулятор является **формальным исполнителем:** он «не понимает» смысла того, что считает,— здесь следует напомнить, как калькулятор «не замечал» ошибок, которые мы вносили в программу.

Система команд исполнителя «Калькулятор» представлена в рассмотренной на уроке таблице кодов; «каждая его команда указывает... одно конкретное законченное действие, и исполнитель должен выполнить его целиком» (дискретность алгоритма), «выполнение всех команд гарантирует правильное решение задачи» (конечность алгоритма) [1].

Калькулятор с введенной программой позволяет решать множество однотипных задач (массовость алгоритма).

Таким образом, работа на калькуляторе позволяет хорошо проиллюстрировать и закрепить основные положения пройденной части курса.

Связь программ для ПМК с алгоритмами на алгоритмическом языке будет рассмотрена в следующих статьях.

Литература

1. Основы информатики и вычислительной техники. Ч. 1. М.: Просвещение, 1985.

Продолжение следует.

Л. ЗЕМЦОВА, А. ЛУКАНКИН
НИИ школ МНО РСФСР

Форма или содержание?

Лабораторией обучения программированию и ЭВМ НИИ школ МНО РСФСР начиная с 1984/85 учебного года ведется систематическая работа по отработке методики преподавания машинного курса ОИВТ. В 1985—1986 гг. совместно с Московским институтом электронной техники эксперимент проводился в школах Зеленоградского района Москвы. Экспериментальная программа курса с самого начала предусматривала максимальное использование имеющейся техники (ДВК-1 и ДВК-2), так как возможность реализации алгоритма на ЭВМ и получение результата работы машины повышает у учащихся мотивацию изучения информатики.

Для оценки правильности того пути, по которому шло экспериментальное обучение, и для своевременной коррекции содержания и методики преподавания курса в ходе эксперимента проводилось психолого-педагогическое исследование особенностей мотивации мышления и некоторых личностных характеристик учащихся. Нами использовались методики ранжирования, анкетирования, а также дидактические методики: «составление задач», «выбор», которые позволяют осуществлять диагностику психологических особенностей учащихся не в результате лабораторного эксперимента, а в процессе выполнения учебных заданий (составление задач, выбор и решение выбранной задачи и т. д.).

При рассмотрении мотивации изучения информатики нами были выделены два основных типа мотивов — «операциональные» и «содержательные». Для того чтобы учение способствовало развитию мышления учащихся, необходимо, чтобы они стремились к овладению «схемами вещей», т. е. общими способами действий по решению классов задач. Этому соответствуют выделяемые нами среди широких познавательных мотивов учения «содержательные мотивы», показывающие степень развития мотивации «приобретения обобщенных способов».

«Содержательные» мотивы проявляются в интересе учащихся к решению задач и характеризуют ориентацию учащихся на овладение способами их решения. Однако в практике обучения нередко ЭВМ рассматривается как «игрушка», учащимся нравится «нажимать клавиши». Выделенные нами «операциональные» мотивы характеризуют интерес учащегося к технической стороне работы на ЭВМ.

Результаты первого опроса, проведенного в октябре 1985 г., показали, что у 10 % уча-

щихся отмечалось негативное отношение как к предмету «информатика» в целом, так и к ЭВМ в частности. Для 48 % характерно преобладание «операциональных» мотивов в общей структуре мотивации, у 42 % — «содержательных». Таким образом, для 90 % учащихся (из 178 обследованных) возможность работы на ЭВМ выполняет стимулирующую роль в изучении курса ОИВТ, однако лишь в 42 % этот интерес определяется и содержательными мотивами.

Недоработка в этом плане связана с некоторым смещением на начальном этапе эксперимента акцента на операторскую деятельность учащихся: составленные общими усилиями наиболее сильных учащихся и учителя на теоретическом занятии программы затем проверялись на ЭВМ. Таким образом, на практических занятиях учащиеся выполняли чисто операторскую работу — набор на клавиатуре готовых программ, исправление допущенных при наборе ошибок.

Полученные в психолого-педагогическом исследовании результаты дали основание для корректировки содержания курса и перестройки методики проведения практических занятий. Результатом первого года эксперимента явилось создание учебно-методического комплекта, в который входят экспериментальная программа, методические рекомендации учителю, по существу являющиеся развернутыми конспектами занятий, а также практикум для учащихся.

Ведущей идеей курса является последовательное изучение алгоритмических структур с параллельным рассмотрением их реализации в изучаемой версии языка программирования. Особое внимание уделяется анализу алгоритмов. Необходимо добиваться четкого понимания учащимися назначения и взаимосвязи всех элементов рассматриваемого алгоритма. На первый план выдвигается задача овладения основными операциями (присваивание, цикл, ввод, вывод, вызов подпрограммы, проверка условия и т. д.). Для учащегося, уяснившего сущность каждой операции на примере конкретной версии языка программирования, реализованной на конкретной ЭВМ, уже не должно вызывать значительных затруднений изучение другого языка и освоение другой машины. При рассмотрении структуры алгоритма удобно записывать его блок-схему и программу на языке программирования параллельно, проводя сравнительный анализ их элементов.

Практикум для учащихся представляет собой подобранную для каждого практического занятия систему заданий и предусматривает самостоятельную работу учащихся.

Основное содержание практикума составляют задания предполагающие вариативную работу учащихся с одной или двумя условно крупными задачами, в которых выделяются подзадачи. Например, на теоретическом занятии какая-либо задача подготавливается для решения на ЭВМ. Определяются исходные данные и искомые величины, строится алгоритм ее решения и записывается в виде блок-схемы. Одним из заданий практического занятия может быть запись алгоритма в виде программы, ее отладка и счет по ней. При этом может требоваться варьирование исходных данных и некоторая модификация программы. Другой тип задания: по приведенной блок-схеме алгоритма (учащийся впервые видит ее на данном занятии) требуется непосредственно за клавиатурой составить программу и произвести по ней счет для заданных исходных данных. Цель первых двух типов заданий — научить учащихся составлять программы по готовым алгоритмам. Третий тип задания: дается текстовая формулировка задачи, требуется непосредственно на занятии построить алгоритм ее решения, записать его в виде программы, ввести программу в машину и получить результат ее работы. Выполнение учащимися такого типа заданий является комплексной проверкой их способности на практике применить полученную сумму знаний. Приведем в качестве примера текст занятия практикума, посвященного вопросам практического использования массивов.

На этом занятии предлагается, используя известные вам алгоритмы обработки массивов, решить некоторые «жилищные вопросы».

Рассмотрим абстрактный типовой микрорайон, в котором построено 20 типовых домов по 100 квартир в каждом. Определите для себя размерность массива, в который можно записать информацию о всех квартирах этого микрорайона.

1. Запишите в двумерный массив A метраж каждой квартиры микрорайона. Размер площади квартир задавайте генератором случайных чисел в интервале от 20 до 60.

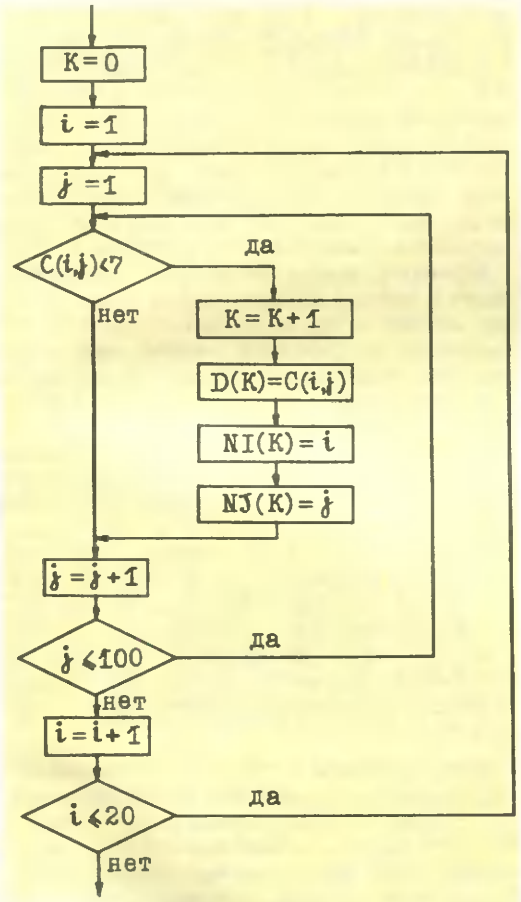
2. В массив B запишите количество проживающих в каждой квартире. Значения элементов также задайте генератором случайных чисел. Пусть в квартирах может проживать от 1 до 10 человек.

3. В качестве значений элементов массива C задайте количество метров, приходящееся на одного жильца в каждой квартире:

$C(i, j) = A(i, j) / B(i, j)$

4. Определите номера дома и квартиры, в которой на одного жильца приходится минимальная площадь.

5. Определите все квартиры, в которых



на одного проживающего приходится менее 7 м^2 жилой площади.

Воспользуйтесь фрагментом блок-схемы: В массив D записываются значения площади, в массив NI — номер дома, в массив NJ — номер квартиры. При последующей одновременной распечатке в одном цикле значений всех трех массивов вы получите список квартир, жильцы которых наименее обеспечены жилой площадью.

Работа с практикумом обеспечивает постоянную занятость всех учащихся на протяжении всего занятия. Значительно облегчается труд педагога, в то же время значительно повышается продуктивность занятия. Подобная организация практических занятий способствует повышению мыслительной активности учащихся, формированию у них направленности на овладение различными способами деятельности.

Изменения, внесенные в содержание и методику преподавания курса, оказали влияние на структуру мотивационной сферы учащихся. Срез показал, что число учащихся с «операциональными» мотивами уменьшилось до 24 %, а группа учащихся с развитыми «содержательными» мотивами выросла до 60 %.

Проведение в ходе экспериментального обучения психолого-педагогического исследования позволило определить некоторые условия наиболее продуктивного подхода к методике преподавания в школе машинного курса информатики:

максимальное использование имеющейся вычислительной техники;

практическая направленность изучаемого материала;

руководство деятельностью учащихся на практических занятиях посредством письменных заданий типа «Практикум».

А. КРИЦКИЙ, В. ЛЕЦКО

Компьютерные сети в обучении

Большая часть современных обучающих систем, основанных на использовании компьютера, так или иначе соотносится с идеологией программированного обучения. В них компьютеру передается ряд функций учителя — предъявление заданий, контроль, оценка и т. д. К достоинствам таких систем относятся индивидуализация обучения, основанная на жесткой обратной связи, адаптивность машины к особенностям деятельности и усвоения учебного материала учащимися. Практика подтверждает правомерность их применения при решении ряда частных задач обучения (прежде всего при формировании практических умений и навыков).

Следует указать и на существенные недостатки подобных систем. Первый — репродуктивно-рецептурная схема процесса передачи знания, встроенная в процесс программированного обучения, серьезно затрудняющая освоение учащимся рефлексивно-теоретического способа мышления, формирующегося при введении их в области научного знания. В теории учебной деятельности показано, что усвоение научных понятий и связанного с ними способа мышления требует воспроизведения в обучении условий происхождения самих понятий, т. е. построения моделей соответствующих ситуаций.

Второй недостаток связан с тем, что уче-

ник в своей работе «замкнут» на компьютер. Взаимодействие «ученик — машина» моделирует взаимодействие «ученик — учитель», а при всяком моделировании неизбежны потери, и важно выяснить, насколько они существенны в той или иной ситуации. Общение и взаимодействие в моделирующей системе обедняются, теряются важные компоненты, прежде всего мотивационные и эмоциональные. Все это сказывается на развивающих и воспитывающих возможностях обучения. Технология, основанная на идеях программированного обучения, дублирует некомпьютерные методы, совершенствуя при этом одни компоненты и разрушая или обедняя другие.

Новые направления в компьютеризации образования связаны с проектированием предметно-ориентированных компьютерных учебных сред, дающих ребенку возможность экспериментировать в конкретных областях, входящих в состав учебных предметов. Наиболее известна среда ЛОГО, изобретенная С. Пейпертом. Он выдвигает новый принцип построения обучающих технологий — конструктивизм. В основу этих технологий положена конструктивная активность ребенка, исследующего предметную среду, что позволяет преодолеть шаблонность компьютерных методик, ориентированных на натаскивание. Однако и здесь ученик остается «замкнутым на компьютер». Общение учащихся предполагается, но не обусловливается структурой обучающей среды. Серьезные трудности порождает также спонтанность диалога ребенка с компьютером.

Мы полагаем, что эти трудности можно преодолеть, если организовать предметно-ориентированную обучающую среду следующим образом.

1. Новое знание должно рождаться не только и не столько во взаимодействии ребенка с компьютером, сколько в активном сотрудничестве детей друг с другом. Компьютер же призван обеспечить естественное разделение ролей в этой совместной деятельности.

2. В функции учителя в такой среде входит контролирование процесса решения учебных задач и внесение изменений в условия совместной деятельности. Цель этих изменений — столкнуть учеников с противоречиями, преодоление которых обеспечит прорыв к новому знанию. Эти же противоречия должны стать отправными точками в дискуссиях (как организуемых учителем, так и возникающих самопроизвольно).

С учетом изложенного нами разработан пакет программ для КУВТ «Ямаха», моделирующий предметную среду для обучения основным понятиям кинематики. Учебные задачи возникают и решаются учениками вместе с учителем в ходе игры. Играющие разбиты на пары; один из участников управляет вертолетом, а другой работает за экраном локатора. Игровая цель пары — с помощью вертолета доставить груз на корабль, терпящий бедствие. При этом «летчик» не видит корабль, и ему должен помочь сидящий у экрана локатора. Для обмена сообщениями между партнерами, работающими за разными дисплеями, используется локальная сеть.

Основная задача учителя — «вставлять палки в колеса» играющим. Для этого он может незаметно для играющих смещать начала координат в системах отсчета партнеров, изменять относительный масштаб, заставить двигаться корабль или одну из систем относительно другой.

Пакет прошел апробацию с испытуемыми различных возрастных групп — от пятиклассников до студентов. Было проведено пять серий экспериментов по 4—6 заданий в каждой. Варьировались как «внешние» условия (члены пары менялись или не менялись ролями, могли общаться непосредственно или только с помощью «компьютерной почты» и т. д.), так и «внутренние» (наличие или отсутствие систем координат на экранах ПЭВМ, число комплектов сбрасываемого груза и т. д.).

Предположение о том, что выбранная при проектировании эксперимента форма организации совместной деятельности будет способствовать формированию учебной направленности этой деятельности, подтвердилось. Полученные данные позволяют по-новому взглянуть на роль компьютера в процессах усвоения научных понятий и соответствующих видах деятельности: компьютерная сеть может выступить мощным средством в системе организации учебного взаимодействия. В этой ситуации существенно изменяются значение и структура компьютерной игры: она выступает как источник возникновения учебно-исследовательских задач и «очеловечивается»: система «ученик — компьютер» замещается системой «ученик — компьютер — ученик». В этой новой системе компьютер выполняет ряд новых функций, среди которых наиболее важная — обеспечивать для учителя возможность активного влияния на развитие учебной кооперации школьников.

Таким образом, использование компьютерных сетей открывает пути к разработкам нетрадиционных форм организации совместной учебной деятельности. Выявление и исследование этих новых возможностей — очень перспективное направление. Но вряд ли можно ожидать здесь серьезных успехов, если не возникнет реальной кооперации педагогов и психологов с разработчиками и производителями учебных ПЭВМ.

Школа в Красноярске

При кафедре вычислительной математики Красноярского государственного университета три года работает школа юных программистов (ШЮП). Ее работу возглавляет проректор по научной работе В. Сапожников. В школе два отделения: очное и заочное. Программа обучения рассчитана на два года.

Преимущества при поступлении имеют старшеклассники, у которых устойчивый интерес к технике и хорошие знания по естественным наукам.

Школьники, зачисленные на очное отделение ШЮП, имеют возможность два часа в неделю заниматься программированием под руководством преподавателей кафедры. Расписание составлено таким образом, что лекции чередуются с практическими занятиями на ЭВМ, по желанию ребята могут заниматься отладкой своих программ еще 2—4 ч в неделю. Проводятся тематические вечера, конкурсы программистов. Ребята, прошедшие двухгодичный курс обучения, получают дипломы об окончании школы юных программистов.

На заочное отделение ШЮП принимаются в основном иногородние учащиеся. Мы высылаем им методические разработки, задачи для самостоятельной работы с подробными решениями и контрольные задания. Программы, составленные школьниками при выполнении контрольной работы, просчитываются на ЭВМ и листинги высылаются учащимся. Старшеклассники, успешно выполнившие несколько контрольных работ, получают приглашения в Красноярск для участия в краевой летней школе юных программистов. Живут они в Доме молодежи, где созданы прекрасные условия не только для работы, но и для отдыха.

Во второй половине июня десять дней машинный парк университета предоставлен школьникам. Дисплейный класс ЭВМ ЕС-1061, классы персональных компьютеров «Искра-1256» и «Корвет» — в общей сложности 40 рабочих мест — вот та реальная база, благодаря которой школьники могут существенно повысить уровень своей компьютерной грамотности. Кроме ежедневных четырехчасовых занятий на ЭВМ в программу школы включены лекции преподавателей университета, конкурс программистов, экскурсии, различные спортивные и культурные мероприятия.

Активную помощь в организации и проведении краевых школ юных программистов оказывают Краевой отдел народного образования, краевой комитет ВЛКСМ и БММТ «Спутник».

В 1987/88 учебном году на заочном отделении обучалось 476 человек, в десяти группах очного отделения — 160 человек.

С 17 по 27 июня проходила краевая летняя ШЮП-88. В ней приняли участие 132 человека из 28 городов и районов края. Для проведения занятий было сформировано девять групп. Шесть состояли из учащихся, успешно закончивших первый год заочного обучения программированию. Теперь они получили возможность закрепить полученные знания непосредственно за дисплеем ЭВМ. Кроме основных занятий по программированию с использованием одного вида вычислительной техники учебная программа предусматривала ознакомление школьников с различными типами ЭВМ, имеющимися в университете и школе № 41, являющейся базовой школой Красноярского филиала ВЦ СО АН СССР.

Для участия в летней школе были при-

глашены не только заочники, но и другие старшеклассники края, интересующиеся информатикой. Для них была составлена специальная программа по основам программирования с использованием персональных компьютеров «Корвет». Эта программа позволила сформировать у старшеклассников представление о возможностях ПЭВМ. Теоретический материал был дополнен разнообразными задачами: вычислительными, учебно-демонстрационными, графическими, игровыми, а также задачами по программированию фрагментов музыкальных произведений.

Не остались без внимания и руководители делегаций школьников (по просьбе организаторов ШЮП-88 учащихся сопровождали учителя информатики). Для них работал методический семинар, на котором обсуждались вопросы преподавания информатики в общеобразовательной школе, проблемы методики обучения и создания кабинетов информатики.

Учителя имели возможность ознакомиться с опытом преподавания курса «Основы информатики и вычислительной техники», а также с разнообразными формами внеклассной работы по информатике с учащимися в школе № 41. Учитель этой школы Е. Кузнецов показал работу нескольких обучающих и учебно-демонстрационных программ, ответил на многочисленные вопросы, высказал свои соображения по усовершенствованию программы курса. В последние дни работы школы состоялся «круглый стол» с дирекцией ШЮП.

У Красноярской краевой летней школы юных программистов уже есть свои традиции. Каждый вечер во время ее работы «Корвет» воспроизводил позывные школы. Есть у нас и свой значок. Дню открытия посвящается большая информационно-музыкальная программа «Добро пожаловать», проводятся тематические вечера.

Н. МОСКВИНА

От игры к серьезному обучению

— Еще одна статья о «вечной» проблеме компьютерных игр? — вздохнет читатель, готовясь перевернуть страницу.

— И да, и нет,— ответим ему с надеждой удержать его внимание.

Давно известно, что, организовав процесс познания в виде игры, можно резко повысить эффективность обучения. Правда, проводя игры-занятия эпизодически, мы уподобляем их встречам с Дедом Морозом: промелькнет захватывающее действие, когда учишься и одновременно отдыхаешь, развлекаешься, и вновь впереди неприступные — или по крайней мере необозримые — высоты полезных знаний.

А если обучение в такой форме продолжать достаточно долго, не сможем ли мы сформировать у детей впечатление, что сам предмет — это в основном интересная игра, и тем самым задать оптимистичный тон дальнейшему образованию?

Информатика более других дисциплин способна преподносить себя в радужном свете и это очень соответствует настрою ребят 8—10 лет. В этом возрасте они еще не утратили любовь к сказочным действам и в большинстве своем не стесняются этого. Такой контингент является особенно благодатным для систематического обучения, выражаясь современным языком, в стиле погружения.

Итак, с чего начать создание «сказочной» атмосферы? Отнюдь не с «фирменных» компьютерных игр. Важно предложить ребенку серьезные по содержанию (с его точки зрения) виды работ на ЭВМ, показав одновременно, что компьютер привносит в них увлекательность.

Загрузите «Реверси» («Отелло») или любую другую позиционную игру, вплоть до «крестиков и ноликов». Не следует пытаться провести большую часть времени за экраном, лучше обсудите занятие с детьми в возможно более романтическом, даже фетишизирующем компьютер духе. На втором занятии перейдите к тренажеру типа «Погрузка корабля»... Не буду перечислять все подходящие программные средства для первых занятий, их подбор — дело вкуса преподавателя. Важно отойти от распространенного стереотипа «познание компьютерных премудростей столь сложно, что в конце занятия обязательно нужно немного поиграть» к более достойному — «овладение информатикой — одна из наиболее увлекательных игр». Однако сохранить такое отношение к предмету удастся, только проявляя об этом постоянную заботу.

Итак, прошла пора первых демонстраций; переходим к систематическому изучению языка программирования Лого, средства, обладающего не только превосходными воспитательными качествами, но и имеющего

значительный производственный потенциал.

Начнем, конечно, с «черепашьей» графики. А как обыграть все базовые конструкции программирования, опираясь на свойственную ребенку тягу к освоению нового? В настоящее время оптимальной представляется работа с версией языка, включающей операторы связи с подвижным объектом. В Ленинградском Дворце пионеров параллельно с началом опытной эксплуатации интерпретатора Лого, созданного в НТО АН СССР (Ленинград), разработана плата сопряжения «Ямахи» с электромеханическими игрушками. Реализуется возможность передачи управляющего сигнала из компьютера и сигналов, формируемых датчиками, установленными на игрушке.

Реальные объекты, действующие по составленной самостоятельно программе,— мощное средство, окрыляющее ребенка и способное растрогать даже взрослого.

Вполне реально создание несложного унифицированного микропроцессорного блока для управления электромеханическими игрушками и выпуск программаторов для обновления программ в его ППЗУ. Подготовка производства конструкторов и готовых игрушек, способных работать под управлением блока, также не является сложной задачей. Если же этот план привлечет внимание достаточно мощного предприятия, оно может рассчитывать на необозримый рынок. Потребуется, конечно, определенная «поддержка», например, в форме журнальных публикаций, но мы помним, что сходный путь был успешно пройден входившими в нашу жизнь в начале нынешнего десятилетия программируемыми калькуляторами.

Путь к «фантастической» картине ползающих на коленях малышей, наблюдающих за поведением и взаимодействием одухотворенных ими игрушек — фактически играющих в «казаков-разбойников» компьютерного века,— не столь длинный, как представляется на первый взгляд.

Т. ВОЛОХОНСКИЙ

ЧТО МОЖЕТ ЭВМ?

Что выбрать

Для любой работы важно правильно выбрать инструмент, чтобы сделать ее качественно и быстро. Именно эти обстоятельства заставляют разработчиков компьютерной техники помимо высокопроизводительных универсальных процессоров создавать специализированные, каждый из которых наиболее оптимально решает одну узкую задачу. Это не означает, впрочем, что специализированные процессоры проще своих универсальных братьев.

Как правило, спецпроцессоры работают не самостоятельно, а под управлением «своего» универсального процессора, с которым они совместимы. Поэтому для каждого универсального процессора разрабатывается свой класс специализированных процессоров, или сопроцессоров.

Наиболее распространенным до последнего времени был сопроцессор, предназначенный для выполнения операций над числами с плавающей запятой. Однако с появлением высококачественных цветных графических дисплеев возникла необходимость и в процессорах, которые могли бы перерабатывать огромные объемы целочисленной информации с высокой скоростью. И такие процессоры появились.

Американская фирма «Интел» предлагает потребителям графический сопроцессор 82786, который позволяет повысить скорость работы графической станции в сто раз. Кроме того, процессор предоставляет пользователям широкий набор укрупненных команд, с помощью которых можно задавать графические примитивы, такие, как линия, дуга, окружность, многоугольник, знак, одной командой.

С помощью сопроцессора можно управлять не только графическим монитором, но и лазерным принтером. Сопроцессор воспроизводит рисунок со скоростью 2 млн. цветных точек (пиксел) в секунду с разрешением 33 млн. черно-белых или 4 млн. цветных точек на экран. Ни один универсальный процессор так быстро с такой работой не справится.

Знания — по почте

Нет, наверное, менеджера, который не согласился бы потратить время на изучение тонкостей науки управления. Но, как и все деловые люди, менеджеры — народ занятой и слушать курс в учебном заведении (как у нас говорится, с отрывом от производства), как правило, возможности не имеют. Чтобы помочь им, а также людям, которые хотят освоить азы науки управления, специалисты американского колледжа менеджеров разработали пакет обучающих программ, позволяющий овладеть менеджментом, не прибегая к помощи преподавателей.

Компьютерный курс состоит из одиннадцати модулей, включая заключительный экзамен, и рассчитан на 30—40 ч работы с персональной ЭВМ типа IBM PC или Zenith 120. Причем последовательность изучения модулей может быть в определенной степени выбрана будущим менеджером. Пакет содержит базовый курс менеджмента, а также модули, которые позволяют освоить технологию обработки данных, составление финансовых отчетов и т. д, и дают не только необходимые знания будущим менеджерам, но и могут служить инструментом в дальнейшей работе.

Аттестация менеджеров, изучивших предлагаемый курс, проводится в колледже, выславшем обучающий пакет. Курс записан на флоппи-дисках и сопровождается документацией по его использованию, методическими пособиями и другим необходимым материалом.

ЭВМ для любителей математики

«...Пьеру было открыто одним из братьев-масонов следующее, выведенное из Апокалипсиса Иоанна Богослова, пророчество относительно Наполеона...» Это не фрагмент трактата по литературоведению, разбирающего сложный и противоречивый характер Пьера Безухова. Отрывок взят из третьей главы под названием «Структуры данных и их реализации» учебного пособия, вышедшего недавно в издательстве «Наука»[1]. Интересно, что великий Л. Н. Толстой на страницах «Войны и мира» рассказывает по существу о реализации хорошо известной программистам хэш-функции: «...Написав по этой азбуке цифрами слова L'impereur Napoleon, выходит, что сумма этих чисел равна 666-ти и что, поэтому Наполеон есть тот зверь, о котором предсказано в Апокалипсисе. Кроме того, написав по этой азбуке слова quarante deux, то есть предел, который был положен зверю глаголати велика и хульна, сумма этих чисел, изображающих quarante deux, опять равна 666-ти, из чего выходит, что предел власти Наполеона наступил в 1812 году, в котором французскому императору минуло 42 года...» Авторы предоставляют самому читателю «проверить алгеброй гармонию» и на практике познакомиться с одним из эффективнейших способов кодирования текстовой информации. Подобных примеров с экскурсом в гуманитарные науки в книге немало.

Как учить программированию? Как правильно расставить акценты в излагаемом материале? Какую ЭВМ и какой язык программирования принять за основу?

Подобные вопросы сразу же встали перед авторами пособия. Академик А. П. Ершов не раз отмечал, что в организации компьютерного всеобуча в нашей стране ситуация сложилась (причем не только в средней школе) весьма своеобразная. Мы знаем, *кого* учить, знаем, *что* надо учить, но еще не знаем, *как* учить наиболее эффективно. Авторы поставили перед собой стратегическую цель — показать, как учить, чтобы заложить фундамент общей программистской культуры читателя. По мнению авторов, освоив пособие, читатель научится грамотно программировать задачи объемом до нескольких тысяч строк.

В основе курса лежат три «кита»: во-первых, понятие «исполнитель», во-вторых, технология программирования «сверху вниз» и, в-третьих, развитые «структуры данных». Основной методический прием, используемый в книге,— обучение на примерах. Любые технические детали изучаются лишь в процессе решения конкретной содержательной задачи. Примечательная особенность — курс можно изучать и без ЭВМ. Ну, и естественно, наличие персонального компьютера и даже элементарного электронного калькулятора существенно повышает эффективность и качество обучения. Значительная часть пособия посвящена разбору «мини-проектов», т. е. законченных систем объемом в несколько сотен строк. В их числе регулировка работы станка с числовым программным управлением (ЧПУ), управление движением «лунохода», реализация простейшего компилятора арифметических формул, графическое построение многогранника (полиэдра) с удалением невидимых линий, реализация простейшей файловой системы, экранный редактор текстов и многое другое, что несомненно заинтересует юного программиста.

Другая особенность пособия — в первых четырех главах программы записываются на русском языке, что снимает у начинающего пользователя психологический барьер и позволяет читателю осваивать курс программирования гораздо быстрее. Заключительная пятая, глава посвящена практике работы на Фортране. Выбор этого языка объясняется тем, что он наиболее распространен в сфере научно-технических расчетов, особенно в системах автоматики и компьютерной обработки инженерных данных. На конкретных примерах показано, как можно на деле применять стиль, методы и, еще раз подчеркнем, культуру программирования, т. е. то, о чем шел обстоятельный разговор в предыдущих главах.

Книга адресована студентам, но может быть с успехом использована (и уже используется) школьниками, в частности в кружковой работе. Книга, несомненно, заинтересует и широкие круги читателей, особенно тех, кто видит в ЭВМ нечто большее, чем пресловутый «черный ящик», и стремится стать архитектором ее электронных дворцов.

В. ФРОЛОВ

[1] *Кушниренко А. Г., Лебедев Г. В.* Программирование для математиков. М.: Наука, 1988.

Журнал в журнале
для школьников, студентов,
учащихся СПТУ и техникумов

Издается при участии ЦК ВЛКСМ

МОЛОДЕЖНАЯ ИНИЦИАТИВА

2

В НОМЕРЕ:

Компьютер не главное

Прошедшее лето кажется мне особенным. В августе 1988 г. в подмосковном местечке Поварово открылся Второй советско-американский лагерь «Heart to Heart» («От сердца к сердцу») программы «Дети — творцы XXI века», в котором мне довелось отдыхать. Среди спонсоров лагеря — советский Фонд мира, советский Красный Крест, БММТ «Спутник», МААЗ, любезно предоставивший нам лагерь.

Если попытаться в двух словах обозначить программу лагеря, то я позволю себе привести слова десятиклассника Димы Царевского: «Познание через общение».

Наш лагерь не был сугубо компьютерным, он включал множество студий: изобразительную, экологическую, театральную, компьютерную, музыкальную и др.

О двух из них хотелось бы сказать особо. В ЭДАСе (эстетическая, детская, архитектурная студия) картину «Мосты» рисовала половина лагеря. А в студии «Юные учителя здоровья» (руководитель Лазарев) я принимала самое непосредственное участие. И если у Кирпичева (руководитель первой студии) лагерь работал, творил, то у Лазарева — лечился.

Настоящим праздником была постановка театральной студией советско-американского спектакля «The Cat that walked by himself» где актерами и оформителями были дети. Авторский текст на английском языке читал советский школьник 12-летний Митя Кондрашов (Пущино). Спектакль имел успех, и, говоря «высоким штилем», овации не утихали, и актеров долго не отпускали со сцены.

В программу нашей жизни входили многочисленные экскурсии: в Москву, в Переславль-Залесский, Клин, Волоколамск и Новый Иерусалим, где нас принимал митрополит Питирим.

Вскоре после открытия в лагере начали выпускать газету «События и размышления». Она была полностью организована и укомплектована сотрудниками в возрасте от 14 до 17 лет. Всего вышло девять номеров. В № 7 говорилось: «Один экземпляр «Событий и размышлений» на черном рынке равнозначен подписке на «Московские новости» на русском языке». Компьютер и принтер любезно предоставлены в распоряжение пресс-центра академиком Е. П. Велиховым.

Заключительным аккордом августовской жизни стал праздник на Арбате, где состоялись импровизированная пресс-конференция, танцы, пение. Это, по словам многих участников, стало «звездным часом» лагеря.

В заключение мне хочется процитировать слова директора программы «Дети — творцы XXI века»: «Лагерь следующего года нужно обсудить сейчас, чтобы в разработке наших планов приняли участие не только взрослые, но и дети. Мне хочется, чтобы дети продумали его концепцию и участвовали в его организации» («События и размышления», № 8).

Юля ШТУТИНА,
VIII класс
СШ № 1268, Москва

Что такое «Юность»?

Этот материал был написан под градом телефонных звонков. Каким образом? Появилась в середине июня в «Рекламном приложении» к «Вечерней Москве» коротенькая заметка о Центре компьютерного творчества молодежи «Юность», и, отвечая на многочисленные вопросы, мы не поленились включить магнитофон, а потом обработать запись.

Звонок.

— Я преподаю программирование в вузе, сам разрабатываю программы. Как мы могли бы сотрудничать?

— Наш центр в своей деятельности имеет четыре ос-

новных направления. Первое. Разработка программного продукта для персональных ЭВМ отечественного производства (ну и чуть-чуть импортного). Второе. Ведение учебных курсов по разным разделам информатики и вычислительной техники. Главным образом это курсы для начинающих. Третье. Досуговый центр для детей, или, по-простонародному, компьютерная игротека. И четвертое. Широкая продажа программной, методической и информационной продукции.

Так что вас мы можем привлечь к ведению учебных занятий или разработке программ для персоналок.

— Вы — хозрасчетная организация?

— Да. Мы работаем по второй модели хозрасчета и являемся специализированным (компьютерным) научно-техническим центром творчества молодежи.

Звонок.

— Я из Урюпинска. Купил компьютер БК-0010...

— Поздравляем!

— А программ к нему нет...

— Соболезнуем, но можем помочь. У нас вы сможете приобрести игровые, системные или прикладные программы, например, для научных и инженерных расчетов.

— А с каталогом ознакомиться можно?

— Каталог стоит 3 рубля.

— Согласен.

— Диктуйте адрес, получите через неделю наложенным платежом.

Звонок.

— У вас можно купить компьютер?

— Нет.

— Жаль, а кто ваша вышестоящая организация?

— Наш центр был создан совместным решением Секретариата ЦК ВЛКСМ, Президиума АН СССР, Главного комитета ВДНХ СССР и Государственного комитета СССР по вычислительной технике и информатике, а затем по согласованию с этими организациями Всесоюзный координационный совет НТТМ (а это орган, которому делегированы права Совета Министров) включил центр «Юность» в систему НТТМ и присвоил ему статус специализированного центра НТТМ. Короче, нашими вышестоящими организациями являются ЦК ВЛКСМ и ВКС НТТМ.

Звонок.

— Вы можете заключать хоздоговоры с организациями?

— Конечно, как любое государственное предприятие.

— Наш завод хотел бы заказать разработку справочно-информационной системы на машине ДВК, «Электроника-85».

— Готовьте техническое задание и приезжайте. Мы подберем коллектив, который выполнит задачу. Если не можете сами подготовить техзадание, то мы пришлем специалиста, который проведет обследование и вместе с вами составит ТЗ, которое можно вручать программистам.

— А по каким расценкам проводится эта работа?

— По договорным, поскольку Постановление ГКНТ СССР № 435 от 19 ноября 1987 г. на нас распространяется.

Звонок.

— Звонят студенты МИЭТ. У нас каникулы, хотели бы поработать, программы писать...

— А программировать вам доводилось?

— Естественно.

— Приезжайте. Что-нибудь подыщем для вас в отделе разработок.

Звонок.

— Говорят у вас есть учебные курсы. Я хотел бы обучиться языку Бейсик.

— Прекрасно! У нас как раз есть 36-часовой курс «Бейсик для начинающих». Приезжайте и записывайтесь в удобную для вас группу.

— Сколько это будет стоить?

— Час занятий — три рубля.

— Сто восемь рублей за курс! Не дорого берете?

— Репетитор взял бы в два с половиной раза дороже. Это во-первых. А во-вторых, вы ведь, перед тем как платить 108 рублей, 108 раз подумаете, нужен ли вам курс Бейсика на самом деле. И если решите, что нужен, отнесетесь к занятиям более серьезно и уж наверняка его изучите.

Звонок.

— Беспокоит руководитель экономического отдела

одного объединения. Мы купили 6 ЭВМ «Роботрон». Вы не могли бы обучить работе на них моих сотрудников, да и меня заодно?

— Вам ведь, наверное, нужен какой-то специализированный курс?

— Да! Что-нибудь с экономическим уклоном.

— Хорошо, мы подыщем преподавателя, затем заключим с вашей организацией договор на обучение. Цены на обучение согласуем в договоре.

Компьютерный центр «Юность» действует.

Центр компьютерного творчества молодежи «Юность» предлагает учебным заведениям и другим организациям, использующим в своей работе КУВТ — YAMAHA MSX, КУВТ-86, Каталог программного обеспечения:

А. Система проектирования машинных игр, мультфильмов и учебных пособий. 11 программ

Б. 1. Алгоритмы и программы элементарных вычислений. 6 вариантов.

2. Программы реализации основных численных методов. 9 вариантов.

3. Спектральный, статистический, корреляционный и регрессивный анализ. 6 вариантов.

4. Вычисление специальных функций. 4 варианта.

5. Прикладные программы технических и экономических расчетов. 8 вариантов.

6. Расширение библиотеки научных программ для персональных ЭВМ IBM PC. 7 пакетов программ.

Центр также предлагает мультикомпьютерную игру «Зоорали II». Игра заключается в том, чтобы, управляя выбранным животным, в ходе гонок привести его к финишу первым. Если играющий проходит все этапы выигрывая, то появляется поощрительная мультзаставка.

Адрес центра: 129223, Москва, проспект Мира, ВДНХ СССР, павильон МОВ № 3, Центр компьютерного творчества молодежи «Юность», тел. 187-03-38.

Характеристику выдает ЭВМ

Что мы обычно связываем с управлением? Пустую трату нервов? Горы ненужных бумаг? Валидол в верхнем ящике стола? А ведь между понятиями «управлять» и «управлять квалифицированно» есть разница. Эффективность производственного процесса во многом зависит от способностей и навыков руководителя, его умения правильно подобрать и расставить кадры, максимально реализовать творческий потенциал подчиненных. Первая в Эстонии таллинская «Школа молодых руководителей (менеджеров)» учит управлять профессионально.

«Школа» образована в 1983 г. в Центре управления и комплектации Минстроя республики по инициативе группы энтузиастов. Поначалу носила скорее самодеятельный характер. Молодые специалисты (их было большинство) постепенно завоевывали авторитет.

В январе 1985 г. в Таллине прошел первый конкурс по отбору перспективных кандидатов в руководители. (Попутно замечу, что теперь такие конкурсы стали в Эстонии традиционными, переросли в настоящую ярмарку управленческих кадров, на которой руководители предприятий подбирают себе резерв. А заявки на проведение таких конкурсов поступают сегодня практически со всех концов страны.) Популярность «Школы» росла. В 1987 г. при поддержке горкома комсомола и ЦК ЛКСМ Эстонии «Школа менеджеров» (под таким названием она известна всюду) получила официальный статус, а вместе с этим возможность вести хозяйственную деятельность, на средства от которой и были приобретены первые компьютеры. Их стали применять, в частности, при составлении характеристики претендента на руководящую должность.

Создается характеристика так. Когда позади обучение и конкурс, проходит аттестация — сослуживцы оценивают друг друга по специальным вопросникам, разработанным в «Школе» с учетом всех требований к руководителю как специалисту и человеку. На одного аттестуемого разными людьми заполняется несколько характеристик-анкет, что позволяет компьютеру в конечном счете быть максимально «объективным».

Вопросы в анкетах, которые вводятся в машину, разработаны так, что при ответе можно указать одну из нескольких степеней какого-либо качества. Аттестующий выписывает шифр, а баллы выставляет сама ЭВМ. Учитывая все критерии, она выводит среднюю оценку, отбрасывая при этом все нереальные. Если, скажем, по какому-то критерию аттестуемого никто не оценил, он в характеристику не попадает, а машина при этом обращает внимание на то, что, видимо, с этой стороны сотрудник недостаточно проявил себя в коллективе.

На следующем этапе характеристики вводятся в машину по группам — цех, специальность, должность. Когда, скажем, требуется выявить лидера по специальности, отбрасываются все оценки, кроме деловых.

В конечном итоге характеристики, выдаваемые ЭВМ, не только точны и объективны, но еще подробны и в то же время лаконичны.

В сентябре этого года в Таллине при Государственном комитете по труду ЭССР на базе «Школы менеджеров» был создан государственный центр по оценке и подбору управленческих кадров «Юхт», что в переводе с эстонского означает «руководитель».

Как говорит его директор Г. Блинчук, и в дальнейшем в объединении планируется как можно шире применять компьютеры в деле подготовки специалистов-руководителей и выявлении претендентов на руководящую должность.

Т. КРАСНОВА
студентка МГУ

Встреча с юными американцами

Маленький кабачок. Скрипят двери. Толстый бармен за стойкой лениво бросает слова таперу; тот сидит за разбитым пианино, что-то наигрывает. На клавишах — свет оплавленных свечей.

Так начинаются самые невероятные истории, случаются неожиданные встречи. Этот кабачок мал и уютен и помещается на сцене. Обычной школьной сцене. А завсегдатаи «салуна»: ковбои, актрисы, искатели приключений, гангстеры — ребята из московской школы № 1155.

Какая встреча случится сегодня?.. Двери зала открыты, звучит песня: «Америка, Америка...» Наши гости — из Калифорнии, Сан-Франциско, Нью-Йорка. Многим из них нет еще шестнадцати, они члены Ассоциации юных астронавтов. Во главе делегации астронавт Дэвид Уокер.

На стене — изображение звездно-полосатого флага. Аплодисменты. Тапер садится за пианино, начинается действо — спектакль по мотивам рассказов О'Генри.

Пока идет представление, мы подробнее расскажем о наших гостях. Ассоциация юных астронавтов существует в США уже двадцать пять лет. А наши юные исследователи космоса объединились совсем недавно. Они и пригласили американцев, как у нас говорят, для обмена опытом. И тем познакомиться хотелось, конечно. Со страной, со сверстниками. Поэтому после официальных встреч ребята с удовольствием приняли приглашение политклуба «Единство» московской школы № 1155 и зашли в их театральный «кабачок». Сейчас мы вернемся туда и узнаем дальнейшую программу.

Соревнование-игра за компьютером. Можно ли удивить американских школьников, таких искушенных в информатике, нашим компьютерным классом? Удивить — нет, а заинтересовать — да.

«Тетрис» — сюрприз хозяев. С ней американцы незнакомы. Не успели познакомиться. Программа создана молодыми специалистами МГУ. Она запатентована и уже продана в Англию.

Итак, мы ведем наш репортаж с компьютерного матча молодежных команд СССР — США. Может быть, первого? Несколько минут до начала. Волнуются организаторы. Не за «Тетрис», нет, за гостей. Игра незнакомая, а нашим ребятам дома, как говорится, и компьютеры помогают. Появились соперники. Сначала небольшая разминка. Зря мы переживали за американцев. Самые юные в команде, тринадцатилетние Сара Дорфф и Майк Маст, с первых секунд заиграли так, что сумма баллов не успевала фиксироваться компьютером. Игра захватила всех. Ребята не замечают ничего вокруг. Смех. Шум. Глаза горят. Видно, «Тетрис» нравится.

Во время матча мы подошли к Дэвиду Уокеру. Вот что он сказал:

— Дельная игра,— улыба-

ется.— Думаю, она нужна и детям, и астронавтам. Развивает логическое мышление и хорошую реакцию. И еще. Посмотрите, как они понимают друг друга — как в хорошем экипаже. Честное слово, не разобраться, кто здесь русский, кто американец. И это хорошо.

Матч закончен. Погасли экраны компьютеров. Сделаны снимки на память. В блокнотах новые адреса. Расставание...

В. БУЛДАКОВ,
И. СЕЛЕЗНЕВА

Пример, достойный подражания

Клуб «Информатика» в гостях у сельских школьников

Для множества людей компьютер — давно не новый вид информационной техники — пока что тайна за семью печатями. Такое положение дел не редкость даже в крупных городах, хотя там в последние годы все чаще и чаще возникают компьютерные центры, клубы, кооперативы. Но это в городах. А кто даст возможность сельским школьникам своими руками поработать на «персоналках», на практике освоить азы программирования, кто поможет им ощутить для себя преимущества нового дела, которое входит в жизнь страны? Ведь и они через десяток лет сменят взрослых в науке, на производстве, в сельском хозяйстве.

В какой-то степени помочь решить эту проблему призван «передвижной компьютерный центр», созданный членами клуба «Информатика» В. Дубасом, Р. Куком и М. Могилевским в г. Львове. Пока что в центре — всего три персональных компьютера «Атари-65XE», у которых есть клавиатура и «джойстик» — ручной манипулятор, в качестве внешней памяти используются кассетные магнитофоны, а дисплеев — советские цветные телевизоры «Электрон 382Д — И».

Первым стал выезд в село Яблоневка Стрыйского района Львовской области. В этот день областной комитет ЛКСМ Украины открывал в селе новый культурный комплекс, в открытии принял участие и клуб «Информатика».

...Большая афиша на двери одного из классов Яблоневской средней школы сразу привлекала к себе внимание. «Компьютерный клуб «Информатика» приглашает всех в увлекательный мир видеоигр. У нас вы можете создать оригинальные графические изображения, написать с помощью компьютера музыкальные мелодии, решить игровые задачи...»

В классе — оживленная обстановка. Компьютерные игры? А почему бы и нет? Ведь они дают возможность «подружиться» с компьютером, сразу войти с ним в контакт. Так и закладываются первые навыки общения с ЭВМ, отступает робость перед «неведомой» техникой. А если возникнут проблемы — М. Могилевский и Р. Кук готовы прийти на помощь, дать любые пояснения.

— Мы привезли с собой около 25 игровых программ,— рассказывает руководитель клуба «Информатика» В. Дубас.— Конечно, играми обучение не ограничивается, однако вызвать заинтересованность, желание научиться новому они помогут. Беда в том, что возможностей для этого у сельских школьников куда меньше, чем у городских...

Эта проблема достаточно острая, и поэтому инициатива клуба «Информатика» — пример, достойный подражания и безусловной поддержки.

Жаль только, времени для работы в этот раз было немного — всего лишь полдня. Надо было видеть, с какой неохотой отрывались ребята от светящихся всеми цветами радуги телевизоров.

Когда-то они еще увидят такое?

И все-таки сейчас новое дело уверенно идет вперед.

А начиналось все так...

Двери компьютерного клуба «Информатика» открылись для жителей Львова в июне прошлого года. С интересом рассматривали львовяне маленькие «Атари-65XE», позволяющие создать высококачественные цветные изображения, быстро сменяющие друг друга на экранах телевизоров. За небольшую плату желающим предоставлялась возможность самостоятельно поработать с компьютером, решать свои задачи, попробовать силы в составлении программ.

Первые посетители ограничивались в основном играми. Поначалу создатели клуба ориентировались на них и ставили перед собой простую задачу — помочь желающим освоить персональные компьютеры, показать их возможности, а где необходимо — и доказать эффективность применения «электронного помощника».

Но с началом работы клуба

открылись и новые перспективы. За помощью стали обращаться школы и даже некоторые предприятия. Ведь для многих выгоднее покупать машинное время, чем приобрести сам компьютер.

С появлением новых заказчиков изменился и распорядок работы клуба. До обеда здесь проходят занятия со школьниками по предмету «Основы информатики и вычислительной техники». «Вшитый» в ПЗУ «Атари» интерпретатор Бейсика позволяет наглядно демонстрировать учащимся и работу их собственных программ, и возможности более сложных, составленных членами клуба.

Различные обучающие программы стали надежными помощниками для учителей. Сейчас, например, около 15 школ Шевченковского района г. Львова заключили с кооперативом «Лотос», на базе которого создан клуб, договоры на весь учебный год. Преимущества «машинного способа обучения» неоспоримы.

Но вот закончилась первая половина дня. Теперь наступило время индивидуальных посещений, во время которых, как сообщает афиша, предоставляется возможность участвовать в решении игровых задач и создании оригинальных графических изображений, сочинять музыку и изучать языки программирования. А те, кто пришел сюда с учебными или чисто деловыми целями, могут использовать персональный компьютер для решения математических, физических или же экономических задач.

Но больше всего среди гостей клуба школьников. И с сожалением смотришь на некоторых взрослых, считающих компьютер чуть ли не детской забавой. Даже придя в клуб, они не садятся за клавиатуру, а лишь наблюдают за детьми.

Как полезно было бы для них организовать показ коротких видеопрограмм, рассказывающих о возможностях применения электронно-вычислительной техники! А еще лучше было бы привлечь к этому телевидение. Скажете, мелочь? А ведь за несколько месяцев существования клуба его посетили свыше 4 тыс. человек...

Не все проблемы «Информатики» связаны только с рекламой. Возникают серьезные трудности и с техническим обеспечением. У клуба пока еще нет ни принтеров, ни графопостроителей, и часто, как это ни парадоксально, приходится сидеть перед экраном терминала с листком бумаги и карандашом в руках. В качестве внешних носителей памяти используются обычные магнитофонные кассеты. В. Дубас поделился своей мечтой о кассетах с цифровой записью информации, а еще лучше о переходе на дисководы.

— Трудностей на первых порах хватает, но энтузиазма мы не теряем,— говорит он.— Главное то, что нас поддерживают. Большую помощь в организации клуба оказал первый секретарь областного комитета ЛКСМУ В. Н. Ковальчук, постоянно курирует нашу работу заведующий отделом науки обкома комсомола О. Наливайко.

Во Львове давно уже существует более десятка видеосалонов. Но они призваны только развлекать посетителей. Иная задача стоит перед «Информатикой» — учить людей обращаться с компьютерами, дать им первые навыки в работе, в конечном счете — открыть для всех интереснейший мир компьютерной техники. И в заключение адрес клуба:

г. Львов, ул. Торфяная, 20, компьютерный клуб «Информатика». Телефон 59-04-43. Время работы с 10.00 до 23.00.

Плата за один сеанс — 1 руб.

Н. ГАФУТУЛИН, И. ПАЩЕНКО,
курсанты 3-го курса
факультета журналистики
ЛВВПУ

Информатика и проблемы молодежи

В эстонском городе Нарве 10—17 ноября 1988 г. на территории международного молодежного центра «Ноорус» проходила Всесоюзная школа молодых ученых и специалистов «Информатика и молодежь» — по социальным и организационно-методическим проблемам любительского компьютерного движения в нашей стране.

В работе школы приняли участие 116 человек из 52 городов страны — молодые ученые и специалисты, разработчики программных и аппаратных средств ЭВМ, преподаватели вузов и школ, руководители компьютерных клубов, директора центров НТТМ, председатели советов молодых ученых и специалистов. Приехала и делегация из Народной Республики Болгарии во главе с Д. Маламовым.

Работа школы велась по следующим темам: социально-экономические проблемы молодежи в условиях информатизации общества; комсомол, молодежь и компьютерные объединения; проблемы информатизации образования, досуга и быта молодежи.

Обсуждались различные

социальные аспекты информатизации в молодежной среде, проблемы использования ЭВМ в различных сферах деятельности человека, проблемы компьютерного обучения и издательской деятельности в условиях информатизации. Было отмечено, что широкое внедрение компьютеров имеет огромное экономическое и политическое значение для СССР. При этом, глубоко проникая во все сферы человеческой жизни, информатизация становится и важным социальным явлением, требующим нового подхода ко многим общественным проблемам.

В связи с этим ощущается большая потребность в изучении социальных аспектов любительского компьютерного движения в частности. Часть руководителей компьютерных клубов и объединений, существующих уже несколько лет, подчеркивали, что недоучет социальных факторов приводит к серьезным трудностям в работе клубов (обучению с целью заработать деньги или, наоборот, к поверхностному овладению ЭВМ для игр). В этом плане проблемы любительского компьютерного движения практикой своей работы подтверждают правильность идей, прозвучавших на I Всесоюзном симпозиуме по социальным и методологическим проблемам информатики (см.: НТР. 1988. № 22. С. 3). Социальные, в том числе моральные, этические, мировоззренческие, культурно-ценностные, аспекты информатизации должны быть изучены и определены прежде, чем принимать крупные государственные проекты по широкому распространению ЭВМ.

На школе была предпринята попытка выявить принципы и главные направления деятельности компьютерных объединений, клубов и ассоциаций; наметить конкретные меры по объединению и интеграции компьютерного движения в условиях интенсивного распространения ЭВМ в учебной и трудовой деятельности молодежи, в острой конкуренции любительского движения с создаваемыми ведомственными системами центров информатики и компьютерных клубов.

В выступлениях представителей ВОКК (Всесоюзного объединения компьютерных клубов), МАКО (Московской ассоциации компьютерных объединений), ЦКТМ «Юность» (центра компьютерного творчества молодежи) и других участников подчеркивалось, что потенциал любительских и коммерческих компьютерных клубов, объединяющих энтузиастов информатизации, используется не в полной мере. Между клубами все еще нет достаточной взаимосвязи и координации действий.

Обсуждение деятельности Всесоюзного объединения компьютерных клубов (ВОКК), созданного на неформальной основе, показало, что оно находится в начальной стадии своего развития и пока не в состоянии решить всех своих проблем. На школе был обсужден проект нового Устава ВОКК и намечен план дальнейшей работы объединения.

Был проанализирован проект ЦК ВЛКСМ по созданию сети центров компьютерного творчества молодежи «Юность», в том числе вопросы, касающиеся его хозрасчетного характера, ориентированного преимущественно на коммерческую деятельность и излишне централизованного управления. Отмечалось, что Всесоюзному ЦКТМ «Юность» следует больше внимания уделять вопросам информационно-методического обеспечения клубов, обобщения опыта работы, фондовой и издательской деятельности, а также организации Всесоюзной заочной школы по информатике.

Представители Центра информатики ЦК ДКСМ Болгарии рассказали об опыте организации сети клубов «Компьютер» у себя в стране, о формах и методах участия ДКСМ в общегосударственной программе компьютеризации.

Участники Всесоюзной школы «Информатика и молодежь» подробно ознакомились с научными взглядами на проблемы информатизации общества. Были намечены возможные пути совместной работы компьютерных клубов СССР и НРБ. Предложено провести социологический анализ деятельности компьютерных клубов и социальных последствий информатизации в молодежной среде, наладить рабочие контакты с секцией «Социальные проблемы информатизации общества» при Философском обществе АН СССР.

Участники школы не ограничились обсуждением научно-практических проблем. В рамках культурной программы были организованы экскурсии в Таллинн, Тарту, Нарву. Прошла интересная дискуссия с представителями Народного фронта Эстонии, встреча с профессором С. Микояном и специальным корреспондентом журнала «Огонек» Ф. Медведевым.

В. ЧЕРЕДНИЧЕНКО

И снова «голос» домашнего компьютера

Прежде чем передать слово С. Ивашинникову, который предлагает простейшую, но довольно эффективную говорящую программу для БК, вернемся к статье «Здравствуйте, говорит «Электроника БК-0010» А. Чистякова (см.: Информатика и образование. 1988. № 4). В тексте статьи допущена опечатка — вместо 1026А2000 G должно быть 1026А2000И. Благодарим читателей, указавших на это.

Программу вводят в память БК с адреса 1000. Прежде чем запускать ее, обнулите ячейки: 1110А36000 DOP. Только теперь включите в режим записи магнитофон, подключенный к БК, и отрегулируйте уровень записи так, чтобы магнитофон хорошо работал с микрофоном. Запустив программу командой 1000 G, продиктуйте как можно более четко короткую фразу, например: «работает программа речевого синтезатора БК». Сразу же вслед за этим остановите исполнение программы, нажав клавишу СТОП. Все готово, и командой 1044 G можно запустить синтезатор. Фраза, которую мы только что начитали, начнет повторяться до тех пор, пока вновь не будет нажата клавиша СТОП. Звук будет слышен и через пьезодинамик БК, но лучшего качества нетрудно добиться, подключив к БК тот же магнитофон. Память БК способна вместить и более длинные фразы, если разблокировать ее клавишами НР/РП. Впрочем, есть возможность обменять время работы синтезатора на качество звучания. Для этого нужно исправить всего четыре команды: 1026А 110420, 104ОА 11042О, 1060АИ151004, 1070АИ152004.

Как же работает программа? Информация, считываемая с магнитофона, записывается в память БК словами (или байтами) с 1102 адреса, а затем последовательно за байтом воспроизводится.

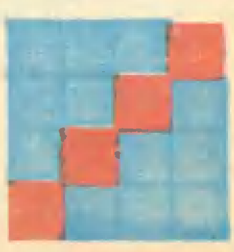
С. ИВАШИННИКОВ
г. Арсеньев, Приморский край

12703	177716	12700	70	60700	12702	40	5004
5204	30213	1775	10420	5004	5204	30213	1375
10420	765	12700	30	60700	12737	100	177716
11004	77401	5037	177716	12004	77401	5710	1365
761	0	0	0				

Радуга на экране БК

Счастливцы, которым удалось подключить свой компьютер к цветному телевизору, знают, что Бейсик позволяет распорядиться всего четырьмя различными цветами — красным, зеленым, синим и черным. Оператор PAINT Бейсика-МСХ позволяет окрасить в заданный цвет любую область экрана, ограниченную замкнутым контуром. В черно-белом варианте красный цвет выглядит белым, синий и зеленый — серым. Можно ли увеличить число цветов на экране, если архитектурой БК предопределено, что на задание цвета отводится два бита памяти? Если речь идет о цвете окраски, то выход есть. Кировчанин С. Зильберштейн, автор известной программы Х Фокал, предлагает получать на экране БК 16 различных цветовых оттенков программным способом.

«Электроника БК-0010.01» действительно «знает» всего четыре цвета, но раскраска и цвет совсем не одно и то же. Раскраска вовсе не обязательно должна быть однородной. Вспомните телеэкран: три расположенные рядом точки — красная, синяя и зеленая — воспринимаются как белый цвет, а различные их комбинации — как множество оттенков. Так и на экране БК в качестве цвета можно вообразить, например, чередующиеся в шахматном порядке красные и синие точки, какую-либо штриховку или периодический рисунок. В качестве минимального элемента раскраски можно выбрать квадратик размером 4×4 точки экрана. Изменяя его кодировку, можно будет в итоге определять цвет раскраски большого фрагмента экрана. Чтобы пояс-



```
10 '  ПРОГРАММА PAINTM  SZ/08-88
20 '    (16-ЦВЕТНАЯ РАСКРАСКА)
30 ' ---------------------------
100 DATA 5599,16222,28,9111,-21324,516
110 DATA 5623,-30696,60,265,9111,-21304
120 DATA 1017,9111,-21296,539,5623,-30694
130 DATA 40,6118,140,3039,192,783
140 DATA 4262,8599,304,-32252,5599,-1
150 DATA 140,-30457,17858,-4,26562,192
160 DATA -28001,140,5506,-30696,5535,140
170 DATA 2,95,-32694,95,-13052,11671
180 DATA 4,-13672,518,5585,16236,503
190 DATA 2591,192,92,9111,-21894,523
200 DATA 9622,-6720,5,3264,3264,26048
210 DATA 256,4127,192,2574,92,9111
220 DATA -13450,739,9622,-23337,67,513
230 DATA -30392,-30462,-23337,61,764,-30377
240 DATA -6715,5,3269,3269,26053,256
250 DATA 5606,3,2551,18,-30378,2766
260 DATA 7636,3030,2551,6,-30376,-30394
270 DATA 135,2598,5570,4,3214,3214
280 DATA -27452,-6716,52,2820,8471,3
290 DATA -32036,196,3204,3204,20750,32398
300 DATA 5508,-28395,135,0,0,0
400 CLEAR  ,&O37400
410 FOR K=0 TO 250 STEP 2
420   READ CODE
430   POKE &O37400+K,CODE
440 NEXT
450 POKE &O30,&O37400

100  ' CIRCLE - СЕКТОРНАЯ ДИАГРАММА ДЛЯ 8 ВЕЛИЧИН
120 DEF C5=(3333,4444,3333,4444)
130 DEF C6=(3434,4343,3434,4343)
140 DEF C7=(3444,4444,4434,4444)
150 DEF C8=(4443,4434,4344,3444)
160 CLS
170 SUM=0
180 PRINT "ЗНАЧЕНИЯ:"
190 FOR K=1 TO 8
200   PRINT "V";MID$(STR$(K),2,2);
210   INPUT V(K)
220   SUM=SUM+V(K)
230 NEXT
240 R=100
250 A2=0.00001
260 FOR K=1 TO 8
270   A1=A2
280   A2=A2+2*PI*V(K)/SUM
290   CIRCLE (140,120),R,1,-A1,-A2,4/3
300   XP=R/2*COS((A1+A2)/2)
310   YP=-R/2*SIN((A1+A2)/2)
320   PAINT @(XP,YP),K,1
330 NEXT
```

Достаточно один раз запустить эту программу, чтобы модифицировать Бейсик до следующего включения питания. Программу, разумеется, можно стереть командой NEW. Сказанного вполне достаточно, чтобы самостоятельно проделать первые эксперименты с добавочными цветами, но все же приведу еще один пример. Программа CIRCLE определяет четыре дополнительных цвета и строит после ввода с клавиатуры восьми числовых значений раскрашенную в восемь цветов секторную диаграмму. Кстати, попробовать могут и владельцы черно-белого варианта — цветам с 5 по 8 соответствуют горизонтальная штриховка, мелкая сетка, отдельные точки и диагональная штриховка.

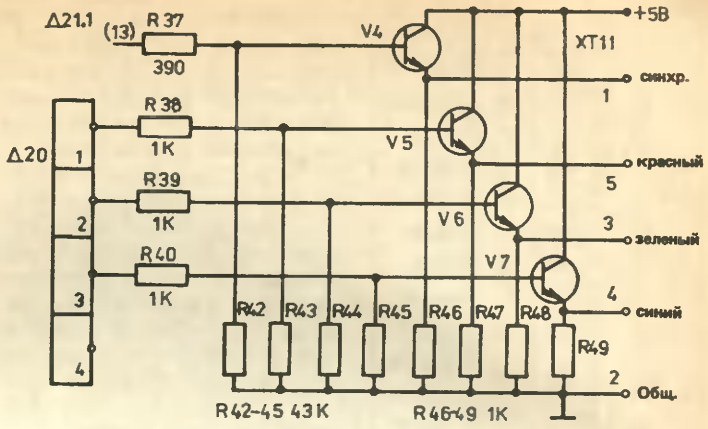
С. ЗИЛЬБЕРШТЕЙН,
г. Киров

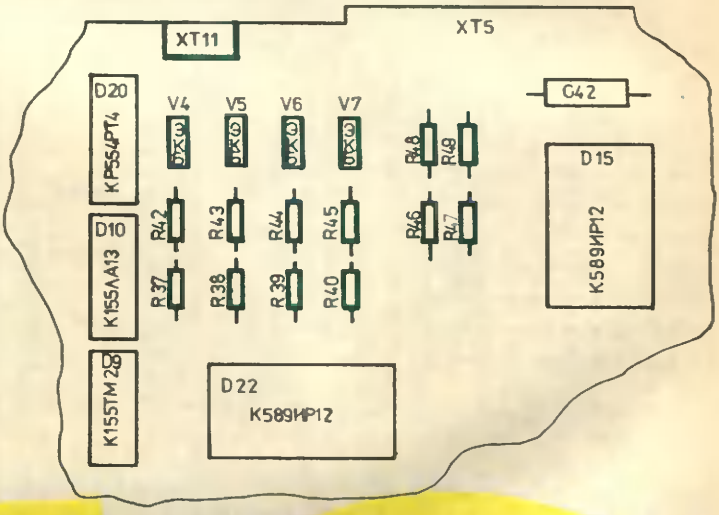
Программный способ, предложенный С. Зильберштейном, весьма интересен, но есть и иной вариант. Владельцы БК, знакомые не только с программирова-

нить сказанное, приведу лишь один пример — матрица, изображенная на рисунке, соответствует диагональной штриховке синим цветом по красному полю.

Если записать матрицу в цифровом виде (каждому цвету соответствует цифра от 1 до 4), то получится строчка — (4443, 4434, 4344, 3444). В развитых версиях Бейсика оператор PAINT позволяет указывать и матрицу раскрашивания. В Бейсике-БК такой возможности нет, но ее можно ввести. Для этого служит программа PAINTM. Запустив ее, вы модифицируете Бейсик так, что оператор PAINT будет воспринимать не только цвета от 1 до 4, но и от 5 до 16. При этом определения псевдоцветов 5—16, т. е. соответствующие матрицы раскраски, программист вводит с помощью операторов DEF. Формат оператора легко запомнить: DEF CN=(ЦЦЦЦ, ЦЦЦЦ, ЦЦЦЦ, ЦЦЦЦ). Здесь N — номер цвета раскраски (от 5 до 16), Ц — цвет отдельной точки (от 1 до 4).

нием, но и с радиотехникой, могут выбрать другой путь — усовершенствовать не Бейсик, а саму «Электронику БК». Доработка компьютера позволяет вместо четырех фиксированных цветов — черного, красного, синего и зеленого — выбирать одну из 63 различных палитр. Каждая палитра — это набор из четырех различных цветов, причем один из этих цветов обязательно черный, а три остальных — любые из семи цветов: белого, желтого, голубого, зеленого, фиолетового, красного и синего. Прежде чем взяться за паяльник, взвесьте свои силы — переделка достаточно сложная, кроме того, доработанный компьютер (и телевизор) уже не удастся сдать в ремонт. Тем, кто отважится на модернизацию своего БК, киевлянин В. Кондрашев предлагает опробованный им на практике способ.





Проще всего, «Электронику БК-0010» подготовить к работе с четырьмя основными цветами. Для этого на печатную плату устанавливают недостающие элементы — четыре транзистора КТ315, двенадцать резисторов и разъем XT11 — точно такой же, как XT9. Отверстия в плате под все элементы уже имеются, остается только правильно разместить детали и аккуратно припаять их выводы. Рисунок принципиальной схемы и фрагмент платы БК помогут избежать ошибки. Зеленым цветом на рисунке выделены новые элементы схемы. Чтобы получить цветное изображение на телевизоре, достаточно соединить кабелем разъем XT11 и входы видеоусилителей красного, синего и зеленого лучей.

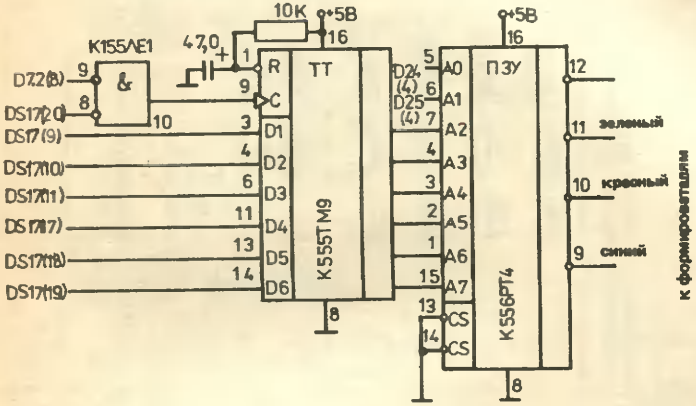
Переделка достаточно проста, но так можно получить лишь четыре основных цвета. Для того чтобы расширить цветовую гамму, я разработал другой способ. Если запрограммировать постоянное запоминающее устройство — ПЗУ так, как показано в табл. 1, то вместо трех цветов мы получим уже восемь. Цифра 0 в коде означает, что на усилитель подается сигнал, т. е. комбинация 001 соответствует тому, что сигнал подан на «зеленый» и «красный» входы — такое сочетание даст на экране желтый цвет. Таким образом, все адресное пространство ПЗУ необходимо

8-ричное число		0	1	2	3	4	5	6	7	Вход видеоусилителя
Код на выходе ПЗУ К556РТ4 Номер выхода	1	0	1	0	1	0	1	0	1	синий
	2	0	0	1	1	0	0	1	1	красный
	3	0	0	0	0	1	1	1	1	зеленый
	4	0	0	0	0	0	0	0	0	не нужен
Цвет на экране		белый	желтый	голубой	зеленый	фиолет.	красный	синий	черный	

Палитра	Адрес	Код	Цвет
1	0	5	красный
	1	3	зеленый
	2	6	синий
	3	7	черный
2	4	0	белый
	5	0	белый
	6	0	белый
	7	7	черный
3	8	1	желтый
	9	2	голубой
	10	4	фиолет
	11	7	черный

и т. д.

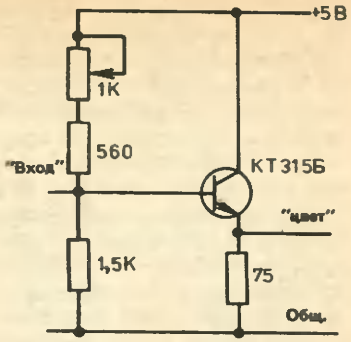
случайных срабатываний, на микросхему К155ЛЕ1 поступает сигнал с 15 разряда магистрали процессора адрес/данные. Входы регистра управления подключены к 9—14 разрядам магистрали. Поскольку в регистр управления информация поступает прямо из магистрали, то записывается она инвертированной, а значит, и код палитры необходимо записывать в инверсном виде, например: код 177000_8 установит нулевую палитру, код 176000_8 — первую, а 175000_8 — вторую палитру. Код требуемой палитры за-



разделить на «палитры» (табл. 2). Требуемый цвет в палитре кодируется соответствующей цифрой — для этого используют 001, но последним кодом в каждой палитре должна быть цифра 7, соответствующая черному цвету. Когда составление табл. 2 будет завершено, информацию переносят в ПЗУ, как говорят радиолюбители, ПЗУ прошивают. Описание процесса прошивки достаточно известно, его нетрудно найти в радиолюбительской литературе, поэтому подробности я опускаю. Поясню работу схемы в целом. Для выбора палитры используются старшие шесть разрядов адреса. Эти разряды записываются в регистр управления — микросхема К555ТМ9 (рис. 3). Чтобы защитить регистр от

писывают по адресу 177716_8. При включении питания автоматически происходит установка режима 0-палитры.

Несколько советов по практической части переделки. С основной печатной платы аккуратно удалите микросхему D20 К155ЛЕ1, обрежьте дорожки, ведущие к ее выводам. На освободившееся место установите панельку для ПЗУ К556РТ4. От D20 осталось только 14 отверстий, поэтому два «лишних» вывода панельки осторожно отогните в стороны. Микросхему К555ТМ9 установите на свободное место, а микросхему К155ЛЕ1 рядом с ней, например так, как я сделал в своем компьютере. Три транзистора, резисторы и потенциометры, необходимые для регулировки яркости, устанавливают на



свободные места. Соединения производите точно по схеме, обрезая все ненужные связи. На разъеме, к которому подключается телевизор, есть перемычка между контактами 3 и 5 — ее удаляют и, отсоединив контакты 1, 3 и 5 от схемы, подключают их к выходам транзисторных формирователей. Перемычку R55—R56, которая находилась на разъеме, установите на плате. Для синхронизации можно воспользоваться сигналом черно-белого изображения, т. е. к телевизору подключаем 1, 3, 5 и 4 контакты разъема XT9. Переделка достаточно сложная, но и результат не обманет ваших ожиданий!

Доска объявлений

Составлять программы управления внешними устройствами удобнее, если воспользоваться разработанным нами устройством индикации на светодиодах. Имеются схемы ключей для управления внешними устройствами на различной элементной базе (от реле до транзисторов). Интересуемся учебными программами и динамическими играми. 6133401, Кировская обл., Куменский район, пос. Речной, ул. Ленина, 5—7, Караваев А. И.

В следующем номере нашего журнала читайте:
в рубрике *«Зарубежный опыт»* статью **В. Тихонова** «О конкурсе американского журнала «Эдьюком»;
в рубрике *«ЭВМ в народном хозяйстве»* статью **Н. Исаевой, Н. Мироносецкого, Л. Парфеновой** «Компьютер и хозрасчет»;
в рубрике *«Внеклассная работа»* статью заслуженного учителя РСФСР **А. Караваева** «Электронный видеосигнал»,
решения задач III Московской олимпиады по информатике и другие материалы.

«Информатика на службе международного диалога» — девиз ***Международного компьютерного клуба,*** открывшегося в декабре 1988 г. в Москве.

К настоящему времени его учредителями стали, вложив крупные суммы (свыше 250 тыс. рублей или соответствующую сумму в валюте), более 20 советских и 4 иностранные организации и фирмы.

Клуб будет принимать заявки к вступлению на правах учредителей до конца 1990 г.

Коллективным членом клуба может стать, имея свежие технические идеи и замыслы, интересный проект и его экономическое обоснование и, конечно, заплатив вступительный взнос не менее 10 тыс. рублей (либо соответствующую сумму в валюте). Заявка прилагается. Ее нужно заполнить в случае вашей готовности вступить в клуб и прислать для предварительного согласования (сначала без перечисления денег). Не менее трех месяцев понадобятся клубу, чтобы определить, интересует ли клуб ваш проект. Деньги следует перечислять только после получения подтверждения из клуба о вашем приеме.

Для индивидуальных членов вступительный взнос составит от 250 рублей и выше (по вашему желанию). Однако отбор индивидуальных членов будет еще жестче. Так что если нет революционных в техническом отношении идей — повремените с отправкой заявки в клуб.

МКК ставит своей целью, используя колоссальный интеллектуальный потенциал нашей страны, вывести на международный рынок наши программные средства, предоставив взамен зарубежным партнерам наш практически необозримый рынок для продажи техники.

Наивно предполагать, что каждый вступивший в клуб будет незамедлительно обеспечен компьютерами, поэтому, если это ваша единственная цель, вам лучше не вступать в клуб.

Однако если ваш проект заинтересует клуб, он может предоставить автору необходимую технику для его реализации.

Клуб будет осуществлять свою деятельность через секции. Одной из первых

101813, Москва, Центр,
проезд Серова, 4,
Правление Всесоюзного
общества «Знание»

зарегистрировалась в МКК секция «Компьютер и познание», объединившая молодых, ищущих ученых и общественных деятелей, которые основными направлениями своей работы считают создание и апробацию нового содержания образования, его методическую и программную поддержку, анализ и распространение опыта использования компьютера в школе. Справки о работе секции у ее технического секретаря *Н. Булановой*. Тел. в Москве: **254-42-59.**

МКК при участии журнала *«ИНФОРМАТИКА И ОБРАЗОВАНИЕ»* объявляет конкурс на лучшую обучающую программу по русскому языку (для иностранцев) и программу по русскому языку для советской школы.

Установлены, соответственно, две первые премии по 2 тыс. рублей, четыре вторых — по 1 тыс. рублей и шесть третьих — по 500 рублей.

Запросы об условиях конкурса присылать по адресу редакции не позднее 30 мая 1989 г. На конверте сделать пометку **«Запрос о конкурсе».**

Генеральному директору
Международного
компьютерного клуба
тов. ЕГОРОВУ А. Ю.

ЗАЯВКА* на членство

Наша организация желает вступить в качестве члена/учредителя в Международный компьютерный клуб (МКК), признает и разделяет его устав.

В качестве вступительного взноса предлагаем перечислить на счет МКК (счет № 70130635 в советских рублях и счет № 18000550/002 в иностранной валюте во В/О «Союзинрасчет» Внешэкономбанка СССР, Москва, Серпуховской вал, 8) _____ в рублях/валюте.

Готовы принять активное участие в основных направлениях деятельности МКК, а также оказывать помощь в его деятельности (например, в виде участия в разработке проектов, реализации его программ, рекламировании его работы и т. д.).

Руководитель

Главный бухгалтер

* В заявке просьба также указать:

сведения о возможности предоставить помещения для фирменных выставок-классов МКК, семинаров и т. п.;

в работе какой секции МКК желали бы принять участие (автоматизации банковского дела, математического моделирования, автоматизированных обучающих систем, информатики в медицине, экологии, экономических и социальных проблем информатизации, компьютеризации шахмат, международных коммуникаций и др.);

какую программу, программный продукт, проект вы бы хотели продемонстрировать, разработать, продать, купить и т. д. через посредничество и с помощью МКК.

Индекс журнала «Информатика и образование» в каталогах «Союзпечати» — 70423.
Свободная подписка на наш журнал на 1990 г. открыта с марта 1989 г.

Точка зрения

А. МАТЮШКИН-ГЕРКЕ

О перспективах развития школьного курса информатики

В настоящее время мы подходим к очередному этапу школьной компьютеризации. Значительное расширение доступа учащихся к вычислительной технике, приобретение учителями информатики минимально необходимой подготовки, разнообразный опыт проведения 68—102-часового курса ОИВТ в IX—X классах — все это создает определенные предпосылки для более широкого и действенного использования компьютеров в учебно-воспитательном процессе. Характерными для нового этапа компьютеризации, по нашему мнению, будут, во-первых, смещение начала систематического изучения информатики в средние классы[1]; во-вторых, переход от случайного, эпизодического к регулярному использованию ВТ в других учебных дисциплинах.

Переход к этому новому этапу требует прежде всего заблаговременной разработки его концептуальной основы. При этом главными являются вопросы содержания и построения школьного курса информатики и вычислительной техники (ИВТ), о том, как должен он соотноситься с курсами смежных дисциплин.

Очевидно, что при определении содержания и структуры любого курса следует ориентироваться прежде всего на те цели, ради которых и вводилась соответствующая дисциплина в учебный план. Отправной точкой для проектирования курса ИВТ являются четко сформулированные партийные и государственные требования: «Привить учащимся навыки пользования компьютерами и вооружить их знаниями о широком применении этой техники в народном хозяйстве» (Правда. 1985. 2 февр.). Поэтому совершенно неправомерны весьма часто встречающиеся сейчас попытки подойти к построению этого курса, отталкиваясь от той или иной трактовки неустоявшегося пока еще термина «информатика».

Никакими обстоятельствами не может быть оправдано и сведение курса к изучению в основном лишь элементов алгоритмизации и программирования, хотя бы и «разбавленных» некоторыми возможностями распространенных ныне операционных систем. К сожалению, тенденции такого рода существуют, они нередко встречают официальную поддержку, порой — широко рекламируются.

Многолетний опыт применения ЭВТ в самых разных областях убедительно подтверждает, что определяющую роль здесь играют правильный выбор и грамотная постановка задач. Применение такого мощного средства, как ЭВТ, существенным образом расширяет человеческие возможности, но в то же время требует от прямо или косвенно (!) использующих ее людей нового, качественно более высокого уровня научно-технического мышления. Ведь *включение ЭВТ в любую деятельность оказывается, как правило, успешным лишь при коренном изменении как содержания (новые цели и задачи), так и характера (новые способы) этой деятельности!*

Насколько непросто все это дается, можно судить по тем значительным упущениям и потерям, которыми уже много лет сопровождаются создание и эксплуатация различного рода автоматизированных систем, применение ЭВМ для решения научных и инженерных проблем. Главная причина этих неудач — неумение или нежелание понять необходимость создания и использования принципиально новых систем моделей тех объектов, процессов и явлений, чье изучение и управление которыми имелось в виду проводить с применением ВТ. В равной мере неприемлемыми оказывались как система управления предприятием, в которой современные ЭВМ пытались «состыковать» с рутинными структурой и функция-

[1] Здесь не имеется в виду первоначальное знакомство с компьютером, которое вполне может начинаться гораздо раньше.

ми административных служб, так и «компьютеризация» расчетов по алгоритмам, ориентированным на «ручной» (или «слабомеханизированный») счет.

Узкий «информационно-кибернетический» подход, примитивное вычленение одной лишь «чисто алгоритмической компоненты» в таких видах деятельности, где решающую роль играют социальные, психологические, экономические, правовые и организационные аспекты,— все это не только нигде и никогда не приводило к успеху, но, напротив, лишь дискредитировало вычислительную технику в глазах многих специалистов и руководителей.

Можно сказать, что главная помеха на пути ее эффективного использования — это методологическая некомпетентность. Великолепное владение техникой, блестящее знание операционных и прикладных систем — ничто без умения правильно выбрать точку приложения сил при решении практических проблем. Поэтому ясно, насколько остро мы нуждаемся в совершенствовании именно общенаучной, общетеоретической подготовки специалистов. Основы же такой подготовки могут и должны закладываться еще в средней школе.

Авторская точка зрения на сегодняшние требования к курсу ИВТ определена в [1]. В укрупненном виде его структура представляется следующей:

Основы алгоритмизации и программирования.

Структура и функционирование средств вычислительной техники.

Характеристика современного программного обеспечения, средств и методов его создания.

Основные и перспективные направления использования вычислительной техники в народном хозяйстве. Компьютерная культура современного специалиста.

Эта модель курса в различных модификациях была успешно реализована [2]. В то же время нельзя признать удовлетворительной привязку курса к 9-му и 10-му годам обучения. Столь позднее начало приводит к тому, что он не успевает получить достаточного выхода в другие дисциплины.

Могло бы показаться, что эта проблема решается простым «параллельным переносом» сроков его изучения. Однако такая, чисто механическая, передвижка курса в целом (!) из старших классов в средние классы невозможна, ибо тогда заключительные его разделы учащиеся попросту не смогут понять. А ведь именно эти разделы являются наиболее важными для формирования научного мировоззрения учащихся, и при любой перестройке курса соответствующую тематику и надлежит как раз усиливать.

Понимание объективных закономерностей, связанных с использованием средств ВТ, способность, образно говоря, видеть свою предметную область через призму вновь открывающихся возможностей, готовность к нестандартным подходам, умение взаимодействовать со специалистами различного профиля — вот ведущие компоненты компьютерной культуры, начальное воспитание которой следует считать *основной задачей* компьютеризации школьного образования.

Разумеется, эта задача должна решаться не только одним курсом информатики, роль других дисциплин тут достаточно велика и со временем будет лишь расти. Однако заключительный, обобщающий раздел («**Вычислительная техника в народном хозяйстве. Компьютерная культура современного специалиста**») следует изучать именно в рамках курса информатики и изучать тогда, когда учащиеся будут в достаточной степени подготовлены к восприятию этого материала. По сравнению с представленной в [1] моделью курса материал тем 27 и 28 должен быть существенно расширен и дополнен. Нетрудно понять, что изучение этого заключительного раздела целесообразно отнести как можно ближе к концу срока обучения в средней школе.

С другой стороны, изучение основ алгоритмизации и программирования, освоение простейших правил и приемов работы с готовым программным обеспечением, в том числе и с учебными диалоговыми системами, являются необходимыми предпосылками для успешного использования ВТ в других предметах, во внеклассной работе и т. д. Следовательно, соответствующие разделы курса информатики желательно изучать в более младших классах.

Не вызывает сомнений и необходимость включения в курс раздела, посвященного изучению структуры и функционирования средств ВТ. При определении места этого раздела в учебном плане необходимо учитывать его существенную зависимость от динамики изучения курсов математики и физики. Здесь весьма желательна возможность опереться на получаемые в этих курсах знания учащихся по таким разделам, как системы счисления, различные формы представления чисел, электрические цепи, основы электроники.

Отрадно, что в последнее время в некоторые варианты курса математики, например, [3] стали включаться вопросы, связанные с использованием программируемых калькуляторов. В настоящее время этот раздел относится к восьмому году обучения, но, видимо, более раннее его изучение не встре-

тит каких-либо принципиальных препятствий. Заметим, что еще длительное время ПМК будут представлять собой единственный из видов ВТ, доступный для широкого использования в этих курсах. Несмотря на кажущийся «примитивизм» этой техники, именно в изучении математики и физики с ее помощью можно достичь очень многого. Кроме того, и это убедительно подтверждается опытом, использование ПМК самым существенным образом облегчает учащимся последующее изучение ряда разделов и тем курса ИВТ.

Отсутствие в настоящий момент общепризнанной концепции нового учебного плана общеобразовательной средней школы не дает возможностей для сколько-нибудь далеко идущей детализации выдвигаемых здесь предложений. Поэтому для раскрытия их сущности нам придется с большой долей условности «спроецировать» эти предложения на ныне действующий учебный план, рассчитанный на 10-летний срок обучения. Применительно к этому плану курс информатики мог бы включать в себя порядка 100 ч, отнесенных к 7-му и 8-му годам обучения (со второй половины VII класса по 2 ч в неделю) и около 35 ч — к 10-му году обучения. Изучение же небольшого вводного раздела, подготавливающее учащихся к простейшим способам использования компьютеров, вполне можно начинать с I класса. При правильной организации учебного процесса в целом такое размещение курса ИВТ отнюдь не будет означать каких-либо «нарушений непрерывности» компьютерной компоненты образования. Образно выражаясь, курс информатики, подобно подземной реке, временно будет уходить в другие дисциплины с тем, чтобы впоследствии уже в новом качестве вновь выйти на поверхность. Многое из того, что предусматривается темами 9, 23 и 24 описанной в [1] модели, гораздо уместнее делать в курсах смежных с информатикой дисциплин.

Один из возможных вариантов такого построения учебного процесса можно проиллюстрировать таблицей.

Намеченный подход предполагает обязательное выполнение ряда условий. К их числу прежде всего следует отнести:

достижение необходимого уровня технического и программно-информационного обеспечения как курса ИВТ, так и курсов смежных дисциплин;

глубокую организационно-методическую проработку всех вопросов согласования и стыковки изучаемых курсов;

фундаментальную переподготовку учителей (прежде всего информатики, математики, физики).

Условия, как можно видеть, далеко не простые, каждое из них требует глубо-

Распределение учебного материала между информатикой, математикой и физикой

Год обучения (полугодие)	Предметы		
	Математика	Физика	Информатика
VI—VII (1) и ранее	Системы счисления. Различные формы представления чисел. Составление и использование простейших программ	Электрические цепи. Основы электроники. Применение ПМК для вычисления по формулам	
VII (2) — VIII	Решение несложных вычислительных задач: нахождение значений функций, вычисление сумм и произведений, решение уравнений и их систем	Решение задач, связанных с действием непостоянных сил, и им подобных	Основы алгоритмизации и программирования. Структура и функционирование средств вычислительной техники. Особенности ее использования при решении прикладных задач
IX	Решение вычислительных задач, обработка результатов наблюдений, машинный (вычислительный) эксперимент	Решение физических задач с использованием различных типов математических моделей	
X	То же, что в IX	То же, что в IX	Характеристика современного программного обеспечения и средств его создания. Вычислительная техника в народном хозяйстве. Компьютерная культура современного специалиста

кого, обстоятельного рассмотрения. Здесь же мы ограничимся лишь краткой характеристикой каждого из них.

По-видимому, на первых порах наиболее характерным техническим обеспечением учебного процесса будет:

ПМК — для систематического использования на уроках математики и физики;

простейшие ПЭВМ — для систематического использования на уроках информатики и эпизодического — на уроках других дисциплин;

профессиональные ЭВМ (как правило, в межшкольных центрах) — для эпизодического использования при изучении заключительных разделов курса информатики.

Состав программно-информационного обеспечения должен давать возможность как самостоятельной разработки учащимися несложных программ на тех или иных алгоритмических языках, так и использования ими продвинутых программных средств, диалоговых систем, в том числе построенных с использованием принципов искусственного интеллекта. Для смежных с информатикой дисциплин нужны будут как библиотеки простейших учебных программ, так и специально ориентированное на конкретные учебные приложения «мягкое» программное обеспечение.

Наивны и обречены на неуспех любые попытки «приспособить», «приложить» компьютеры к существующим в своем сегодняшнем виде курсам. *В условиях систематического использования средств ВТ не только методика преподавания, но и содержание (!) ряда дисциплин должны будут самым существенным образом измениться.* Мы уже упоминали методологический принцип, лежащий в основе этого утверждения: «Новые средства→расширение возможностей→новые цели и задачи→новые подходы к их решению». Первоначально нам потребуются совершенно новые (!) курсы математики и физики, а затем и ряда других дисциплин. Как уже отмечалось, гораздо более значимой станет роль межпредметных связей.

Без такой кардинальной перестройки учебного процесса его «компьютерная компонента» окажется в нем бесполезным, если не вредным, чужеродным телом. Подготовка к этой перестройке должна активно и широко проводиться уже в настоящее время. Особенно необходимо проведение глубоких педагогических экспериментов, их широкая гласность, тщательный и всесторонний анализ. Следует создать для таких исследований «режим особого благоприятствования», обеспечить действенное привлечение к ним нужных специалистов, независимо от их «ведомственной принадлежности».

Совершенно ясно, что невозможно будет строить по-новому учебный процесс, не опираясь при этом на соответствующим образом подготовленные учительские кадры. На этом этапе им потребуется гораздо более обстоятельная подготовка, чем та, которую можно обеспечить в стандартных 36—72-часовых рамках. Минимально необходимый объем только аудиторных занятий здесь оказывается не менее 800—1000 ч[2]. Осуществить такую подготовку без продолжительного полного или частичного освобождения учителей от их основных служебных обязанностей вряд ли окажется возможным. Необходимо как можно скорее реально уравнять учителей с другими категориями специалистов в правах на переподготовку и повышение своей квалификации. (До года — с полным отрывом от работы или до двух лет — с отрывом на 50 %.)

Коренной перестройки потребует и система подготовки будущих учителей в педагогических институтах. Готовность большинства институтов к такой перестройке в настоящий момент вызывает большие сомнения. Значит, и самим этим институтам тоже надо «садиться за парту»!

Более чем трехлетний опыт «внедрения» курса основ информатики в среднюю школу принес с собой немало поучительного. Сейчас уже должно быть совершенно ясно, что «облегченный» («облегченный» как в интеллектуальном, так и в материальном плане) подход к решению столь сложных проблем ничего хорошего не сулит. Поэтому в заключение хотелось бы выразить пожелание и надежду, что недавно созданный Госкомитет по народному образованию сумеет, не в пример своим предшественникам, правильно организовать всю деятельность по компьютеризации учебно-воспитательного процесса, оптимально использовать имеющийся для этого в стране потенциал.

Литература

1. *Матюшкин-Герке А. А.* Каким быть школьному курсу информатики // Информатика и образование. 1987. № 6.

2. *Зив Б. Г., Матюшкин-Герке А. А.* Из опыта преподавания курса основ информатики и вычислительной техники // Опыт и проблемы внедрения компьютерной техники в процесс обучения. Л., 1986. С. 31—34.

3. *Алимов Ш. А.* и др. Алгебра. 8 класс. М.: Просвещение, 1987.

[2] Эта оценка обусловлена многолетним опытом работы ЛИМТУ, в том числе продолжающейся с 1985 г. подготовкой учителей.

СВД—ВУЗ. Система виртуального доступа для высших учебных заведений

Предназначена для оборудования дисплейных классов, учебных кабинетов, залов курсового и дипломного проектирования, лабораторий высших учебных заведений и других организаций; для выполнения достаточно сложных учебных, вычислительных, научно-исследовательских и других видов работ, когда используется принцип коллективного и равноправного пользования единым внешним оборудованием общего назначения.

Конструктивно система состоит из главной ЭВМ ДВК-4 (3) «Электроника МС 0507.04», звездообразно соединенной с шестью учебными терминалами (микро-ЭВМ УКНЦ «Электроника МС 0511») через шестиканальный интерфейс радиальной последовательной связи (ИРПС-6) с помощью витых пар проводов. В качестве внешнего оборудования главной ЭВМ, которое является общим для всех шести пользователей, используется накопитель на жестких магнитных дисках «Электроника МС 5401» типа «Винчестер», накопитель на гибких магнитных дисках «Электроника МС 5305» и печатающее устройство типа «Роботрон СМ 6329.02М» (ГДР).

Система обеспечивает одновременную (параллельную) загрузку с жестких дисков главной ЭВМ в микро-ЭВМ учебных терминалов полной операционной системы RT-11 с сохранением всех ее функций. Отметим, что драйвер канала связи эмулирует устройства прямого доступа (диски), обеспечивая полную прозрачность доступа к удаленным устройствам внешней памяти. При этом создается возможность интенсивной работы с файлами равноправно всеми пользователями, свободно обеспечивается запуск оверлейных программ, т. е. снимаются ограничения на размер оперативной памяти.

Возможность использования существующего математического обеспечения операционной системы RT-11, широкий спектр языков программирования (Фортран IV, Паскаль, Бейсик) без каких-либо ограничений функций позволяют потребителю создавать свои эффективные прикладные программы и добиваться совместимости с семейством ЭВМ «Электроника-60», ДВК, СМ-3 и СМ-4.

Система разработана совместно МИТХТ и НПО «Научный центр».

На Международной весенней Лейпцигской выставке-ярмарке этого года система завоевала Золотую медаль.

Авторы В. И. Авдеев, В. Г. Беляев, А. Ю. Жедь, В. Ф. Корнюшко, А. А. Попов, Н. А. Смирнов, Г. М. Фролов. Ориентировочная стоимость комплекта на 6 учебных терминалов 100 тыс. рублей.

Адрес: 119831, Москва, Г-435, М. Пироговская, 1, МИТХТ, кафедра вычислительной математики.

Справки по московскому телефону: 246-46-50.

Во Франции нас читают

В ноябре в редакцию пришло письмо: французская фирма «Эрэстия» приглашала посетить свою экспозицию на международной выставке «Наука-88». Приглашение, прямо скажем, неожиданное, особенно если учесть, что предмет выставки — приборы для научных исследований — не вписывается в тематику нашего журнала. Конечно, в научных приборах используются ЭВМ, но стоит ли еще раз писать об этом? Тем не менее приглашение заинтересовало, и я отправился к французам.

Однако, попав в выставочные залы, я забыл и о 40 минутах ожидания автобуса у метро «Краснопресненская», и о нелегком штурме подошедшей машины, и даже о самой «Эрэстии».

Компьютеры! Они были везде. Большие и маленькие, автономные и встроенные, дорогие и дешевые. Только на второй день мне попались несколько стендов, на которых ЭВМ не присутствовали ни в каком виде. В целом выставку можно было бы назвать «Компьютеризация науки-88».

В зале № 1 прямо у входа — стенд «Интерквадро», совместного советско-франко-итальянского предприятия, рекламу учебного центра которого вы могли видеть в нашем журнале. На выставке «Интерквадро» предлагает программное обеспечение для научных экспериментов, решения статистических, физических и химических задач, пакеты прикладных программ...

Чуть дальше — фирма «Диалог», тоже совместная — советско-американская. Стенд оформлен в виде русской избы с компьютерами и почему-то с пальмой. Дизайн производит впечатление, но и рождает печальные ассоциации (изба...). На стенах — отпечатанные по-русски рекламные плакаты фирмы «Микрософт»: один из крупнейших поставщиков программного обеспечения вышел на советский рынок. «Мышь фирмы Microsoft совсем ручная!» Хотелось бы подержать ее в руках, но...

Но к делу. «Диалог» — предприятие по сборке ПЭВМ и разработке программного обеспечения. Оно также продает в СССР продукцию фирмы «Микрософт», совместно с ней готовит русские версии программ. Продает компьютеры, совместимые с IBM PC XT/AT, гибкие диски, красящую ленту (пока, увы, только за валюту). Организует обучение работе на IBM PC — 700 рублей для электронщиков, 1200 — для программистов. Не дешево! Впрочем, тот, кто может позволить себе купить IBM PC, безболезненно оплатит и учебу.

Несколько вопросов стендисту.

— Что «Диалог» предлагает для науки?

— Как ПЭВМ, так и разнообразные программы к ним — системные и прикладные, западные разработки и советские.

— Как идут дела с продажей советских программ на Западе?

— На расширение. Сейчас, например, рассматривается возможность заключения контрактов на поставку экспертных систем.

А пока «Диалог» предлагает советским покупателям оболочку экспертной системы «Шедл-1». Она работает в операционных системах MS—DOS, RT-11, VAX, VMS и является одновременно и «сырьем», и инструментом для самостоятельного создания экспертных систем. Как говорится: *возьмите одну «Шедл-1», вложите свои знания, организуйте сценарий диалога, добавьте сервиса по вкусу и запекайте в духовке до готовности.*

Обратим внимание на стенд кооператива «Сотрудничество». Приятно видеть на такой представительной выставке кооператив, тем более что он специализируется на разработке программных средств для обучения. Сейчас кооператоры работают на IBM PC AT и машинах с ОС UNIX, однако утверждают, что адаптация на другие ПЭВМ трудностей для них не составляет. В планах — «одеть» «Корвет».

Выставочный зал похож на город с четкой прямоугольной планировкой. Стенды — одно-

этажные домишки, потолок высок, как небо, узкие «тротуары» заполнены людьми... Слева стенд с микроскопами. ЭВМ не видно. Наконец-то! Приглядевшись, «разочаровываюсь»: и здесь компьютерные приставки! А людской поток увлекает дальше.

Что говорить о микроскопах — простая (когда-то) бюретка для дозирования жидкости становится электронной и цифровой.

Фирма «Хьюлетт-Паккард» демонстрирует медицинское оборудование. Дефибриллятор-монитор: естественно, с микропроцессором. Ультразвуковой диагностический комплекс: встроенная ЭВМ — основная часть прибора. Электрокардиограф: встроенный микропроцессор. Монитор-«сиделка», следящий за состоянием пациента: ЭВМ, к которой можно подключить до шести датчиков (из набора в 16 штук), каждый следит за своим параметром (температура, пульс, давление и т. д.). Данные записываются на магнитный диск, которого хватает на 48 ч. При необходимости «сиделка» поднимает тревогу*.

«Периферийное оборудование» — еще одно подходящее для этой выставки название. Когда компьютеров становится достаточно много (надеюсь, мы к этому все же придем), начинаются активные разработка и производство разнообразной периферии. И если в электрокардиограф микропроцессор встроен просто для удобства, а в дефибрилляторе-мониторе медицинская и компьютерная части примерно равноправны, то во множестве других случаев (монитор-«сиделка» — только один из примеров) ЭВМ играет явно лидирующую роль. Так что иногда следует говорить не «микроскоп с компьютером», а «компьютер с микроскопом».

А вот периферийные устройства в привычном смысле слова. Японская фирма «Джуки» демонстрирует цветные принтеры. Достаточно установить дополнительную графическую плату — и принтер «7200» превращается в цветной быстродействующий плоттер (графопостроитель). Гномик вверху слева напечатан на нем. С обслуживанием у покупателей проблем нет: при Институте экономики и прогнозирования научно-технического прогресса АН СССР создан техн. центр фирмы.

Снова компьютеры — карманные. Фирма «Касио» утверждает, что и микрокалькулятору есть место в науке. Спорить трудно — ведь у «калькулятора» полная алфавитно-цифровая клавиатура, жидкокристаллический дисплей на две строки по 32 знака, встроенный Бейсик, 8К байт ОЗУ, возможность подключения портативной внешней памяти объемом 32К байт, 116 процедур для научных расчетов...

А здесь ксероксы. Или алкатели? В общем, ксерокопировальные аппараты «Алкатель» фирмы «Уэстрейд», предлагаемые англо-пакистанской компанией «Текнолоджи энд Трейд». Она специализируется на бартерных сделках, иначе говоря, производит обмен «товар — товар». Уплатить за ксероксы (или компьютеры — они тоже есть в продаже) можно, например, металлоломом (это не шутка). Пожалуй, такая валюта доступна и для школьников!

Польская экспозиция, пятая по площади, кажется особенно крупной — ее стенды

сгруппированы в одном секторе, а не разбросаны по залам. Разнообразной периферии и компьютеров здесь множество — от простейших восьмиразрядных (но все же посложнее, чем «Микроша», и с бо́льшим ОЗУ, чем у БК-0010) до АТ-совместимых. Меня заинтересовали два КУВТа. Тот, что попроще, с черно-белыми мониторами, продавался за 32 500 СЭВовских рублей. Для сравнения: КУВТ-86 в магазине «Электроника» на Ленинском проспекте стоит 34 600 рублей, правда, советских.

Стенд французской фирмы «Аниваль Ютек» (кстати, одного из учредителей «Интерквадро») скромен по размеру, но чрезвычайно привлекателен по дизайну. Маленькие компьютеры бежево-шоколадных то-

* Не подумайте, что для работы со всем этим нужно быть программистом. Компьютеры настроены к пользователю дружественно, и медсестра свободно может управлять сложной техникой.

нов, телефон на клавиатуре... Эти «игрушки» — часть информационной системы МИНИТЕЛЬ-10.

Большая ЭВМ в центре снимка — часть другой системы, состоящей из двухпроцессорной центральной машины (ОЗУ — 1—4М байт, винчестер, гибкие диски и принтер) и трех периферийных (ОЗУ — 640К байт, винчестер, гибкий диск). Это проблемно-ориентированный комплекс по цене, как сказано в рекламе, «от 60 000 рублей, из них только 15 000 рублей в конвертируемой валюте». Для советского рынка очень дешево.

Когда, проголодавшись, я направился к единственному замеченному мной пункту питания — кооперативному кафе, обнаружилась любопытная организационная особенность выставочного комплекса. Кафе находится прямо напротив выхода из цокольного этажа здания. Выйти не представляло труда, а вот вернуться — невозможно. Обращенная на улицу табличка гласит: «Вход по спецпропускам». Чтобы оказаться с другой стороны «полупроводниковой» двери, пришлось преодолеть немалый и замысловатый путь, в основном по улице. Мне показалось, что «фирмачи» (вышедшие перекусить, не одеваясь) так и не поняли смысла сего загадочного действа.

— biocor —

С новыми силами — за осмотр. Во втором зале видное место занимает экспозиция «Биокора». Его «горячее сердце» привлекает внимание издалека. Основная цель этого кооператива — биологическая коррекция организма человека. Председатель — А. В. Кузин, известность которому принес новый метод лечения диабета. Несмотря на появившиеся впоследствии сообщения разочарованных пациентов о несостоятельности метода и даже о приостановке деятельности кооператива, «Биокор» приобрел патрона, хотя и очень молодого,— фирму «Биокор Интернэшнл», зарегистрированную в Люксембурге в августе 1988 г. и за три месяца успевшую обрасти филиалами в 35 странах (так утверждает стендист).

Возвращаясь к «электронной» теме, могу отметить, что компьютеры «Биокор» выпускает и продает хорошие. Быстрый Z-248/12, пятикилограммовый переносной Supers Port, Z-159 — вариант старого доброго PC XT и, наконец, Eazy-PC, самый простой, но и самый дешевый. Кстати, о ценах — они могут быть довольно низкими, но только «для своих»: здесь продают ПЭВМ на рубли тем, кто заинтересует кооператив своей программой применения ВТ, согласующейся с основными целями «Биокора».

Между фирмами «Текнолоджи энд Трейд» и «Файрекс АГ» — внешне неброские стенды Киевского механического завода и инженерного кооператива «Мысль». Киевский мехзавод предлагает микропроцессорную систему «КАМАК АН Мультитест» для испытания натурных конструкций на прочность, «Мысль» принимает заказы на проектирование, изготовление, установку и обслуживание оптоволоконных систем связи, а также на разработку программного обеспечения для информационных систем. Информатика все активнее входит в нашу жизнь!

ТЕКСТО

И вот, наконец, я добрался до стенда крупнейшей в Европе французской компьютерной фирмы «Бюль», где разместилась и фирма «Эрэстия». Меня встретили две милые француженки, говорящие по-русски,— директор фирмы Эммануэль Винь и переводчица Жислен Летурно-Цыганоф*. Выяснилось, что треть работников фирмы (иначе говоря, один человек — Эммануэль) выписывает, более того,

* Это они приветствуют наших читателей с четвертой страницы обложки.

читает ИНФО и даже находит там нечто для себя интересное. Не скрою, слышать это было чрезвычайно приятно.

«Эрэстия», состоящая лишь из трех человек (Кооператоры! Берите не числом, а умением!), создана специально для торговли в странах Восточной Европы программными продуктами «Тексто» и «Логотель» (о них чуть позже), разработанными другой небольшой фирмой — «Кемдата» (существующей с 1975 г. и насчитывавшей в 1987 г. 21 работника). Судя по тому, что Эммануэль и Жислен приехали в Москву уже седьмой раз, хотя фирма существует всего год, дела идут неплохо. Ну а если судить по списку западных покупателей (в нем среди крупнейших фирм и организаций — «Форд», «Дженерал моторс», «Эссо», «Рено», Британская библиотека, Комиссия европейских сообществ, Канцелярия президента Франции; среди образовательных — Министерство народного просвещения Франции, 15 университетов Европы, Азии и Африки), «Тексто» пользуется большой популярностью.

В чем же секрет успеха? «Тексто» — нечто вроде базы данных, ориентированной на управление документооборотом любой организации, не обязательно бюрократической. Более того, чем дальше организация от бюрократизма, тем более осмысленны ее документы и заключенная в них информация, тем важнее не просто их хранение и поиск (в бумажных ли архивах, в памяти ли ЭВМ), а разнообразное и активное использование, возможность переформатировать, менять структуры, добавлять данные и «передвигать» их из документа в документ. Все эти возможности у «Тексто» реализованы лучше, чем у обычной базы данных.

Созданная в конце 70-х гг. научным директором «Кемдаты» Н. Винем, «Тексто» постепенно завоевывала рынок. Со временем она была адаптирована почти ко всем операционным системам — от MS-DOS до VMS и UNIX, множеству различных ЭВМ — от персоналок до больших машин, диалог стало возможным вести на восьми языках (в том числе на арабском). Теперь бизнесмен, использующий «Тексто», в любой стране может найти партнера, документацию у которого ведет она же. Естественно, все это еще более повысило популярность программы. К тому же для «вдумчивых пользователей» был создан «Логотель» — язык, позволяющий из кубиков «Тексто» строить новые конструкции, максимально учитывая запросы конкретного потребителя. В итоге «Тексто» стала лидером рынка.

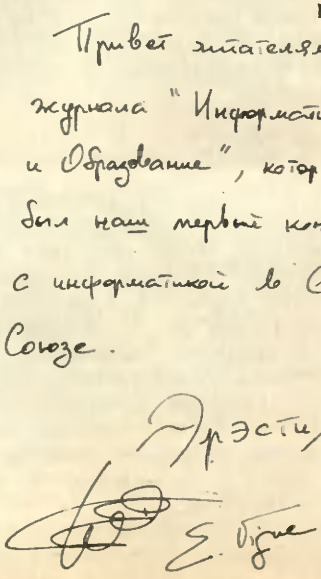
Какие отсюда можно сделать выводы? Во-первых, не боги горшки обжигают. Каждую программу, даже самую знаменитую, пишут не могущественные корпорации, а обычные люди (ну, может быть, талантливые), порой даже в одиночку. На мой взгляд, только отсутствие информации о западном рынке и западной жизни, их особенностях и потребностях мешает советским программистам показать себя миру во всей красе.

Во-вторых, торговля программами требует (повторюсь) не числа, а умения, не больших ассигнований, а инициативы, и если будут решены вопросы авторского права и защиты программных продуктов (к этой теме наш журнал обязательно вернется), то для кооперативов откроется очень интересное поле деятельности, благо последние постановления Совмина СССР значительно упрощают внешнеторговые операции.

◆

В заключение напомню, что перед вами не обзор выставки, а, так сказать, путевые заметки. Упомянута лишь малая часть экспонатов, в частности, вне поля зрения остались крупные экспозиции СССР и ФРГ. Внимание сосредоточено на ЭВМ не только потому, что наш журнал — «Информатика и образование», но и потому, что компьютер — характернейшая черта выставки. Почти не освещена торговля программным обеспечением, а она была очень активной! И все же думаю, что мои впечатления достаточно верно отражают и атмосферу выставки, и тенденции нашего времени. Уверен, что не прогадает тот, кто изберет своей профессией одну из принадлежащих к широчайшей сфере информатики.

К. ШЕХОВЦЕВ



Нам пишут

О преимуществах алгоритмической нотации

В апреле 1986 г. на совещании преподавателей информатики вузов СССР среди присутствовавших преобладало мнение о ненужности алгоритмической нотации (АН). Ее называли мертвым языком, имея в виду невозможность применения для решения задач на ЭВМ. Выступившему на совещании академику А. П. Ершову была передана записка с предложением с помощью голосования убедиться в преобладании среди преподавателей мнения о необходимости полной замены АН в школьной программе по информатике Бейсиком.

Впоследствии авторам этой статьи пришлось преподавать АН студентам нематематических специальностей «Физика и астрономия» и «Общетехнические дисциплины и профориентация». Необходимость отойти от привычной математизированной формы изложения выявила преимущества, которыми обладает АН по сравнению с языками программирования.

Алгоритмы невычислительного характера.

Начнем с того, что школьные учебники информатики неправильно очерчивают сферу применения АН. В начале учебника для IX класса приводится несколько алгоритмов невычислительного характера, а затем идут лишь алгоритмы работы с величинами. В результате складывается впечатление, что АН предназначена для вычислений; однако она не является языком программирования — так что же можно с ее помощью вычислять? Отсюда и следует кажущийся естественным вывод о ее ненужности. Однако этот вывод ошибочен.

Дело в том, что АН можно применять в гораздо более широкой области, чем вычисления, и именно благодаря ее немашинности. Какие операции можно выполнить с помощью Бейсик-программы на персональной ЭВМ? Действия над числовыми и строковыми величинами, печать текста или рисунков, воспроизведение звука, на некоторых ЭВМ — что-то еще, но всегда набор операций сильно ограничен. Алгоритмическая же нотация разрешает запись операций любого вида, независимо от того, будут ли эти операции выполнять ЭВМ, человек, несколько предприятий или живая клетка. Записанный алгоритм отвечает на вопрос, что и в каком порядке делается или следует делать, а такой вопрос может возникнуть в любой области жизни.

Рассмотрим алгоритм судейства игры в настольный теннис.

```
алг судейство игры в настольный теннис
нач объявить счет 0:0
    жеребьевкой определить подающего игрока
    пока один из игроков не набрал 21 очко
    нц
        разыграть подачу
        добавить очко выигравшему подачу
        объявить счет
        если сумма очков делится на 5
            то смена подающего
        все
    кц
    если у другого игрока меньше 20 очков
        то объявить победителем игрока, набравшего 21 очко
        иначе пока разница очков не равна двум
            нц
            разыграть подачу
            добавить очко выигравшему подачу
            объявить счет
            смена подающего
```

```
кц
    объявить победителем того игрока,
    у которого на 2 очка больше
все
кон
```

Как видите, судейство игры в настольный теннис наглядно и кратко может быть представлено на АН в виде алгоритма. Так и должно быть, иначе при проведении игр судейство вызывало бы многочисленные споры. Но не думайте, что так просто записать судейство любой спортивной игры. Попробуйте, например, самостоятельно записать алгоритм судейства игры в волейбол. Здесь трудность возникает вследствие того, что добавление очка или смена подающей команды зависит от того, какая команда выиграла мяч: подававшая или ее соперница. Тем более трудной является задача записи алгоритма судейства игры в большой теннис. В этой игре приходится вести счет в каждом гейме, счет самих геймов и счет сетов.

Все же правила проведения спортивных соревнований достаточно точно описаны, поэтому их запись оказывается много проще, чем некоторых алгоритмов из обыденной жизни. Но прежде чем рассматривать сложные алгоритмы, выделим достаточно простую группу, которую можно назвать алгоритмами классификации. Типичным примером является определитель растений. С помощью вопросов о признаках растения (форма листьев, порядок следования листьев на стебле, число лепестков в цветке и т. д.) мы переходим от одного предположения к другому, пока не выясним точного названия растения. К этой же группе относятся определение минералов, диагностика заболеваний, поиск неисправностей. Алгоритмическое описание в этих областях постепенно применяется все шире и приносит немалую выгоду.

Алгоритмический принцип классификации был бы весьма полезен и в толковых словарях. Пока же, например, с помощью «Словаря иностранных слов» (М.: Русский язык, 1984) невозможно отличить джемпер от пуловера, хотя они описаны и не совсем одинаково.

Вот еще один алгоритм классификационного типа. Им подсознательно пользуются владеющие русским языком (к сожалению, не все). Мы говорим: «двадцать один метр», «тридцать два метра», «одиннадцать метров». Падеж и число существительного (в данном случае слова «метр») зависят от числительного. Мы легко ставим существительное в нужном падеже, однако предлагаемое в качестве упражнения задание записать соответствующий алгоритм неизменно вызывает у студентов затруднения. На первый взгляд кажется, что на падеж влияет последняя цифра числительного, однако 11 и 21 имеют одинаковую последнюю цифру, но требуют разного падежа. Дело в том, что роль играет не последняя цифра, а последнее слово при произнесении числительного, ведь запись чисел появилась уже после того, как люди научились говорить.

Алгоритм имеет следующий вид.

```
алг согласование существительного с числительным
нач
  выбор
    при последнее слово числительного=
       =«один»:
       единственное число, именительный падеж
    при последнее слово числительного=
       =«два» или «три» или «четыре»:
       единственное число, родительный падеж
    иначе множественное число, родительный падеж
  все
кон
```


В этом алгоритме даже хочется выделить аргумент и результат — жаль, в АН нет возможности определить тип величины «падеж».

Существует еще много интересных алгоритмов, взятых из жизни, например алгоритм приема в вуз, алгоритм определения наказания за преступление в соответствии с уголовным кодексом, алгоритмы вязания.

АН и программирование. У читателей могло сложиться впечатление, что мы вообще против применения АН для целей программирования. Вовсе нет. Она исключительно полезна при программировании сложных задач. Неопытные программисты совершают типичную ошибку — бросают писать программу, еще толком не разобравшись в постановке задачи, не составив плана программы. Если же мы попытаемся отвлечь их от написания программы, дав задание сначала составить ее на АН, то их все равно остановят вычислительные трудности. Где же выход? Он заключается в простом правиле: сначала программа составляется на АН, но запрещается явно использовать математические обозначения. Например, вместо + надо писать *прибавить*, вместо $\sin\alpha$ — вычислить синус угла поворота зеркала и т. д. Эффект от такого запрета совершенно неожиданный. Дело в том, что при таком условии даже простое выражение очень трудно записать, например $S=\frac{at^2}{2}$ превратится в фразу *путь положить равным ускорению, умноженному на время движения в квадрате и деленному на два*. Гораздо проще написать

вычислить путь. Но именно этого мы и добиваемся: чтобы программист, не вдаваясь в подробности, в краткой фразе выразил суть выполняемой операции.

Продемонстрируем плодотворность такого подхода на примере. Пусть требуется получить разложение натурального числа n на простые множители, т. е. представить в виде $p_1^{a_1} \cdot p_2^{a_2} \cdot \ldots \cdot p_k^{a_k}$, где p_i — различные простые числа, а a_i — натуральные показатели степени.

Сначала запишем идею алгоритма в словесной форме: будем пытаться делить n на последовательные числа, начиная с двойки, до тех пор пока не получим единицу. При каждом «удачном» делении будем подсчитывать, сколько раз n поделилось на данный делитель.

Теперь несколько формализуем алгоритм.

Делителю присвоить значение два

Номеру простого множителя присвоить значение ноль

пока n больше 1
нц
  если n делится на делитель
    то номер простого множителя увеличить на 1, запомнить делитель в таблице простых множителей под своим номером.
    Подсчитать, сколько раз n делится на делитель (при этом каждый раз делить n на него). Запомнить это число в таблице показателей степени под номером простого множителя
  все
  делитель увеличить на 1
кц

Анализируя словесную запись операций, видим, что некоторые из них, например: «если n делится на делитель» или «делитель увеличить на 1», совершенно элементарны. В то же время операция «подсчитать, сколько раз n делится на делитель» вовсе не элементарна. Значит, она должна представлять собой вспомогательный алгоритм. Его более подробная запись представлена ниже.

Степени присвоить значение ноль
пока n делится на делитель
нц
  n присвоить значение n, деленное на делитель.
  Степень увеличить на 1
кц

Необходимость в разработке блок-схем отпадает, так как приведенные записи алгоритмов обладают не меньшей наглядностью.

Теперь без труда выполняется следующий этап — полная запись алгоритма. Здесь обнаруживается еще одно преимущество наложенного запрета на использование математических обозначений. В результате этого запрета переменные были обозначены не одной буквой (как это принято в математике), а одним или даже несколькими словами, выражающими основное назначение переменной. Следует сохранить эти обозначения, тогда алгоритм будет легко читаться в соответствии с современными требованиями к разрабатываемым программным средствам.

```
алг разложение (нат n, количество простых множителей, нат таб простой множитель [1:20], показатель степени [1:20])
  арг n
  рез количество простых множителей, простой множитель, показатель степени
нач нат делитель, номер простого множителя
  делитель: =2; номер простого множителя: =0
  пока n>1
  нц
    если [n/делитель] = n/делитель
    то номер простого множителя: = = номер простого множителя +1
      простой множитель [номер простого множителя]:=делитель
      показатель степени [номер простого множителя]:=степень (n, делитель)
    все
    делитель:=делитель+1
  кц
  количество простых множителей:=номер простого множителя
кон
  алг нат степень (нат n, делитель)
  нач
    знач:=0
    пока [n/делитель] = n/делитель
    нц
      n:=n/делитель;  знач:=знач+1
    кц
  кон
```

Мы не приводим обоснования алгоритма — читатель легко сделает это сам. Отметим только, что лишние проверки делимости на составные числа компенсируются отсутствием необходимости создания таблицы простых чисел.

Е. БАКМАН, А. ПЕНЬКОВ

Открытое письмо

Уважаемые коллеги!

Как известно, в настоящее время, в соответствии с решениями руководящих органов, проводится работа по внедрению вычислительной техники в народное образование, что является одной из важнейших сторон информатизации советского общества.

Мы, методисты и заведующие кабинетами ИВТ восьмидесяти институтов усовершенствования учителей, то есть всех регионов РСФСР, крайне обеспокоены существующим положением дел, а именно:

1. Выпускаемая отечественной промышленностью вычислительная техника не обеспечивает потребностей учебного процесса из-за чрезвычайно низкой надежности, наличия конструктивных недостатков, отсутствия программного обеспечения.

2. Планы поставки вычислительной техники срываются заводами-изготовителями, что крайне затрудняет работу по опережающей и своевременной подготовке кадров на местах.

3. В создании программного обеспечения царит хаос — отсутствует планирование заказов, нет защиты авторских прав, не решены вопросы финансирования.

4. При существующей системе снабжения вычислительной техникой завод-изготовитель совершенно не заинтересован в укомплектовании своих изделий программным обеспечением, так как спрос и так гарантирован.

5. Поступающие в школы новые учебные классы «Корвет» и УКНЦ недоработаны. Так, например, компьютер УКНЦ через полтора часа работы перегревается и перестает работать, дисковод не приспособлен для работы с двойной плотностью записи. Период наработки дисковода на отказ, как показывает практика, 1—2 месяца. Ремонт дисководов на местах практически никто не может осуществить. Крайне плохо работает локальная сеть класса «Корвет». До сих пор не налажено сервисное обслуживание УКНЦ и «Корвет». Не учитывается необходимость работы школьной техники в режиме реального времени.

Убедительно просим редакцию журнала «Информатика и образование» найти возможность для реализации сбора объективной информации по вопросам внедрения вычислительной техники в народное образование. Может быть, объединив усилия всех заинтересованных сторон, удастся навести порядок в очень важном и нужном деле.

С уважением методисты и заведующие кабинетами информатики и вычислительной техники МНО РСФСР.

Л. ВОЛКОВА, С. КРАСИКОВ,
Г. КУЗНЕЦОВА, Т. КУШНИРЕНКО,
Б. МЕДВЕДЬ, М. РАТИНСКИЙ,
В. САВИЦКАЯ, А. СМИРНОВА,
Н. УГРИНОВИЧ, И. ЧЕРНИКОВА
И ДР. (ВСЕГО 109 ПОДПИСЕЙ)

Компьютерные игры и нравственный выбор

Распространенность компьютерных игр сегодня колоссальна — мировой объем их продажи оставляет позади и детскую литературу, и видеопродукцию. Соответственно огромно их влияние на формирование личности, так как их основной потребитель — дети, причем начиная с 5—6 лет. В то же время компьютерные игры — мощнейший инструмент ускорения развития интеллекта. Подтверждение тому — огромный успех компьютеризованных систем обучения с игровыми компонентами в Израиле и Японии, получивших наивысшие оценки в мире.

Однако история человечества с очевидностью показала, что развитие интеллектуальной мощи, не подкрепленное опережающим ростом нравственности, приводит к тяжелейшим последствиям! Сегодня же развитию интеллекта ребенка везде придается первостепенное значение, а задача нравственного совершенствования нередко отходит на второй план. Но если развитие интеллекта невозможно без постоянного интеллектуального тренинга, то и «воспитание чувств» неосуществимо без «тренировки» нравственности, без душевной работы.

В мире же наших, ограждаемых от сложностей жизни, детей проблемы нравственного выбора встречаются редко. Мир ребенка во многом стерилизован. Результат — уровень нравственности в мире падает.

Поэтому, учитывая выход компьютерных игр на видное место в формировании личности ребенка, предлагаю безотлагательно

ввести в них моральный и нравственный аспекты.

Основными принципами сценариев могут быть, например:

для самых маленьких — прямая «выгодность» добра, правды и любви;

для более старших — опосредованное торжество добра и справедливости, причем носящее вероятностный характер.

Неиссякаемым источником подобных сценариев является, безусловно, фольклор — сказки всех времен и народов проникнуты проблемами нравственного выбора, начиная с Библии и мифов Древней Греции и до русских народных сказок. В них таится неоценимый потенциал воспитания. Поэтому вовсе не обязательно интерьером компьютерной игры должно быть фантастическое окружение, как это сейчас бывает. Тем более если в нем находят отражение худшие из земных реалий, например «звездные войны».

Учитывая великую способность обучающихся «обходить» любые препятствия, не следует бояться того, что пользователи довольно быстро могут разобраться, что правда для них «выгоднее» лжи, а добро — «полезнее» зла. Ничего страшного, пусть уверенность в этом укрепляется. Главное, чтобы «выгодность» не была слишком жестко связанной с выбором и чтобы вероятность победы добра была, скажем, не 1, а 0,8, а может быть, и 0,51.

Короче говоря, надо — по мере роста сложности игр — стремиться ко все большей «правде жизни».

Неоднократно отмечалось, что можно с выгодой использовать наше отставание по уровню компьютеризации от развитых стран. Другими накоплен ценный опыт, и глупо его не использовать. Общеизвестен уже отрицательный психофизиологический эффект большинства из ныне существующих компьютерных игр. Думаю, социальный эффект не заставит себя ждать.

Чтобы овладеть ситуацией, необходимо незамедлительно, не дожидаясь насыщения отечественного рынка хорошими компьютерами, создать группу специалистов высшей квалификации, включающую сценаристов, режиссеров и художников, психологов и педагогов, программистов. Существовать подобное объединение могло бы, к примеру, под эгидой Союза кинематографистов и Гособразования СССР, тем более что технически задача близка к созданию компьютерных мультфильмов.

Уже в скором будущем, когда домашний компьютер, «поглотив» видео, вытеснит телевизор и кино, компьютерные игры превратятся в мощнейший инструмент воспитания. И если нам небезразличен результат воспитания, нужно хорошо овладеть инструментом.

В. КОРЕЦКИЙ

ЧТО МОЖЕТ ЭВМ?

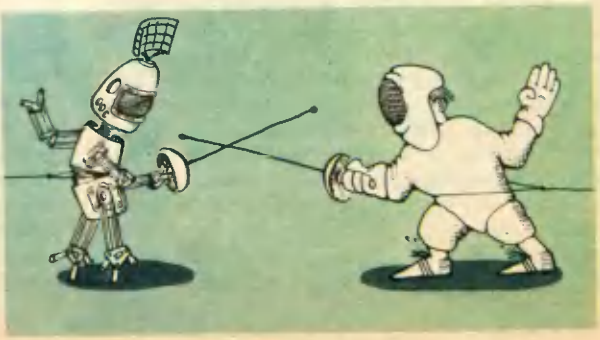
Кто лучше фехтует?

О том, что может ЭВМ в Японии, регулярно и не без восхищения рассказывают многие издания мира.

Вот, например, фирма «Исидзоку Кокурай» разрабатывает тренажер фехтовальщика. Робот в виде фигуры человека среднего роста и веса будет работать по специальной программе.

Электронный тренер отключится, если фехтовальщик попадет ему в «голову», «грудь» или «живот», признавая тем самым свое поражение. Если же неудача постигнет спортсмена, отрабатывающего технику, в схватке с молниеносно и разнообразно реагирующим «электроником», робот воскликнет «Попал!» — и вспыхнет красная лампочка, закрепленная на нем.

Сообщение другой фирмы («Норицц»), скорее, вызывает улыбку — разработана кабина для автоматического купания людей. Введение шампуня и его расход, включение и выключение душа, сушка человека теплым воздухом, уровень пены — все это регулируется программой. А что же человек? Он нажимает соответствующие кнопки. Может быть, это и удобно, но фирма не сообщает, возможен ли переход к обычной процедуре мытья в случае отказа компьютера.



«Круглый стол» в Ленинграде

В декабре 1988 г. в Ленинграде состоялся «круглый стол» по обсуждению концепции информатизации образования, проведенный в рамках второй ленинградской научно-методической конференции «Проблемы образования в области информатики, вычислительной техники и автоматизации».

Вел «круглый стол» доктор физико-математических наук Ю. В. Матиясевич (ЛОМИ АН СССР). Основной доклад сделал кандидат физико-математических наук С. Г. Смирнов (НИИ ШОТСО АПН СССР).

В дискуссии участвовали работники народного просвещения и научные сотрудники Ленинграда, Москвы и других ведущих центров развития образовательной информатики в СССР. Основой дискуссии послужил проект концепции информатизации образования, созданный Рабочей группой под руководством академика А. П. Ершова по заказу Гособразования СССР. Основной целью дискуссии было выявить недостатки предложенного варианта концепции и определить направления его улучшения.

Участники «круглого стола» ознакомились с содержанием проекта концепции информатизации образования, сочли этот проект интересным, а его публикацию для широкого обсуждения — своевременной и полезной. Вместе с тем, по мнению участников дискуссии, предложенный проект имеет ряд существенных недостатков, содержит много спорных утверждений и поэтому нуждается в значительной доработке.

Основные замечания участников дискуссии по проекту концепции таковы.

Проект рассматривает лишь один вариант информатизации народного образования — массированное внедрение ЭВТ и нового учебного предмета во все школы страны. Проект игнорирует ряд альтернативных вариантов информатизации образования, таких, как

факультативное преподавание ОИВТ по решению советов школ или региональных советов народного образования;

обязательное изучение курса ОИВТ в тех и только тех учебных заведениях, которые оснащены доброкачественной УВТ и соответствующим программным обеспечением;

выбор содержания курса ОИВТ и формы его изучения по решению региональных Советов народного образования;

дифференциация содержания курса ОИВТ по глубине его изучения, по уровню требований к знаниям и умениям учащихся в зависимости от их личных склонностей.

Практическое использование концепции информатизации образования в учебной и организационной работе потребует от нее большей гибкости и многовариантности.

Авторы проекта декларируют (в рамках избранного ими варианта развития образовательной информатики) четкую систему последовательных целей информатизации образования во всех звеньях существующей учебной системы. Однако проблема ресурсного обеспечения этих целей и их достижимости в намеченные сроки изучена недостаточно, и в этом плане многие утверждения данного проекта вызывают сомнение.

В проекте не рассмотрены варианты деятельности образовательных учреждений в том случае, если задания последующих пятилеток по оснащению школ УВТ будут сорваны промышленными организациями так же, как это было в 1985—1988 гг.

Авторы проекта подробно анализируют динамику оснащения учебных заведений ЭВТ, но уделяют гораздо меньше внимания свое-

временному синтезу и производству соответствующих ППС, а также обучению основной массы педагогов пользованию этими ППС и в дальнейшем их самодеятельному педагогическому творчеству в этой сфере. Решение этой проблемы также требует своевременного выделения значительных ресурсов (финансовых, технических, интеллектуальных). Количественный анализ этих ресурсов и путей их мобилизации отсутствует в данном проекте концепции.

Информатизация народного образования — комплексный процесс, и управление им требует интенсивного соучастия ведущих научных центров страны, в первую очередь Академии наук и ведущих университетов. В предложенном проекте нет анализа динамики и ресурсного обеспечения таких связей между государственной системой научных исследований и общественно-государственной системой образования. Этот серьезный пробел должен быть устранен в рабочем варианте концепции.

В современной практике преподавания курс ОИВТ сводится, как правило, к изучению программирования. Это вынужденное искусственное сужение содержания курса уже привело к негативному отношению основной массы учителей к новому курсу, и такое отношение передается большинству учащихся. Чтобы переломить эту опасную тенденцию, необходимо зафиксировать в концепции право учителей на эксперимент в определении содержания и формы изложения учебного материала. Нужно также указать возможные направления педагогического поиска в изучении ВТ и в стыковке этого курса с другими учебными предметами. Эти рекомендации нужно обосновать разбором образцов передового опыта учителей-практиков в СССР и за рубежом. Отсутствие анализа таких образцов и тенденций развития образовательной информатики в США и других странах делает многие утверждения авторов проекта неубедительными для основной массы работников народного образования.

В данном проекте основное внимание уделено информатизации (точнее, компьютеризации) школы. Проблемы информатизации всех других звеньев системы народного образования рассмотрены очень бегло, а вопросы стыковки этих звеньев только намечены. Такой подход препятствует эффективному сотрудничеству, например, между школьной и вузовской системами изучения ИВТ.

Для содействия развитию таких связей во всей системе народного образования концепция должна содержать явную динамическую модель взаимодействия разных звеньев образовательной системы. Только при этом условии инициативные работники разного профиля смогут находить общий язык и сотрудничать в разработке полноценных образовательных проектов.

Известно, что ГКВТИ и ряд других ведомств разрабатывают свои варианты концепции информатизации нашего общества и его образовательной системы, независимо от данного проекта, выполненного по заказу Гособразования. Конкуренция нескольких проектов в этой сфере, а также их рабочая стыковка представляются необходимыми для достижения высокого научного качества и организационной полноты в окончательном рабочем варианте концепции. С этой целью участники «круглого стола» призывают к разработке и широкой публикации альтернативных вариантов концепции информатизации образования.

Выставка, которой не могло быть три года назад

В декабре 1988 — январе 1989 г. павильон «Народное образование» ВДНХ СССР притягивал к себе тех, кто учится, и тех, кто работает в сфере народного образования. Там проходила выставка «Компьютеризация образования (средства, методы, передовой опыт)», организованная Гособразованием СССР. В целом выставка правильно осветила сегодняшнее положение дел с внедрением вычислительной техники в учебный процесс: достижения и промахи, активность отдельных энтузиастов и тягу учащихся, трудности снабжения и низкое качество производимой техники. Можно сказать с уверенностью: учебный процесс во всех звеньях поворачивается лицом к компьютерам, а компьютеры, в свою очередь, заставляют учебный процесс подстраиваться к требованиям времени.

Трудно было три года назад предположить такое разнообразие типов компьютеров да и сам факт их проникновения в школы, ПТУ и техникумы. Но качество, надежность... И неспроста больше всего школьников и

преподавателей собираются у ярких экранов «Ямах»... Будем надеяться на успехи нашей электроники.

Вопросами: «Сколько стоит?», «А весь класс вместе?», «А каждая машина отдельно?», «На каком заводе выпускают?», «Где учат работать с ними?» и т. д. посетители выставки буквально засыпали немногочисленных стендистов. Будучи методистами, программистами и сотрудниками НИИ, они при всем желании не всегда могли дать ответа на эти, казалось бы, простые вопросы, особенно про деньги. Можно было бы всю эту информацию дать на табличках, да и сами таблички сделать поярче, как, например, у конструктора микро-ЭВМ «Электроника К-001 «Дон», хотя и здесь нет цены и адреса торгующей организации.

Нет необходимости обращаться к посетителям выставки, чтобы утверждать: «Техники не хватает, не хватает катастрофически!» Из-за ее нехватки изобретаются такие поделки, как «деревянная модель» компьютера Омского педагогического института, или оборудуются, как в Азербайджанском центральном институте усовершенствования учителей, передвижные классы в автобусах. Не от хорошей жизни появились на выставке эти экспонаты, и демонстрируют они не только оригинальные решения авторов, но и нерешенные проблемы нашей компьютеризации.

На этом фоне становится особенно дорога та техника, которая есть. И на ней «народные умельцы» воистину творят чудеса. Об этом говорят многочисленные обучающие, игровые и методические разработки. Представлены практически все регионы страны, авторами разработок являются научные работники и сельские школьники, аспиранты и учителя, преподаватели вузов и учащиеся ПТУ. Десятки пакетов программ описаны в каталоге программных средств. Прочитав его, можно сделать вывод — проблема с программными средствами решается. Приходи, школьный учитель, и бери всё, что нужно. Именно так было бы три года назад.

Но сейчас... На выставке ничего никому не дают, только сообщают координаты организаций и кооператива, у которых представленные программы можно купить. А разве в школе появились средства на их закупку? Каким же образом получать школьному учителю все эти оригинальные редакторы, базы данных, средства программной и методической поддержки школьных предметов? Ответа на этот вопрос выставка не дает. И что думают об этом в двух госкомитетах СССР — по народному образованию и вычислительной технике?

Программ представлено много. Но выставка еще раз показала, что ценность компьютеров определяется квалификацией людей, работающих с ними. Большинство программных средств до посетителей выставки не добралось — они были скрыты в недрах дискет. И огромным успехом пользовались программные средства и методические разработки, которые демонстрировали их авторы. Тут «приз зрительских симпатий» поделили между собой Г. А. Кручинина из г. Горького и ее совместные со студентами методические разработки по школьным урокам химии, а также психодиагностическая

система «Поиск» Львовского политехнического института.

Мало разработать хорошие программно-педагогические средства, нужно научить преподавателей применять эти средства, а лучше всего — показать в деле. Именно это и сделала опытный педагог и хороший методист Г. А. Кручинина, и поток восторженных отзывов в книге посетителей — яркое тому подтверждение.

Читая эту книгу, нельзя было не отметить: на каждой странице «Поиск», «Поиск»... И хотя рабочий день уже заканчивался, возле стенда толпилась большая группа ребят. Каждый умолял проверить его тоже, ведь вопрос «Кем быть?» волнует всех школьников VIII—X классов. Компьютерные тесты, которые демонстрировали Г. В. Католик и А. Б. Пилинчук, интересовали всех. Я тоже не удержался и сел за клавиатуру. Рекомендация, выданная компьютером: математика, совпала с записью в моем дипломе... Но и эту разработку можно только купить, а нужна она в каждой школе, ПТУ, вузе...

Успех некоторых экспонатов во многом объясняется энтузиазмом их авторов. Как и любое дело, наше тоже держится на энтузиастах, и пока в основном на них. Получат ли авторы лучших разработок какие-либо награды? Конечно, не ради наград они тратят свое личное время, часами отвечая на вопросы посетителей, за дело обидно...

В. КОЗЛОВ

Координация разработок программного обеспечения

В настоящее время разработка ППС проводится большим количеством коллективов: НИИ, университетами, педагогическими институтами, другими организациями, а также отдельными разработчиками в частном порядке, но эффективной системы в этой области пока не создано.

Существующий механизм координации основан на формировании и реализации координационных планов. Первый план 1986 г. представлял собой перечень уже разрабатываемых программ, планы 1987 и 1988 г. составлялись на основании заявок организаций, ведущих разработки ППС. Функции головной организации по составлению и контролю за выполнением этих планов были возложены на НИИ ИВТ АПН СССР.

Для реализации координационного плана 1987 г. был разработан временный порядок организации работ по созданию ППС в рамках КП «ЭВМ в школе». Его основой являлось заключение творческих договоров о сотрудничестве с соисполнителями. Все работы проводились по планам работ организаций без целевого финансирования. Практически этот план был выполнен, за исключением разработки ППС для ЭВМ «Корвет» и УКНЦ из-за непоставки этих видов техники. Однако в процессе реализации этого плана стало очевидно, что своевременная и качественная разработка необходимого количества ППС возможна только при обеспечении разработчиков достаточным количеством инструментальной техники и целевом финансировании работ.

Поэтому при формировании координационного плана на 1988 г. в рамках направлений, согласованных с Управлением информатики и вычислительной техники Минпроса СССР, большинству инициативных разработчиков предполагалось выделить дополнительное количество инструментальной техники и средств финансирования. Координационный план был сформирован, но дополнительных средств и ресурсов выделено не было. Кроме того, практика составления этого плана показала, что требуется создание специального организационно-экономического механизма, позволяющего координировать и объединять усилия организаций-исполнителей и системы информационного и ресурсного обеспечения разработчиков. Отсутствие такого механизма и целевого финансирования привело к тому, что реализация координационного плана 1988 г. оказалась невозможной.

Так из 77 тем, включенных в координационный план 1988 г., фактически выполняется около 30. Большая часть организаций сняла свои предложения из-за отсутствия финансирования работ и поставок инструментальной техники. Многие организации не предоставляют в головную организацию сведения о выполнении заданий.

Таким образом, координация разработок ППС в рамках формирования и реализации координационных планов оказалась **малоэффективной.**

На состоявшемся в октябре 1988 г. координационном совещании разработчиков педагогических программных средств, проводимом на базе НИИ ИВТ АПН СССР был осуществлен анализ выполнения координационного плана и сложившейся системы координации в области создания программных средств (ПС) для обеспечения учебного процесса в целом.

Работа совещания проводилась в рамках лаборатории-семинара с тематическим названием **«Проблемы координации научно-исследовательских программ в области компьютеризации образования».** Проблематика обсуждалась одновременно в пяти группах по следующим направлениям:

1. Компьютеризация обучения общеобразовательным предметам и универсальным видам деятельности.

2. Компьютеризация трудового обучения, профориентации и профессионального обучения.

3. Компьютеризация управления школами и органами народного образования, повышение квалификации работников народного образования.

4. Компьютерные средства обеспечения курса и факультатива по ОИВТ.

5. Инструментальные средства для разработчиков программно-методического обеспечения учебно-воспитательного процесса.

На совещании были обсуждены итоги деятельности в области координации разработок ППС: определены потребности разработчиков в координации; содержание, статус и организационно-экономические механизмы координационной деятельности.

С точки зрения разработчиков ППС выделены следующие основные потребности в координации исследований и разработок:

создание системы информационного обеспечения;

организация маркетинга;

создание и реализация организационно-экономического механизма финансирования исследований и разработок и обеспечения их средствами ВТ;

создание и реализация системы экспертных оценок для определения приоритетных направлений исследований и разработок и для распределения на этой основе средств финансирования и средств ВТ;

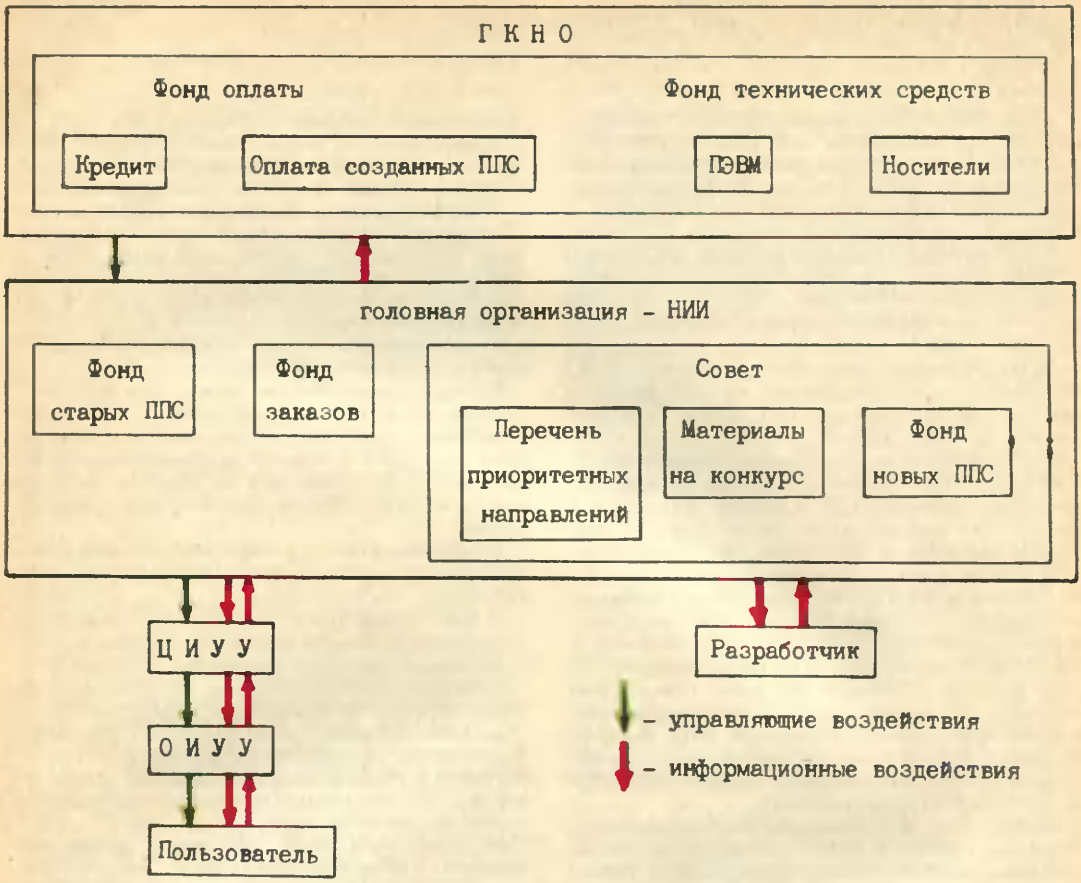
создание и реализация системы регулярных совещаний, конференций, семинаров (обмен опытом).

Информация, интересующая разработчиков и потребителей ППС, предварительно определена по следующим направлениям:

1. Разработанные программные средства.
2. Разрабатываемые программные средства.
3. Исследования в области создания ППС

Материалы подготовлены членами оргкомитета Координационного совещания разработчиков ППС, проходившего в г. Новосибирске 20—23 октября 1988 г.— *Прим. ред.*

Модель № I координации исследовательских разработок в области создания ППС



в стране и за рубежом (по планам НИР, комплексным программам, инициативным разработкам и т. п.).

4. Приоритетные направления исследований и разработок.

5. Сведения об организациях и специалистах, которые ведут разработку, об их оснащенности средствами ВТ.

6. Сведения о потребителях ПС и их оснащенности средствами ВТ.

7. Предложения и заявки по разработке ПС от разработчиков.

8. Заказы на разработанные ПС и на новые разработки от потребителей.

9. Сведения о распределении государственных средств финансирования и ВТ на разработку ПС.

10. Сведения о тиражах ПС.

Были предложены следующие формы представления информации: информационные бюллетени, информационные диски, публикации в прессе, в частности в журнале «Информатика и образование», сообщения по телевидению и радио, реклама на конференциях и совещаниях, рассылки по информационным системам, в том числе с использованием телефонных каналов связи между ЭВМ, справки и обзоры по заказам.

Вся информация может храниться в информационном банке или ряде банков в зависимости от организационной структуры системы компьютеризации.

Необходимо отметить, что столь детально переперечисленные потребности в информации охватывают широкий круг задач, требующих специальной проработки. Эти задачи не могут быть решены в комплексе ни в одной из существующих в настоящее время организационных форм. С другой стороны, это далеко не все потребности в информационном обеспечении как разработчиков, так и потребителей ППС. Например, существует информационный голод на сведения о нормативных актах, постановлениях и решениях, на сведения о нормативах для разработчиков, на сведения о результатах внедрения ППС в учебный процесс и т. д. Более того, с развитием информационного обеспечения появятся новые потребности в информации.

Потребность в создании и реализации экономического механизма финансирования исследований и разработок определена во многом тем, что в настоящее время практически отсутствует целевое финансирование разработок ППС, за исключением финансирования работ в НИИ ИВТ АПН СССР и в ряде НИИ и вузов по отдельным темам. При этом практически средства

выделяются в составе госбюджетных ассигнований на содержание организаций, а не для выполнения конкретных тем. Распределение осуществляется в основном методом администрирования. При наличии целевого финансирования из государственного бюджета вопрос эффективного распределения средств встанет еще острее. Этим вызвана потребность в создании и реализации системы экспертных оценок для определения приоритетных направлений исследований и разработок и для распределения на этой основе требуемых средств. Фактически необходимо создание социального механизма, обеспечивающего привлечение к решению наиболее важных задач компетентных лиц.

Одним из важных элементов при распределении средств должен стать конкурс проектов и предложений для разработок ППС.

Многие разработчики считают необходимым осуществлять государственное финансирование потребителей, т. е. непосредственно школ или районных (областных) отделов образования, которые могут объединяться в советы потребителей. В число потребителей могут входить ИУУ.

В то же время разработчикам должна быть представлена возможность самофинансирования на основе хозрасчетной деятельности. В этом случае потребители будут иметь возможность выбора и закупки наиболее качественных ППС.

В условиях явно выраженного дефицита средств ВТ их распределение предложено осуществлять в том числе и на основе аренды под конкретные разработки.

Участникам совещания было предложено несколько схем координации исследований и разработок в области создания ППС. Обобщение этих схем позволит выделить две основные модели. Модель 1 (рис. 1) представляет собой линейную субординационную схему с управлением «сверху вниз» через головную организацию. Для этой модели были отмечены следующие недостатки:

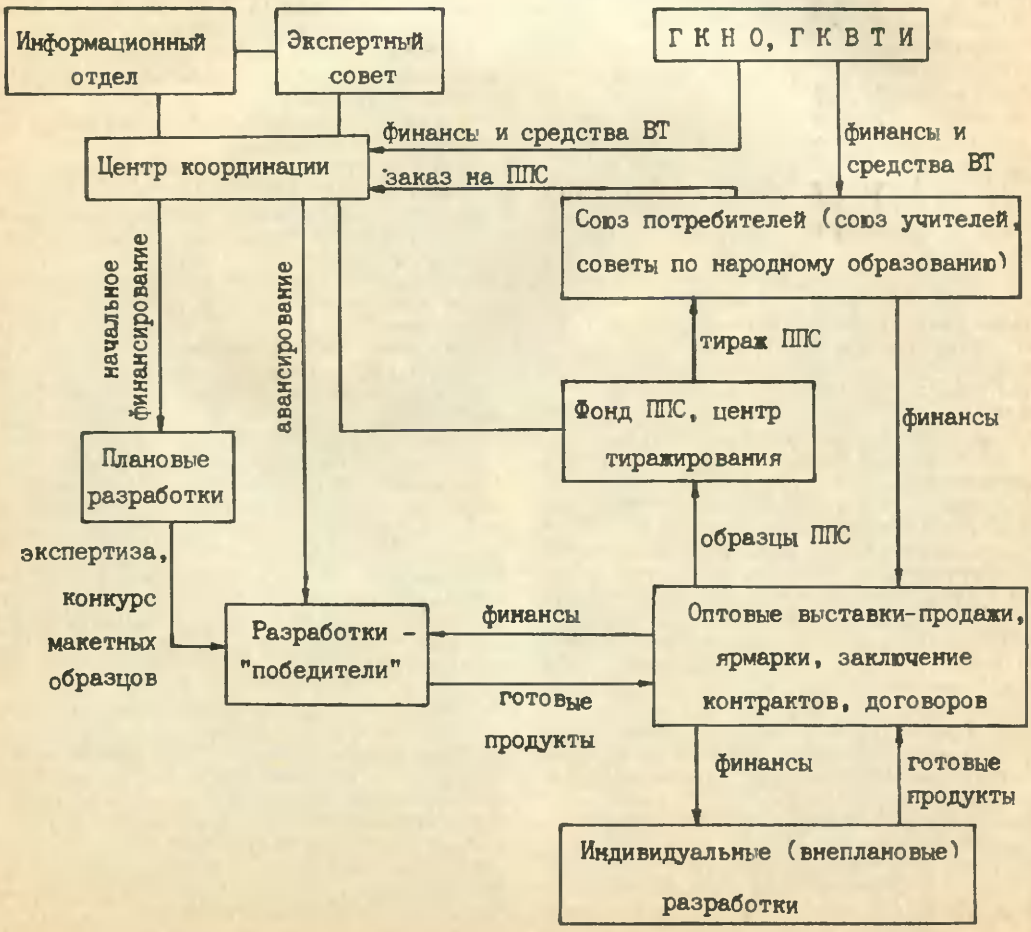
отсутствие реальной гибкой связи между разработчиками и пользователями;

отсутствие оперативного механизма стимулирования разработчиков ППС;

излишняя централизация и, следовательно, потенциальная возможность для принятия медленных и неверных административных решений;

поручение НИИ — головной организации не

Модель № 2 координации исследований и разработок в области создания ППС



свойственной ей функции практической координационной деятельности.

Модель 2 (рис. 2) основана на сочетании модифицированной субординационной схемы (через координационный центр) и схемы самофинансирования разработчиков посредством прямых связей с потребителями через договоры, контракты, выставки и ярмарки. При этом необходимо, чтобы потребители ППС имели экономическую самостоятельность с целевым государственным финансированием, ресурсным обеспечением. Естественно, что и разработчики должны иметь полную экономическую самостоятельность, вплоть до получения кредитов в банке при необходимости.

Для этой модели были отмечены следующие недостатки:

невозможность ее быстрой реализации в полном объеме, поскольку она требует изменения в существующей системе управления народным образованием;

неопределенность вопроса о финансировании фундаментальных разработок в рамках данной модели;

отсутствие проработанного организационно-экономического механизма для реализации этой модели.

В целом участники совещания пришли к выводу, что для реализации действенного управления процессом разработок педагогических программных средств необходимо создание научно обоснованной системы координации разработок ППС, учитывающей потребности разработчиков и потребителей.

Основными задачами координации определены:

1. Информационное обеспечение разработок.
2. Ресурсное обеспечение.
3. Создание организационно-экономического механизма взаимодействия между разработчиками, а также между разработчиками и потребителями ППС.
4. Создание системы определения приоритетных направлений разработки ППС.
5. Создание системы экспертной оценки ППС.
6. Организация системы внедрения ПС в практику учебного процесса и анализа полученных результатов.
7. Организация системы обмена опытом в области разработки ППС.

Выполнение этих задач невозможно в рамках реализации координационного плана — необходимо создание целевой координационной программы, охватывающей весь процесс создания педагогических программных средств, от планирования разработок до их внедрения в практику учебного процесса. Такая программа должна охватывать разработку педагогических программ, инструментальных и базовых программных средств, программных средств управления школами и органами народного образования и их методическое обеспечение.

Задачу формирования целевой координационной программы и организацию работ по ее реализации мог бы взять на себя координационный центр.

Необходимо отметить, что участники координационного совещания проработали ряд содержательных вопросов по осуществлению системы координации между разработчиками в рамках намеченных пяти направлений. Почти в каждой группе к концу работы совещания были определены и подготовлены предложения в проект решения, содержащие как общие положения, так и целый ряд частных, но очень важных конкретных предложений по вопросам взаимодействия разработчиков. Всего в работе совещания приняли участие 89 специалистов из 54 организаций, а также представители ГКНО СССР и ГКВТИ СССР.

Совещание приняло следующие рекомендации:

1. Считать целесообразным создание сети региональных центров новых информационных технологий обучения, организацию работы которой осуществляет головное учреждение по компьютеризации среднего образования.

2. Финансирование целевых программ ГКНО СССР начиная с 1990 г. для организации фундаментальных исследований в области компьютерной технологии обучения и разработки программно-методического обеспечения должно распределяться гласно, на конкурсной основе.

3. Определение перспективных направлений, конкурсный отбор и оценку результатов фундаментальных и прикладных исследований организует Координационный совет ГКНО СССР, включающий представителей от каждой из союзных республик.

4. Необходимо создание информационного банка проектируемых и разработанных программно-методических средств и т. п. для пользователей и разработчиков отрасли народного образования.

5. Обратить внимание на разработку системы тестирования результатов обучения, воспитания и развития учащихся.

6. Проводить координационные совещания разработчиков ППС ежегодно.

7. Приоритетными направлениями разработок на 1989 г. считать:

разработку и внедрение программно-методического обеспечения имеющихся учебников по курсу ОИВТ;

разработку и внедрение программно-методического обеспечения курса ОИВТ в педагогических вузах;

проведение стандартизации разрабатываемых ПЭВМ, КУВТ и ППС для обеспечения возможностей настройки диалога на все языки народов СССР;

проведение классификации ППС.

8. Перспективными направлениями считать:

разработку альтернативных учебно-методических комплексов компьютерной технологии обучения для ОИВТ и других предметов естественнонаучного и гуманитарного циклов, а также комплексов инструментальных средств и других систем и подсистем;

разработку инструментальных средств создания ППС на базе проведенной стандартизации ПЭВМ, КУВТ и ППС.

От редакции. Необходимо отметить, что координационное совещание проводилось в форме организационно-деятельностной игры. Большинство участников столкнулись с ней впервые, что предопределило неприятие одних, энтузиазм других и неизбежное в подобных ситуациях столкновение мнений.

Свои впечатления об атмосфере совещания передает на третьей странице обложки наш фотокорреспондент Э. Б а ж и л и н.


PESYMBIAT

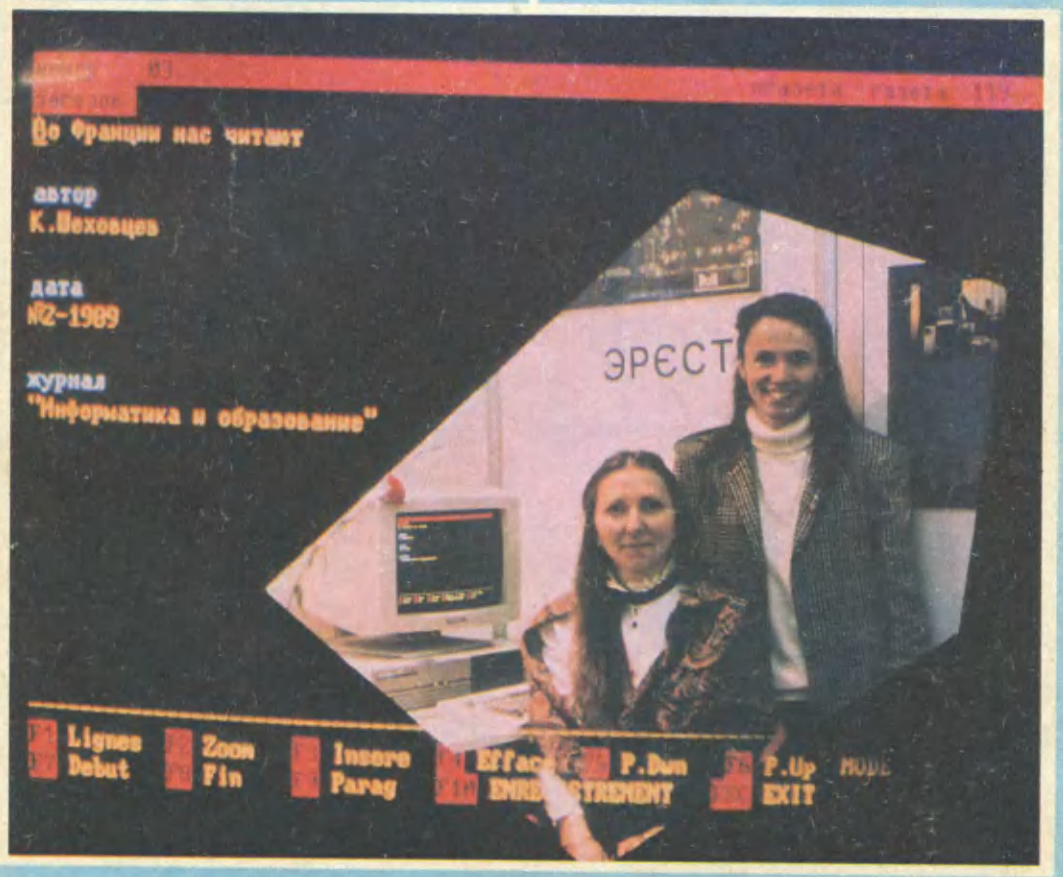
60 коп.
Индекс 70423

ИНФО

2'89

ИНФОРМАТИКА И ОБРАЗОВАНИЕ

База данных «Тексто» и фирма «Эрэстия» в репортаже номера



ISSN 0234—0453. Информатика и образование, 1989, № 2, с. 1—128