

В.Н. Решетников
А.Н. Сотников

Информатика- что это?



«РАДИО И СВЯЗЬ»

НАУЧНО-ПОПУЛЯРНАЯ БИБЛИОТЕКА ШКОЛЬНИКА

НАУЧНО-ПОПУЛЯРНАЯ БИБЛИОТЕКА ШКОЛЬНИКА

В. Н. Решетников
А. Н. Сотников

Информатика- что это?



МОСКВА
«РАДИО И СВЯЗЬ»
1989

ББК 32.81
Р 47
УДК 007 + 681.3

Рецензенты: д-р физ.-мат. наук С. А. Ашманов,
д-р физ.-мат. наук Е. А. Гребеников

Редакция литературы по информатике

Решетников В. Н., Сотников А. Н.

Р 47 Информатика — что это?.— М.: Радио и связь, 1989.—
112 с.: ил.— (Межизд. серия «Научно-популярная библио-
тека школьника»).

ISBN 5-256-00185-X.

Книга содержит первоначальные сведения об ЭВМ, истории, современном состоянии и направлениях развития информатики и вычислительной техники, знакомит читателей с основами построения алгоритмов автоматизированной обработки информации. В ней излагаются основы математического моделирования как важнейшего метода решения практических задач.

Для школьников старших классов, интересующихся развитием процесса компьютеризации и современной информатики.

1404000000-121

Р ————— 71-89
046(01)-89

ББК 32.81

ISBN 5-256-00185-X

© Издательство «Радио и связь», 1989

Эта книга посвящена информатике — новой области знаний, переживающей сейчас бурное развитие. В центре внимания информатики — современные электронные вычислительные машины (ЭВМ), или так называемые компьютеры, и процессы обработки информации с их помощью.

Хотя ЭВМ служат нам менее полувека, трудно найти сферу человеческой деятельности, где бы не применялись или не могли найти применение средства вычислительной техники. Новые информационные технологии представляют собой основу создания гибких автоматизированных производств, систем автоматизации научных исследований и проектирования, организационно-экономического управления, процессов обучения, развития сферы услуг и т. д.

Сегодня в сфере управления и обработки информации занято около четверти трудоспособного населения страны. По прогнозам специалистов, без широкого использования достижений информатики к концу века в этой сфере потребовалось бы занять и остальные три четверти.

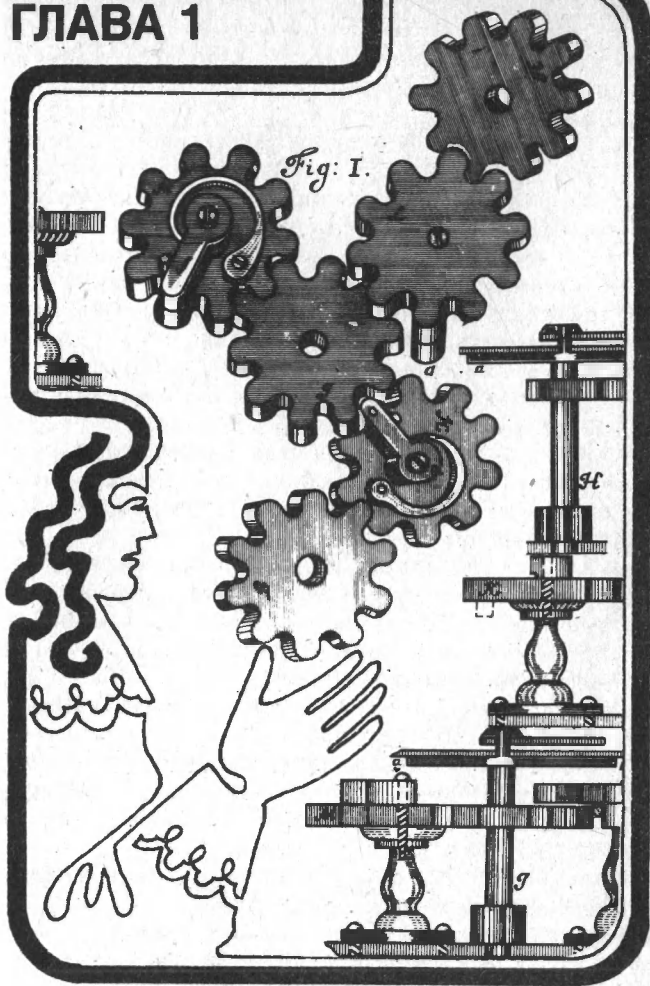
В наши дни происходит настоящая компьютерная революция. Во всем мире ЭВМ производятся миллионами экземпляров. Появились компактные персональные компьютеры (некоторые вполне помещаются в портфеле), а возможности их по сравнению с предшественниками, занимавшими целые комнаты, даже увеличились. Услугами такой ЭВМ может воспользоваться любой из нас, а не только специалисты в области электронной обработки информации.

Введение в программу средней школы предмета «Основы информатики и вычислительной техники» поможет достичь в нашей стране высокого уровня компьютерной грамотности, позволяющего раскрыть и использовать огромные возможности, заложенные в ЭВМ.

Эта книга адресована всем, кто проявляет интерес к самостоятельному овладению основами информатики, прежде всего — школьникам. Она будет полезна учителям, руководителям кружков в школах и внешкольных учреждениях, поможет в выборе тематики их работы.

Подробно рассказать об информатике в такой небольшой по объему книге, конечно, невозможно, но мы надеемся, что она станет еще одним шагом на пути компьютерного обучения.

ГЛАВА 1



СТАНОВЛЕНИЕ ИНФОРМАТИКИ

Когда говорят об информатике, то каждый понимает, что речь идет о какой-то информации, эта информация касается чего-то и т. д. Но как она получена, где и как хранится, как получить к ней доступ? Время меняет ответы на эти вопросы. В глубокой древности летописцы сначала на камнях, потом на глиняных табличках, позже на листах пергамента записывали информацию о своем времени и о делах своих современников. Записи хранились в монастырях, допускался к ним только определенный, очень узкий круг людей. Массы, не умевшие читать и писать, такой информацией пользоваться не могли. Сведения, помогающие выращивать хорошие урожаи, или секреты мастерства ремесленников, передавались устно из поколения в поколение и тщательно оберегались от конкурентов. Так зарождалась информатика.

Менялось время, менялись поколения, изменялся социальный строй общества. Рост населения требовал увеличения как ремесленного, так и сельскохозяйственного производства. Сохранять секреты в пределах одной семьи стало трудно, а часто и не нужно. Для обработки больших полей, плантаций, производства большого количества изделий потребовалось привлечь дополнительную рабочую силу и обучить ее эффективным приемам труда. Появилась новая информация — как лучше управлять коллективами, как добиваться наивысшей производительности труда и максимальной прибыли. Круг лиц, интересующихся таким опытом, расширялся. Актуальной стала задача размножения и распространения информации, зафиксированной на носителях, а это были в основном книги, рисунки и др. Так родилось книгопечатание. Среди первых книг, сначала рукописных, потом печатных, можно найти записи стихов, философские трактаты об устройстве мира и т. д. — человек стал записывать информацию не только об описании материального мира, но отражать свой духовный мир. Издание книг большим тиражом и их распространение дали возможность изучать достижения не только отдельных личностей, но и целых государств. Особенно широкое распространение получили книги по философии, литературе и искусству.

Был и другой вид информационной деятельности. Отдельные государства, стремясь к расширению своих территорий, проводили агрессивную политику по отношению к своим соседям. История изобилует захватническими войнами, не будем их вспоминать. Подготовка и ведение военных действий требовали информации о военном потенциале противника — какие и где расположены воинские части, чем они вооружены, какова выучка воинов, как долго сможет воевать противник, хватит ли у него

запасов, пороха, ружей, хлеба, мяса и т. д. Ее добывали, например, через разведчиков или покупали у предателей. Тогда остро встал вопрос защиты информации от утечки в посторонние руки. Стали развиваться методы кодирования, разрабатываться способы быстрой и безопасной пересылки информации.

Шли годы, рос объем информации, которой обменивалось общество. Для сбора, переработки и распространения информации создавались издательства, типографии — родилась информационная промышленность. Газеты, журналы, книги и другие издания, выпускающиеся большими тиражами, кроме полезной информации обрушили на человека огромное количество зачастую и ненужных, бесполезных сведений. Для обозначения таких лишних сведений придумали специальный термин «информационный шум».

Газеты сообщали, что и где произошло, кто и куда поехал, где, что можно купить и много других новостей, рассчитанных на массового читателя. Помимо печати появились и другие органы массовой информации — радио и несколько позже телевидение. И общество привыкло к тому, что когда говорят об информации, то речь идет о сведениях, полученных через газеты, радио и т. д. Затерялся основной смысл этого слова, утонул в потоке новостей, поступающих через органы массовой информации.

Что же означает «информация»? В Древней Греции в этот термин вкладывали смысл «придать форму», «обрисовать». А сейчас как это понимать?

Придать форму — это реализовать конструкторскую мысль в виде набора чертежей, снять кинофильм, специальную последовательность углублений на дорожке грампластинки преобразовать в мелодию и т. д. В этих примерах хорошо прослеживается идея преобразования наших наблюдений, знаний и т. д. из одной формы в другую, более удобную для работы, хранения или восприятия.

Обрисовать — можно понять как смоделировать, т. е. построить модель. Какую модель, как она представлена и в какой форме — это зависит от конкретной задачи, от языка и метода, применяемых исследователем. Например, рисуя портрет человека, художник не может изобразить его полностью (хотя бы из-за того, что изображение на холсте является плоским, а не объемным), поэтому он отбрасывает отдельные, несущественные, на его взгляд, детали и рисует конкретную модель. Насколько эта модель близка к объекту, зависит от мастерства художника и его способа представления информации. Физик, изучая следы, оставленные элементарными частицами на фотопластинке, строит из формул и уравнений математическую модель, психолог строит описательную модель поведения человека в различных ситуациях и т. д. Практически каждый человек в своей работе в той

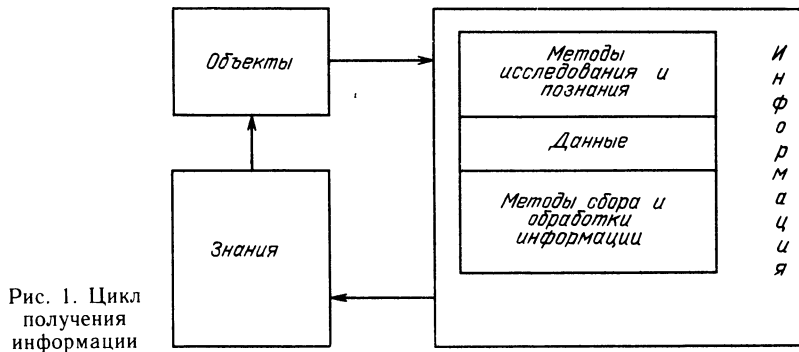


Рис. 1. Цикл получения информации

или иной форме занимается моделированием или описанием поступающей, или, как еще говорят, входной, информации.

Бурное развитие науки и промышленности в XX в., неудержимый рост объемов поступающей информации привели к тому, что человек оказался не в состоянии воспринимать и перерабатывать все ему предназначенное. Возникла необходимость классифицировать поступления по темам, организовать их хранение, доступ к ним, понять закономерности движения информации в различных изданиях и т. д. Исследования, позволяющие разрешить возникшие проблемы, стали называть информатикой. На данном этапе информатика являлась базой библиотечного дела и многие годы занималась теорией и практикой его совершенствования. Мы будем называть ее информатикой-0 как начальный, нулевой этап современной информатики.

Информатика-0 занимала странное промежуточное место между изучаемыми объектами природы и знаниями о них. Действительно, человек, изучая объекты окружающего мира, получает информацию, которую фиксирует на каких-то носителях (литература, магнитные ленты, карты, схемы и т. д.). Обработывая информацию, мы получаем знания об окружающем нас мире, позволяющие создавать новые методы исследования, получать новую информацию, фиксировать ее, обрабатывать и т. д.

Схематично этот замкнутый цикл изображен на рис. 1. Естественно, хочется назвать информатикой тот круг вопросов, который связан с разработкой эффективных методов сбора, хранения, обработки и преобразования имеющейся информации в знания, т. е. с обеспечением связей цепочки «Информация ↔ Знания», а не только с изучением, где и в каких журналах чаще появляются статьи по данной теме, как лучше расставить книги, каталожные карточки и др.

До 50-х годов нашего столетия такая постановка вопроса была неправомерной, так как не существовало почти ничего общего в методах сбора и обработки информации у медиков,

географов, психологов, физиков, филологов и т. д. С этой точки зрения много общего между собой имели математика и физика, химия и медицина. Примеров отдельных связей было много, но общего стержня, вокруг которого объединились бы все науки, не было (методологическим стержнем всех наук является диалектический материализм, но здесь мы говорим только о методах сбора и обработки информации). Положение существенно изменилось с рождением ЭВМ.

Широко известно, что первые ЭВМ создавались для проведения расчетов в атомной физике, в летательной и ракетной технике. Последовавшее далее внедрение ЭВМ в области административного управления и экономики дало не только большой экономический эффект, но и привело к созданию и бурному росту новой промышленной отрасли — средств и методов электронной обработки информации.

Появились новые ЭВМ, новые методы и средства общения с ними. Возникла новая информационная промышленность, производящая дорогую и малоосязаемую продукцию. Информация стала товаром. Электронно-вычислительные машины, созданные первоначально для решения вычислительных задач, стали обрабатывать числовую, текстовую, графическую и другую информацию.

Вычислительная техника сразу же показала всю эффективность в тех областях человеческой деятельности, где широко использовались методы математического моделирования — точные количественные методы. Сюда относятся физика, механика, химия, геофизика и т. д. Но есть такие области человеческой деятельности, которые еще недавно считались недоступными для методов математического моделирования, а следовательно, и для ЭВМ (ботаника, юридические науки, история и т. п.). В них шло накопление отдельных фактов, давалось качественное описание объектов и событий. Их и называли описательными науками. Развитие электронно-вычислительной техники, средств и методов общения с ней, создание автоматизированных информационно-поисковых систем, методов распознавания образов привели к тому, что ЭВМ стали способны проводить описательный анализ изучаемых объектов. Появилось новое направление исследований — разработка машинного (искусственного) интеллекта. Описательные науки получили ЭВМ в качестве нового рабочего инструмента. Никого сейчас не удивляют сообщения такого типа: «Проведенный с помощью ЭВМ анализ показал, что такое-то стихотворение не принадлежит Шекспиру», или «Ученые, обработав на ЭВМ портрет Леонардо да Винчи и изображение Монны Лизы на его картине, утверждают, что везде изображено одно и то же лицо».

Миниатюризация средств вычислительной техники, снижение ее стоимости позволили создавать станки с программным управлением, гибкие автоматизированные производства, станки-

роботы, сейчас уже говорят о заводах-роботах. ЭВМ обслуживают животноводческие фермы — собирают информацию о состоянии животных, составляют оптимальные рационы кормления, управляют различной обслуживающей техникой. Во всех примерах ЭВМ решает задачи сбора, хранения, обработки, преобразования информации и на основании ее анализа принимает соответствующие решения. В более сложных задачах человек, используя электронную технику, берет ответственность за принятие решения на себя. Электронный помощник анализирует громадные объемы информации и предлагает возможные варианты. Человек, познакомившись с этими вариантами, либо выбирает лучший с его точки зрения, либо ставит перед машиной новые условия и ждет следующего совета. Так, в режиме диалога происходит процесс принятия решения.

Что же такое информатика? Если это сбор и обработка информации об окружающем нас мире, то как отличить ее от химии, физики, геологии и остальных наук? А может быть, все остальные науки являются ее составной частью? Нет, информатика не включает в себя ни химию, ни физику, ни медицину и т. д., хотя с каждой имеет много общего. Она существует для помощи другим наукам и вместе с математикой снабжает их методами исследований и обработки информации.

Действительно, проводя эксперимент, ученый с помощью различных датчиков получает информацию, которую надо принять и записать, обработать по специальных алгоритмам, преобразовать к удобному для анализа виду. Далее, исследуя полученные результаты, необходимо сделать выводы.

Мы специально говорим об абстрактном эксперименте, он может быть физическим, геофизическим, химическим и т. д. Датчики могут передавать данные прямо в ЭВМ, а может быть и так: лаборант записывает показания приборов в тетрадь, а потом вводит их в машину. Главное в том, что нужны алгоритмы сбора данных и записи их в запоминающие устройства (ЗУ) в таком виде, который позволяет находить эти данные повторно, считывать и анализировать.

Другой важнейшей составной частью эксперимента является обработка данных. Откуда появляются алгоритмы обработки информации? Их готовят ученые и специалисты, проводящие эксперимент. Вместе с программистами они составляют программы для ЭВМ и обрабатывают данные. На этом этапе основное — разработка алгоритмов и составление на их основе программ для вычислительной машины.

На следующем этапе активно используются программы преобразования данных к удобному для исследования виду (построение графиков, таблиц, рабочих чертежей и т. д.) и их выдача. Как правило, такие программы не ориентированы на конкретную предметную область, они достаточно универсальны.

Таким образом, мы выделили задачи, которые являются общими для всех наук при обработке информации с помощью ЭВМ. В конкретных науках решение каждой из них имеет свои особенности, учитывающие специфику изучаемого предмета.

Но мы забыли про ЭВМ. Электронно-вычислительная машина является ядром всех наших рассуждений. Следовательно, ее рождение, становление и обучение — тоже предмет нашего исследования. Практика показала, что использование ЭВМ резко повысило производительность труда на производстве и в науке, оказало сильное влияние на научно-технический прогресс. Но существует обратное влияние — задачи науки и практики предъявляют конструкторам и разработчикам программ требования для создания новых, более высокопроизводительных ЭВМ, ориентированных на решение конкретных проблем.

В развитии ЭВМ можно выделить три этапа: вычислительный, общепersonaционный и интеллектуальный. Наука и технология находятся сейчас на пороге третьего этапа — развития машинного интеллекта. Машинный интеллект войдет в жизнь в виде ЭВМ, выполняющих такие функции, которые ранее были привилегией работников умственного труда. Рождаются новые машины, создаются более совершенные программы, «растет» машинный интеллект — появляются новые возможности для исследования и познания окружающего нас мира.

Перечисленные ниже три направления: 1) разработка методов и алгоритмов автоматизированного сбора, хранения, поиска и передачи информации; 2) разработка методов и алгоритмов обработки и преобразования информации; 3) разработка технологии и электронно-вычислительной техники, позволяющих развивать первые два направления, — и составляют современную информатику.

В английском языке синонимом слова информатика является computer science (вычислительные науки), перевод которого недостаточно точно отражает суть предмета информатики, хотя смысл такого термина хорошо понятен. Термин информатика пришел к нам из французского языка, где он обозначает науки об ЭВМ и их применения.

1.2. КАК ПОЯВИЛИСЬ ЭВМ

Электронно-вычислительные машины — откуда они пришли в нашу жизнь, что привело к их рождению?

С древнейших времен люди пытались понять окружающий мир и использовать свои знания для защиты от всевозможных бедствий. Заметили, например, что приливы и отливы связаны с различными положениями Луны, и возник вопрос: «А можно ли построить математический закон изменения положения Луны и,

используя его, прогнозировать приливы?» Ученые составляли громадные таблицы, где фиксировали изменения лунных положений, которые использовали для проверки верности (истинности) предлагаемых различных формул движения естественного спутника Земли. Такая проверка опиралась на громадное число арифметических вычислений, требовавших от исполнителя терпения и аккуратности. Для облегчения и ускорения такой достаточно «механической» работы стали придумывать вычислительные устройства. Так появились различные счеты и другие механизмы — первые вычислительные машины.

Человечество развивалось, расширялось его научное любопытство, выросло математическое образование, появились возможности глубже заглянуть в природные процессы. Были построены достаточно точные математические модели физических явлений, например законы Ньютона, а новые открытия стали появляться не только как результат наблюдений, но и как результаты математической обработки и вычислительного анализа. Примером является работа французского астронома Леверье, в которой, анализируя таблицы движения планеты Уран и опираясь на огромное количество вычислений, предсказано существование планеты Нептун (неизвестной в то время) и указано, где ее искать.

В те годы уже был накоплен большой материал по математическому описанию различных явлений природы, но при его использовании нужно было проводить такое громадное количество вычислений, что время, требуемое для этого, превышало время жизни человека. Поэтому при решении практических задач математические описания (модели) подобного типа не использовали. Искали, как правило, такие упрощенные математические модели, которые были доступны с вычислительной точки зрения.

В XIX—XX вв. началось бурное развитие промышленности. При изготовлении сложных технических изделий нельзя было удовлетвориться, например, такими рекомендациями ученых: «Поверхность детали должна представлять эллиптический цилиндр $x^2/3,21 + y^2/6,12 = 1$ высотой 5,34, все числа даны в см (рис. 2). Рабочий, изготавливающий на станке деталь, не обязан знать формулу, по которой можно вычислить все размеры детали, но должен иметь заранее вычисленные конкретные значения (данные) для контроля выполняемой работы. Технический прогресс настойчиво ставил задачу — найти способы организации быстрого и надежного вычислительного процесса. Появились механические вычислительные машины, получившие массовое распространение в начале XX в.

Однако развитие автомобилестроения, авиации, первые работы в области ракетостроения, исследования атомного ядра, планирование экономики в больших регионах и т. д. не могли опираться

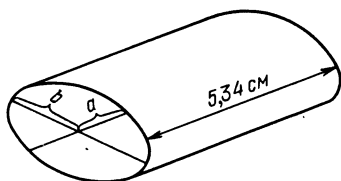


Рис. 2. Эллиптический цилиндр $x^2/a^2 + y^2/b^2 = 1$:
 $a^2 = 3,21 \text{ см}^2$, $b^2 = 6,12 \text{ см}^2$ — квадраты полуосей

на такую медленную и бедную по своим возможностям счетную базу, как механические вычислительные машины.

Именно в это время стали появляться первые проекты электронно-механических устройств. В 1937 г. проф. Дж. Атанасов (США) начал разработку арифметического устройства на специальных электромеханических блоках. В эти же годы инженер-энергетик С. А. Лебедев, будущий академик, приступил к разработке проекта первой советской ЭВМ. Вторая мировая война не дала Дж. Атанасову закончить свою разработку и задержала до 1947 г. работы по реализации проекта С. А. Лебедева.

Мощным импульсом для рождения первых ЭВМ послужили работы по созданию атомного оружия. В секретных лабораториях Лос-Аламоса (США) ежедневно проводилось большое количество вычислений, связанных со всеми этапами рождения первой атомной бомбы. Выдающийся физик Э. Ферми, обладавший незаурядными вычислительными способностями, неоднократно обращал внимание на необходимость создания такого автоматического вычислительного устройства, которое перестало бы сдерживать создание нового оружия¹. В этих условиях военного времени и развернулись работы по созданию первых ЭВМ «Марк-1», ENIAC, которые появились в начале 40-х годов.

В СССР в 1947 г. под руководством академика С. А. Лебедева были развернуты работы по проекту первой советской ЭВМ, которая была создана в 1951 г. и вошла в историю под именем МЭСМ (малая электронная счетная машина).

Сергей Алексеевич Лебедев родился 2 ноября 1902 г. в г. Горьком (Нижегород), в 1921 г. поступил на электротехнический факультет МВТУ им Н. Э. Баумана, после окончания которого работал во Всесоюзном электротехническом институте, где занимался решением задач устойчивой передачи электроэнергии на большие расстояния. Осуществление ленинского плана ГОЭРЛО привело к созданию единой энергосистемы на территории нашей страны, поэтому устойчивые и надежные

¹ В то время ученые Лос-Аламоской лаборатории опасались, что гитлеровская Германия создаст атомную бомбу и применит ее для завоевания мирового господства. Поэтому создание первыми ядерного оружия многими физиками в США связывалось с освобождением мира от фашистской чумы.

методы и средства передачи электроэнергии из районов Украины на Урал, объединение с энергосистемами Сибири имели важнейшее значение.

За заслуги в этой области Академия наук Украины в 1945 г. избирает С. А. Лебедева своим действительным членом (академиком).

Понимая, что для успешного решения важнейших народно-хозяйственных задач, в том числе и задач обороны нашей страны, необходима новая вычислительная техника, опирающаяся на электронику, С. А. Лебедев, будучи директором Института электротехники АН УССР, организовал в 1947 г. лабораторию по разработке макета МЭСМ. Так, в возрасте 45 лет Сергей Алексеевич принял решение резко изменить свои научные интересы и занялся новым, неизвестным направлением — созданием электронно-вычислительной техники.

Переехав в 1950 г. в Москву, С. А. Лебедев приступает к созданию ЭВМ БЭСМ-1 (БЭСМ — Большая электронная счетная машина).

Основу лаборатории составили молодые специалисты — выпускники Московского энергетического института, Московского университета и других вузов.

Академик В. А. Мельников, один из создателей современной советской электронно-вычислительной техники, так вспоминает об этом времени:

«В 1950 году я учился на V курсе Московского энергетического института на кафедре «Автоматика и телемеханика». Это был первый послевоенный выпуск. Предмета «Вычислительная техника» тогда еще не преподавали в институтах. Перед преддипломной практикой весной нас пригласили в деканат, где мы встретили человека в очках, невысокого роста, лукаво посматривающего на нас. «Ну, как, — спросил он, — будем делать большую электронную машину?» Мы дружно закивали, смутно представляя себе, что это такое — электронная машина.

Так произошло знакомство с Сергеем Алексеевичем Лебедевым — основоположником отечественной вычислительной техники, главным конструктором первой электронно-вычислительной машины в нашей стране.

Во время прохождения практики Сергей Алексеевич подробно рассказал нам, что такое ЭВМ, для чего она нужна, из каких устройств состоит и какое у нее быстродействие. Каждому практиканту определили тему дипломного проекта так, чтобы тема оправдала разработку какого-то узла или устройства будущей машины. В сентябре 1950 года нас, студентов-дипломников, приняли на работу как сотрудников Института, а в марте 1951 года мы успешно защитили свои дипломные работы, которые легли в основу проекта будущей машины БЭСМ-1. Сергей Алексеевич

всегда доверял молодым и опирался на молодежь. В нем естественно сочетались доброта и чуткость, высокая принципиальность и требовательность. Личный пример у Сергея Алексеевича был главным принципом воспитания».

Работы по созданию БЭСМ велись в Институте точной механики и вычислительной техники (ИТМ и ВТ), созданном в 1948 г. Первым его директором был известный советский математик Михаил Алексеевич Лаврентьев, который внес большой вклад в развитие вычислительной техники в СССР. С 1953 г. и до конца своей жизни ИТМ и ВТ возглавлял академик С. А. Лебедев. В этом институте под руководством Сергея Алексеевича было создано три поколения советских ЭВМ — это и знаменитая серия БЭСМ-1, -2, -3М, -4, -6 и машины М-20, М-220. Электронно-вычислительная машина БЭСМ-1 была не только первой советской быстродействующей ЭВМ, с производительностью 8...10 тыс. оп./с, но и самой высокопроизводительной машиной в Европе и одной из лучших в мире. Созданная в 1965 г. ЭВМ БЭСМ-6 до настоящего времени надежно и успешно работает во многих крупнейших вычислительных центрах нашей страны.

За работы по созданию советской электронно-вычислительной техники С. А. Лебедев в 1953 г. был избран действительным членом (академиком) Академии наук СССР, а в 1956 г. ему было присвоено высокое звание Героя Социалистического Труда.

Высокая стоимость, трудоемкость изготовления, нацеленность только на решение счетно-вычислительных задач сдерживали развитие парка ЭВМ. Так, в 1954 г. во всем мире насчитывалось примерно 100 ЭВМ. В последующие годы ситуация резко изменилась, и в 1985 г. число ЭВМ в мире превысило 10 000 000. В нашей стране были разработаны и выпускались ЭВМ «Стрела» с 1953 г., серия ЭВМ «Урал» с 1954 г., ЭВМ «М-20» с 1959 г., серия ЭВМ БЭСМ с 1952 г. и т. д. Необходимо отметить, что первые советские ЭВМ превосходили зарубежные вычислительные машины и по быстродействию, и по своим конструкторским характеристикам. Многие конструкторские решения, реализованные впервые в мире в ЭВМ серии БЭСМ, вошли в мировой фонд и используются до настоящего времени в лучших ЭВМ различных типов.

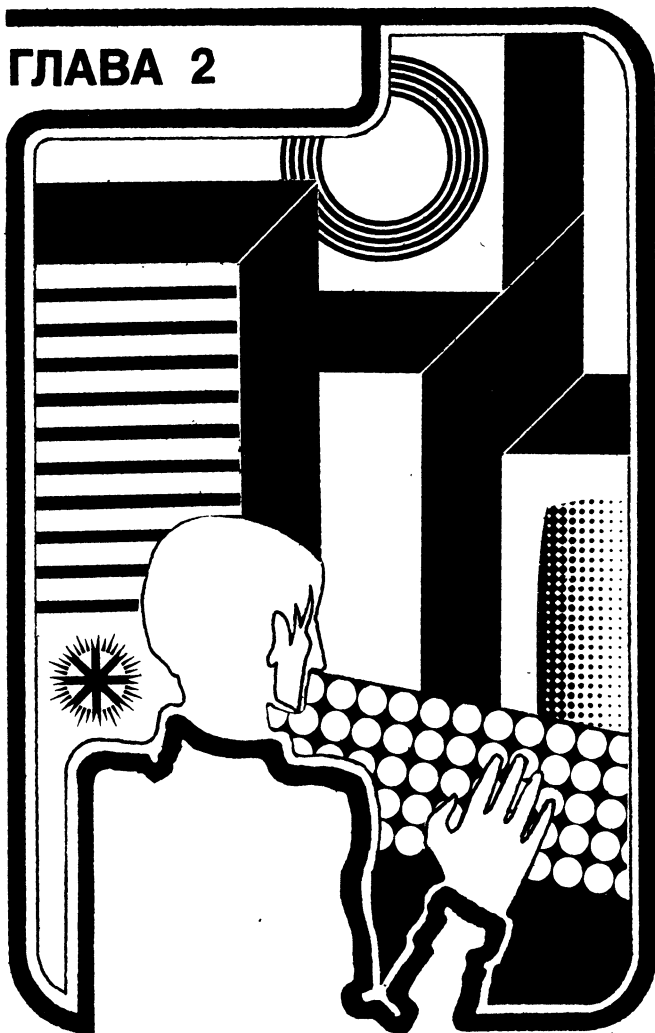
В мире нет других примеров такого долголетия, каким обладала БЭСМ-6 (конструкторы академик С. А. Лебедев, академик В. А. Мельников), — она серийно выпускалась с 1964 по 1984 г.

В начале 50-х годов открываются новые области применения электронно-вычислительной техники — управление и экономика. Своеобразие решаемых задач наложило отпечаток на архитектуру и типы ЭВМ. Появились классы больших, средних, малых, мини-машин, различающихся точностью вычислений, быстродействием, объемами памяти и т. д. В СССР с 1960 г. начался

выпуск ЭВМ серии «Минск», в основном ориентированной на обработку экономической информации.

Интенсивное развитие промышленности элементной базы электроники, создание и внедрение новых технологий позволили США и Японии в 60—70-е годы сделать резкий скачок вперед по внедрению и созданию новой вычислительной техники и обогнать нашу страну в этой области.

ГЛАВА 2



ПОРТРЕТ ЭВМ

Так что же такое ЭВМ? В этой главе мы познакомимся с различными ее портретами, и этому не надо удивляться. Когда вы, читатель, приходите фотографироваться, то прежде всего вас встречают вопросом: «На что сниматься желаете — на паспорт, на удостоверение или на память?». Все три перечисленных выше фотопортрета будут отличаться друг от друга не только размерами, но и скоростью исполнения заказа, стоимостью выполненной работы, способом представления информации. А портрет, сделанный врачом-рентгенологом при обследовании ваших легких, будет совершенно не похож на предыдущие.

Портреты ЭВМ также могут различаться. Мы построим портрет технических средств и их связей, потом перейдем к портрету программных средств, и все это будет различное видение одной ЭВМ. И это еще не все существующие портреты, созданные разработчиками элементной базы разработчиками архитектур ЭВМ, систем охлаждения и т. д. Совершенно особое место занимают портреты ЭВМ, созданные ее «пользователями» — физиками, медиками, геологами, историками, филологами и др. Эти портреты многообразны, удивительны, увлекательны и все вместе представляют собой большую часть современной информатики.

Электронно-вычислительная машина — это слово часто встречается в разговорах, в печати, звучит по радио и телевидению. Но встречаются еще и такие слова, как компьютер, вычислительная система. Эти слова — синонимы, смысл их одинаков.

Определим смысл этих слов. Вычислительная система (ЭВМ, компьютер) — это система по переработке информации, состоящая из:

аппаратных средств (электронные и электромеханические устройства);

программных средств (программное обеспечение);
документации.

2.1. АППАРАТНЫЕ СРЕДСТВА

Аппаратные средства любой ЭВМ включают в себя центральный процессор, периферийные устройства, каналы связи. Отсюда возникает первый грубый портрет ЭВМ, показанный на рис. 3.

Ядром, основной частью вычислительной системы является *центральный процессор (ЦП)*, в котором происходит обработка и преобразование информации. Он состоит из следующих основных блоков (рис. 4,5): арифметико-логического устройства (АЛУ); оперативного запоминающего устройства (ОЗУ); устройства управления (УУ).

В *арифметико-логическом устройстве* выполняются арифметические и логические операции. Кроме этого, АЛУ обрабаты-

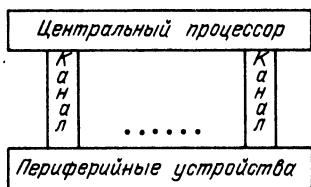


Рис. 3. Аппаратные средства ЭВМ (основные элементы)

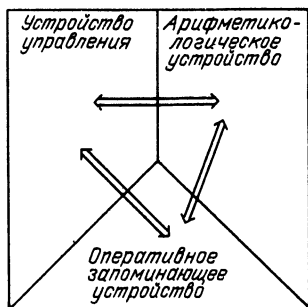


Рис. 4. Схема центрального процессора (стрелками указаны направления движения информации в ЦП)

вает отдельные управляющие сигналы, позволяющие УУ принимать решение о подготовке команд работы АЛУ в зависимости от полученных результатов вычислений.

Информацию для своей работы (операции, которые надо выполнить, и данные (как правило, числа), над которыми выполняется операция), АЛУ получает из ОЗУ.

Автоматические запоминающие устройства характерны только для ЭВМ. Ни одна вычислительная машина предыдущих поколений таким устройством не обладала. Именно автоматическая память, позволяющая запоминать, хранить и извлекать информацию о том, что надо сделать, какие данные использовать и как поступить с результатом, дала возможность решать электронному механизму задачи, ранее доступные только человеку.

Оперативное запоминающее устройство — это такая память, которая позволяет с очень большой скоростью записывать и считывать информацию, подготовленную для АЛУ и УУ. Скорость работы АЛУ в настоящее время очень велика, поэтому естественным является желание подготовить и разместить в ОЗУ как можно больше информации и максимально загрузить АЛУ и УУ. Следовательно, быстродействие и объем являются главными характеристиками ОЗУ, по которым его образно сравнивают с запоминающей средой нашего мозга в отличие от таких запоминающих устройств, как книги, магнитофоны, записные книжки и т. д. Однако объем ОЗУ ограничен, так же, как и объем памяти человека. Разработчики запоминающих устройств (ЗУ) хорошо знакомы с противоречием между объемом и быстродействием ЗУ. Несмотря на то, что объем ОЗУ современных ЭВМ измеряется довольно большим числом, всегда находятся такие задачи, для решения которых этого объема не хватает. Как, в каких единицах измерять объем памяти вычислительной системы? Для подготовки ответа на этот вопрос необходимо знать, что вся информация попадает в ЭВМ в закодированном



Рис. 5. ЭВМ МЕРА-125:

1—АЛУ; 2—ОЗУ; 3—УУ; 4—ЗУ на гибких магнитных дисках; 5—устройство ввода-вывода на перфолентах; 6—ЗУ на жестких магнитных дисках

виде. Все преобразования, пересылка информации осуществляются над числовыми кодами. Очевидно, что любую информацию можно представить в виде наборов символов — числовых, буквенных и специальных ($=$, $>$, $<$ и т. д.). Каждому символу ставится в соответствие некоторое число, его код, причем код выбирают таким образом, чтобы по закодированной строке всегда было возможно однозначно восстановить исходную информацию¹. Для запоминания закодированной информации в ЗУ отводятся определенные участки памяти, которые называются

¹ Для кодирования информации в ЭВМ, как правило, используют двоичные коды, в которых каждое число представляется в виде набора цифр 0 и 1. Как пример приведем двоичное представление чисел от 0 до 10: 0000, 0001, 0010, 0011, 0100, 0101, 0110, 0111, 1001, 1010. Подробнее о двоичном представлении чисел см. § 4.2.

ячейками. Все ячейки одинаковы и состоят из таких устройств, называемых разрядами, которые могут хранить одно из чисел 0 или 1. Число разрядов во всех ячейках одинаково.

Длиной ячейки или машинного слова называется число разрядов в ячейке. Количество информации, которое можно поместить в один разряд, называется битом. Число битов информации, которое можно записать в ЗУ, и будет составлять его объем.

Далее будем считать, что длина каждой ячейки кратна 8 и каждые 8 разрядов объединены в 1 байт. Это дает возможность выбрать 1 байт за единицу объема ЗУ и измерять объем памяти в байтах, килобайтах (К байт), мегабайтах (М байт), гигабайтах (Г байт) и т. д., причем

1 байт = 8 бит;

1К байт = 2^{10} байт;

1М байт = 2^{20} байт;

1Г байт = 2^{30} байт.

Объемы оперативной памяти современных ЭВМ зависят от классов машин. Имеются микроЭВМ с объемом ОЗУ 16К байт, а также ЭВМ, объем ОЗУ которых достигает 2М байт и более.

Устройство управления является одним из важнейших узлов ЦП. Получая информацию из ОЗУ и АЛУ, оно организует работу ЦП, определяя, какую необходимо выполнить операцию и над какими данными, куда поместить результат и что делать на следующем шаге.

Таким образом, УУ и АЛУ, получая из ОЗУ подготовленную к обработке информацию, преобразует ее и передает опять в ОЗУ. Каким бы большим ни был объем ОЗУ, скопление входной и вновь полученной информации быстро приводит к переполнению памяти. Естественно, возникает вопрос: может быть, управление сбором, подготовкой, хранением и выдачей информации поручить другим устройствам, освободив от этой работы ЦП? В самых первых ЭВМ все эти функции выполнял ЦП, что было препятствием на пути увеличения скорости работы машины. Скорость работы ЦП в тысячи раз превышала скорость работы устройств ввода, вывода данных и скорость обмена данными между различными устройствами вычислительной системы. Поэтому конструкторы оставили ЦП только те функции, о которых мы говорили выше, а сбор, подготовку и выдачу информации и управление этими процессами возложили на специальные блоки — *периферийные устройства (ПУ)*. Периферийные устройства через специализированные вычислительные машины — каналы — связаны с ЦП (см. рис. 3). Они накапливают, хранят и выдают поступающую информацию, не загружая этой работой ЦП. Обмен информацией осуществляется только через ОЗУ.

Периферийные устройства подключаются к каналам через

специальные блоки — контроллеры, которые принимают команды из канала, расшифровывают их и запускают соответствующие ПУ.

К периферийным устройствам относятся в основном внешние запоминающие устройства (ВЗУ) и устройства ввода-вывода информации.

Внешние запоминающие устройства (иногда говорят «внешняя память») служат для накопления и хранения информации. Существуют три класса ВЗУ, отличающиеся видами накопителей: на бумажных носителях, на магнитных носителях и на оптических дисках.

В накопителях на бумажных носителях (перфокартах или перфолентах) закодированная информация представляется в виде совокупности пробитых и непробитых отверстий. Специальное считывающее устройство определяет, в какой позиции пробиты отверстия, и выдает соответствующий сигнал, преобразует последовательность полученных сигналов в цифровые коды, понятные ЭВМ, и передает их в ОЗУ. Бумажные носители обладают многими недостатками: трудно исправить ошибки, например заклеивать неправильно пробитое отверстие; в зависимости от влажности может нарушаться гладкость бумаги и плотность ее прилегания к считывающей пластинке — пойдет неверная информация; а как легко рвется бумага! Перфокарты в настоящее время практически не используются; перфоленты применяются, как правило, для хранения контрольных тестов проверки правильности работы устройств ЭВМ.

Массовое распространение получили накопители на магнитных носителях — магнитных лентах (МЛ), магнитных дисках (МД), гибких магнитных дисках (ГМД), магнитных барабанах и т. д. Принцип работы накопителей на магнитных носителях очень похож на принцип работы обыкновенного магнитофона. Информация записывается на магнитный носитель (ленту, диск, барабан), который движется вдоль головок считывания-записи. Если это магнитная лента, то она протягивается вдоль неподвижных головок считывания-записи, передавая или записывая обрабатываемую информацию. Накопители на магнитных лентах (НМЛ) являются устройствами последовательного доступа. Такое название они получили, например, в силу того, что при необходимости прочитать информацию, находящуюся в конце ленты, надо последовательно просмотреть все предыдущие записи. Время, затраченное на просмотр записей и перемотку пленки (носителя), называется временем доступа или временем ожидания. Такие накопители относятся к медленным устройствам памяти, время доступа измеряется минутами. На МЛ длиной 720 м шириной 13 мм при плотности записи 640 бит/см можно записать около 5М байт информации (рис. 6).

Накопители на магнитных дисках (НМД) объединили принцип работы магнитофона и проигрывателя грампластинок. Сам маг-

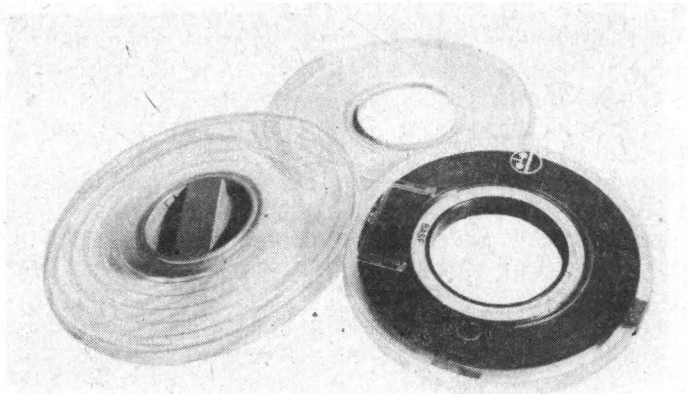


Рис. 6. Бобина магнитной ленты с коробкой для хранения

нитный диск внешне похож на грампластинку, над которой движется блок головок считывания-записи. Информация на диске записывается на концентрических дорожках. Блок считывания-записи находит нужную дорожку, зависает над ней и обменивается информацией. Скорость вращения диска довольно высокая, поэтому время доступа на НМД измеряется сотыми долями секунды. Эти накопители являются устройствами произвольного доступа, так как головки можно установить сразу над нужной дорожкой, не просматривая все остальные.

Широкое распространение получили накопители на гибких магнитных дисках — дискетах (НГМД). Дискета — это запеча-

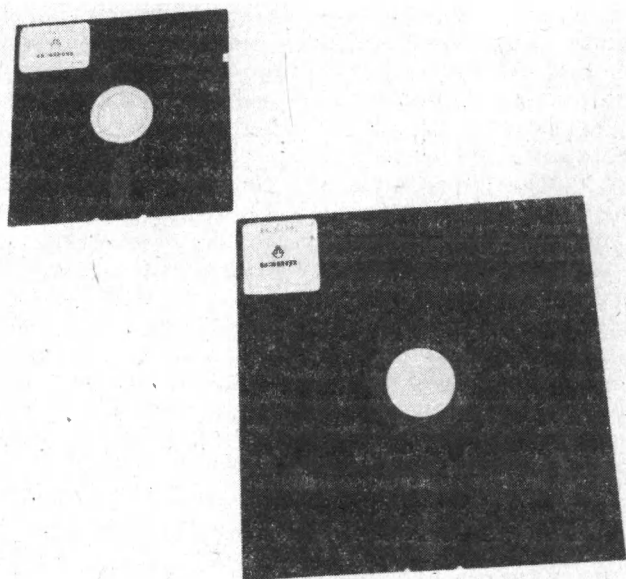


Рис. 7. Гибкие магнитные диски размером 133 и 203 мм в диаметре



Рис. 8. Накопители на гибких магнитных дисках:

I — «НГМД — Электроника 6022А» (диаметр дисков 133 мм); II — «НГМД — Электроника 7017»

танная в специальный конверт пленка с магнитным покрытием (рис. 7). Принцип работы НМД и НГМД одинаков.

Объем информации, которую можно записать на дискету, зависит от устройства памяти, подключенного к ЭВМ (рис. 8). Существуют различные устройства считывания-записи, отличающиеся своими возможностями. Объем памяти НГМД диаметром 203 мм составляет 500К байт...1,6М байт (1М байт информации соответствуют примерно 500 страницам машинописного текста). Гибкие дискеты легко меняются в процессе работы, и пользователь ЭВМ может иметь собственную библиотеку дискет с информацией.

В последние годы стали появляться накопители на оптических дисках (НОД), запись и считывание информации на которых производится лучом лазера. В отличие от магнитных носителей запись на оптическом диске производится только один раз, а считывание — бесчетное число раз. Объем информации, который можно записать на оптический диск, в тысячи раз больше объема информации, размещенного на магнитном диске того же диаметра, и измеряется гигабайтами. Выше и скорость считывания. Уже созданы магнитно-оптические ЗУ, которые объединяют

достоинства магнитных (многократность записи на один носитель) и оптических носителей.

Широко используется другой тип ЗУ на жестком магнитном диске — так называемые винчестерские диски¹. Объем памяти такого диска изменяется в диапазоне 5...300М байт в зависимости от модификации. Жесткий диск типа «Винчестер» вместе с головкой считывания-записи, упакован в герметичный корпус. Герметичность конструкции предохраняет от попадания частичек пыли и других загрязняющих частиц из окружающей среды в промежуток между диском и головками. Это позволило сократить расстояние между ними, повысить плотность записи информации и увеличить более чем в 10 раз скорость вращения диска по сравнению с гибким магнитным диском. Винчестерский диск относится к фиксированным носителям информации — с. 10 замена производится специалистами, а не пользователями. Примерные характеристики винчестерского диска:

Объем памяти, М байт	5...300
Скорость передачи, К бит/с	5000
Размеры, мм	200×200×120

Вторую группу ПУ составляют *устройства ввода-вывода*. Их основная задача — организовать диалог пользователя с ЭВМ.

Ввести информацию в машину можно различными способами, например взять дискету с записанными данными, вставить в устройство считывания-записи и загрузить их в ЦП (аналогичная процедура для ввода информации с магнитной ленты, магнитного диска, перфоленты, только для каждого носителя свое устройство считывания-записи). Следовательно, внешние запоминающие устройства НМЛ, НМД НГМД и т. д. тоже можно отнести к устройствам ввода-вывода информации.

Сейчас все ЭВМ оснащены дисплеями с клавиатурой, напоминающей обычную пишущую машинку (рис. 9). Клавиатура используется для ввода информации, которая изображается на телевизионном экране дисплея. Это позволяет не только визуальное контролировать ввод данных, но и исправлять допущенные ошибки. Для хранения изображенной на экране информации, как правило, используется специальная «дисплейная» память. На дисплей выводятся результаты обработки данных, информация об этапах ее прохождения, диагностика работы отдельных блоков ЭВМ и т. д. Широкое распространение получили дисплеи

¹ Изначально такие накопители рассчитывались на 2 диска по 30М байт, составляющих единый блок: результирующая емкость получаемого НМД обозначалась цифрами 30/30, подобно калибру старинного охотничьего ружья «Винчестер». Отсюда и возникло название «винчестерский диск», или «Винчестер». — *Прим. ред.*



Рис. 9. Дисплей с клавиатурой МЕРА-7848 производства ПНР

с цветным изображением. Недавно стали выпускаться дисплеи с сенсорным экраном, что позволяет во многих случаях не пользоваться клавиатурой при вводе информации.

Общение пользователя с ЭВМ через дисплейные устройства бывает недостаточным, когда результаты обработки данных важно иметь в виде «твердой копии», т. е. документа на бумаге, который может храниться некоторое время. Отпечатанные там данные можно анализировать не только около дисплея, но и в другой комнате, а иногда дома после чая. Такую копию получают печатающими устройствами следующих типов: алфавитно-цифровым печатающим устройством (АЦПУ), принтером, графопостроителем (плотером).

Алфавитно-цифровое ПУ напоминает электрическую пишущую машинку с большими возможностями вывода информации (рис. 10). Печать текстов, содержащих символы русского, латинского алфавита, цифровую информацию и некоторые специальные символы, осуществляется с достаточно большой скоростью и обеспечивает качественное изображение. Работают АЦПУ в основном на специальной бумаге с перфорацией.

Принтер — это, как правило, точно-матричное печатающее устройство ударного типа (рис. 11). Набор специальных игл в определенной последовательности наносит через красящую ленту

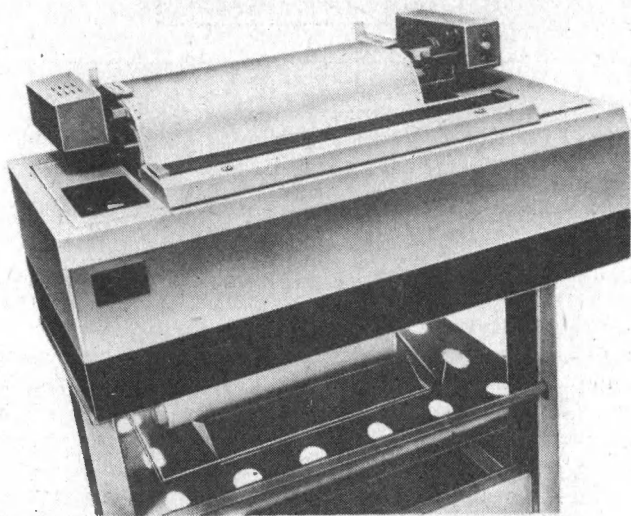


Рис. 10. Алфавитно-цифровое печатающее устройство

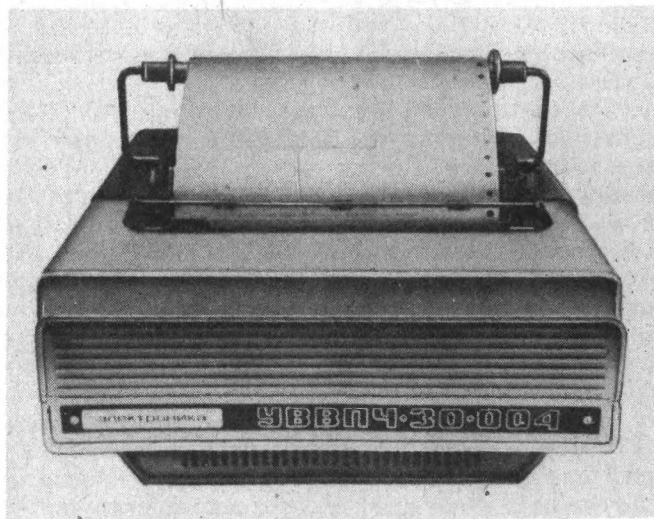


Рис. 11. Принтер с печатающим устройством игольчатого типа «УВВП4-30-004»

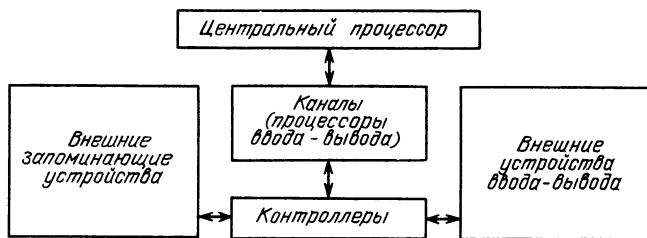


Рис. 12. Архитектура аппаратных средств вычислительной системы

удары по бумаге, следы от ударов — точки — формируют нужное изображение (буквы, цифры, рисунки и т. д.). Плотность печати достигает восьми точек на 1 мм. Это позволяет получать хорошее качество изображения (близкое к топографическому) информации, как графической, так и текстовой. Выпускаются принтеры, печатающие цветные изображения, например струйные принтеры, которые, разбрызгивая специальные чернила, рисуют тексты, чертежи, формулы и т. д.

Вывод графической информации является достаточно медленным, особенно по сравнению с выдачей алфавитно-цифровых данных, что является недостатком всех таких устройств. Самыми большими возможностями обладают лазерные печатающие устройства, которые позволяют выводить информацию с высо-

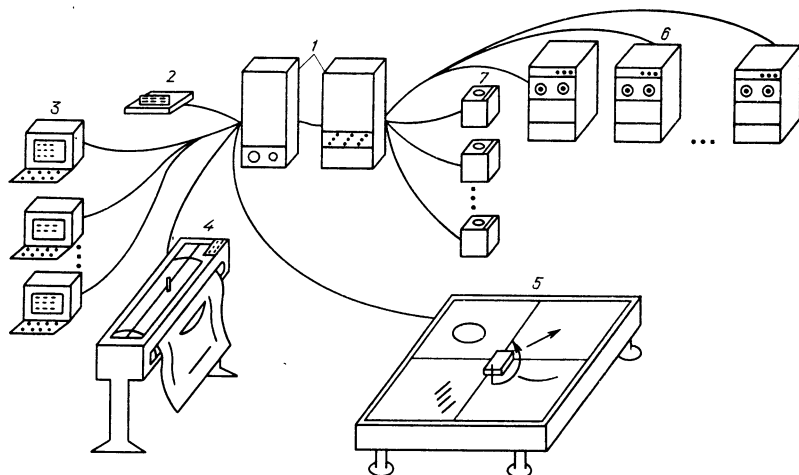


Рис. 13. Аппаратные средства ЭВМ:

1—главная стойка, содержащая ЦП, пульт управления и др.; 2—5—внешние устройства ввода-вывода (2—устройство вывода на печать (принтер); 3—устройство ввода-вывода на электронно-лучевой трубке (дисплей); 4—устройство вывода графической информации на бумагу (плотер); 5—графопостроитель); 6, 7—внешние ЗУ (6—НМД, 7—НМД)

кой скоростью. При этом качество ее представления не только сравнимо с типографским и фотографическим, но зачастую и превосходит их. В принтерах может использоваться как специальная перфорированная, так и обычная писчая бумага.

Графопостроители (плотеры) служат для выдачи с ЭВМ информации в виде проектно-конструкторских чертежей на больших листах бумаги. При рисовании таких чертежей графопостроители используют восемь цветных фломастеров или карандашей, которые, в случае необходимости, меняются автоматически. Качество изображения любых сложных геометрических фигур устойчивое и очень высокое. Производительность труда при подготовке чертежей с помощью ЭВМ увеличивается в тысячи раз по сравнению с традиционным ручным способом.

Заканчивая описание аппаратных средств ЭВМ, хочется сказать, что построенный нами портрет (рис. 12, 13) не является полным. Остались в стороне устройства сопряжения с пользователями и другими ЭВМ, удаленными на большие расстояния, устройства энергоснабжения, устройство охлаждения и др., но это сделано специально. Желющие подробно познакомиться с аппаратной частью вычислительных систем могут это сделать, обратившись к литературе по архитектуре и конструированию ЭВМ, а мы перейдем к портрету программных средств.

2.2. ПРОГРАММНЫЕ СРЕДСТВА (ПРОГРАММНОЕ ОБЕСПЕЧЕНИЕ)

Из портрета аппаратных средств ЭВМ видно, что простое подключение друг к другу таких сложных устройств, как ЦП, НМЛ, диски дисплеи и т. д., не обеспечит согласованной работы всей вычислительной системы. Необходима программа их совместных действий, определяющая порядок включения, как и какие операции надо выполнять каждому устройству в соответствующий момент времени и т. д.

Для управления аппаратными устройствами и создания удобств пользователю при общении с вычислительной машиной создается программное (математическое) обеспечение (ПО), которое освобождает от необходимости знать и учитывать специфические свойства каждого устройства, дает возможность общаться с ЭВМ на языке, близком к профессиональному языку пользователя, и не заботиться при решении своей задачи о выделении и распределении памяти в ОЗУ и ВЗУ под программу, данные, полученные результаты и т. д.

Так что такое программа для ЭВМ?

Программа для ЭВМ — это набор команд, управляющий устройствами вычислительной системы для выполнения определенной последовательности машинных операций при обработке заданного набора данных. Процесс создания, отладки, тестиро-

вания этих программ называется *программированием*. При подготовке программы пользователь, опираясь на свою задачу и правила работы с командами машины, создает набор командных или информационных данных для ЭВМ, который называют файлом. Файлу присваивают имя и тип.

Как правило, при присвоении имени и типа стараются отразить содержащуюся в файле информацию. Например, файл, содержащий список вопросов для подготовки к экзамену по информатике, можно назвать «СПИН.ТКТ». Имя «СПИН» построено по первым буквам слов «СПИСОК» и «ИНФОРМАТИКА», а написанное через точку расширение имени — тип «ТКТ» — говорит, что информация текстовая.

Различают два вида программирования; системное и прикладное.

Системное программирование. Системные программы разделяются на три класса: управляющие программы; программы распределения ресурсов; системы программирования. Что делают системные программы?

Управляющие программы:

помогают пользователю преодолеть трудности управления аппаратными средствами вычислительной системы;

освобождают от необходимости знать свойства и особенности каждого технического устройства;

контролируют работу электронных и механических устройств.

Программы распределения ресурсов:

освобождают пользователя от необходимости при решении своей задачи заботиться о выделении и распределении памяти в ОЗУ и ВЗУ;

предоставляют одновременно нескольким пользователям возможность общаться с ЭВМ;

защищают программы пользователя от умышленного или неумышленного постороннего воздействия.

Системы программирования:

предоставляют пользователю возможность создавать программы на языках программирования высокого уровня, т. е. на таких языках, которые не зависят от конкретной ЭВМ и ориентированы на свой класс задач;

переводят (транслируют) программы с языков программирования на язык конкретной машины. Такие программы называют компиляторами или интерпретаторами.

Комплекс программ, осуществляющий функции управления, распределения ресурсов и поддерживающей системы программирования, называется операционной системой (ОС)¹.

¹ Иногда под ОС понимают комплекс программ, осуществляющий только функции управления и распределения ресурсов, а системы программирования выделяют в отдельный класс.

Операционная система, таким образом, организует работу всех аппаратных средств, осуществляет общение между ЭВМ и пользователем.

Различают следующие режимы использования ЭВМ, которые организуют ОС: однопользовательский (монопольный); пакетный однопрограммный и мультипрограммный; мультидоступ; режим реального времени.

Однопользовательский (монопольный) режим был характерен для работы на самых первых ЭВМ, а сейчас используется на персональных компьютерах. В этом режиме пользователь один работает с вычислительной системой и управляет ею с клавиатуры пульта ЭВМ. В этом случае задача ОС — максимально упростить общение человека с машиной. Такое общение организуется как диалог, в котором ЭВМ предлагает пользователю выбрать для работы или язык программирования, или необходимую программу, входящую в ОС. Формы диалога могут быть разные. Например, на экране дисплея пользователю предлагается меню — перечень услуг, которые может выполнить ЭВМ. Выбирая нужное из этого меню и задавая необходимые данные, можно приступить к выполнению работы.

Пакетный однопрограммный и мультипрограммный режим, или просто пакетный режим, лишает пользователя прямого общения с ЭВМ. Специалист составляет текст программы и готовит необходимую информацию, потом передает их для переноса на перфоленты, магнитные ленты или на гибкие магнитные диски. Тексты на носителях передаются оператору ЭВМ, который собирает в один пакет программы, полученные от разных пользователей, и пропускает пакет через машину. ЭВМ, закончив обработку одной программы из пакета, прочитывает следующую по порядку задачу и приступает к ее обработке. Так идет обработка потока заданий, полученных от разных пользователей.

Мультипрограммный режим пакетной обработки в основном используется на больших ЭВМ. В этом режиме в оперативную память загружаются несколько задач из пакета, которые, можно сказать, «одновременно» решаются. В действительности, в некоторый момент машина прерывает обработку первой задачи и переходит к обработке второй, прерывая обработку второй задачи, переходит к третьей и т. д. Так почти одновременно и обрабатывается пакет нескольких программ.

Откуда и как возникают прерывания? Например, пусть в программе стоит указание: «После вычисления по первым трем формулам результаты вывести на дисплей». Процессор, проведя вычисления по первым трем формулам задачи, должен передать полученные результаты в промежуточную память и далее на экран дисплея. Арифметико-логическое устройство прерывает обработку, и пока идет передача данных, АЛУ загружают другой программой, которая выполняется до ближайшего прерывания.

Прерывания работы ЦП происходят не только из-за передачи данных другим устройствам, но и по другим причинам. Мы не будем их описывать, важно понять, что они возникают при выполнении почти каждой программы. Для реализации пакетного мультипрограммного режима ОС должна обеспечить:

возможность одновременной (параллельной) работы внешних устройств ЭВМ;

защиту программ и результатов обработки от взаимного влияния, перемешивания.

Режим мультидоступа позволяет работать на одной ЭВМ одновременно с нескольких терминалов (*терминалом* называют удаленное на некоторое расстояние от машины устройство ввода-вывода информации. В настоящее время в качестве терминалов чаще всего используют дисплеи, персональные компьютеры и др.). В этом режиме аппаратные средства одной вычислительной машины распределяются между несколькими одновременно работающими за своими терминалами пользователями. У каждого пользователя создается впечатление, что он один управляет работой ЭВМ в монопольном режиме.

При работе ЭВМ в этом режиме ОС выполняет роль диспетчерского пункта, куда поступают заказы на обслуживание, и важно выполнить эти заказы с минимальной потерей времени на простаивание в очереди на обработку. Чем меньше очередь на выполнение программы, тем меньше уходит времени на ожидание ответа ЭВМ в диалоге с пользователем, работающим за терминалом. Так как режим мультидоступа, как правило, реализуется на больших ЭВМ, то задание и программы, поступающие с терминалов, обрабатываются в мультипрограммном режиме. Иногда объединяют мультидоступ и пакетную обработку, «одновременно» обслуживая терминалы и обрабатывая пакеты программ.

В режиме реального времени ОС распределяют все заявки на выполнение программ по приоритетам в зависимости от времени их исполнения и сроков их обязательного завершения. Понятно, что в таком режиме для осуществления правильного управления процессом обработки ОС должна своевременно получать информацию о прохождении программ. Эта информация предоставляется системой прерываний, суть которой заключается в следующем. В систему в любой момент времени от любой программы может поступить команда «Внимание!!»; тогда ОС, работающая в режиме реального времени, должна прервать выполнение текущей программы и переключится на программу, подающую сигнал. Для предотвращения хаоса создается система приоритетов, позволяющая правильно организовать систему прерываний и очередь из сигнализирующих программ.

Режимы реального времени и мультипрограммирования имеют много общего, так как основная проблема у них одна — как быстро

прервать выполнение текущей программы и какое принять решение: перейти к выполнению другой или продолжить обработку исходной.

Немного о системах программирования. Мы уже говорили, что ЭВМ воспринимает информацию, закодированную двоичными числами и преобразованную в электрические сигналы. Программа, написанная на языке команд устройства вычислительной системы, перечисления исходных данных и адресов памяти ЭВМ (такой язык называется языком машинных кодов), является большой, сложной и при ее написании можно сделать много ошибок. Писать программы на языке машинных кодов довольно сложно, поэтому в начале 50-х годов была создана специальная система обозначений — команд, которую называли языком ассемблера.

Команды ассемблера, как правило, обозначались буквами соответствующего слова — имени команды машинного кода (например, команда «add» — от английского слова add — складывать). Для перевода программы, написанной на ассемблере, на язык машины составлялись таблицы соответствия команд ассемблера и их машинных кодов. Вначале такой перевод осуществлялся вручную, потом этот процесс автоматизировали и поручили выполнять самой ЭВМ. В настоящее время на языке ассемблера пишут программы, когда нужно эффективно использовать все машинные ресурсы и достичь высокой скорости ее исполнения. Но основным недостатком ассемблера является то, что программа составляется в терминах «что надо делать машине», а не в терминах, близких к языку задач пользователя. Это создает барьер между ЭВМ и массовым пользователем.

В конце 50-х годов появились новые средства общения с вычислительной машиной — языки высокого уровня. Команды языка высокого уровня определяют сразу несколько машинных команд, что сокращает тексты программ, снижает трудозатраты по их написанию, а сам текст стал лучше отражать свойства решаемой задачи. Введение в язык понятий, используемых при решении конкретных задач, позволяет уменьшить вероятность появления ошибок, сделать программы не зависящими от конкретной ЭВМ.

Каждый язык высокого уровня определяется системой записи и набором правил, определяющим синтаксис. Грубо говоря — это набор слов (словарь) и правил составления предложений. Но написав программу на языке высокого уровня, надо перевести ее на машинный язык, т. е. провести трансляцию. Трансляцию языка осуществляют программы-переводчики: компиляторы и интерпретаторы.

Компиляторы обращаются к уже написанной программе на языке высокого уровня, проверяют, есть ли орфографические и синтаксические ошибки, и переводят ее в машинные

коды. В итоге создается программа на машинном языке, которая в дальнейшем может выполняться без участия компилятора.

Интерпретатор переводит предложения в машинные коды по мере их написания. Можно сделать такое сравнение — компилятор подобен переводчику книги, а интерпретатор похож на переводчика устной речи. Первый берет книгу на одном и создает новую книгу на другом языке. А переводчик устной речи переводит предложения сразу после их произношения.

Из языков высокого уровня, получивших наиболее широкое распространение, назовем следующие:

Алгол — ALGOL (Algorithmic Language — алгоритмический язык). Создан в 50-х годах, использовался при программировании вычислительных задач в математике, физике и т. д.;

Фортран — FORTRAN (Formula Translation — трансляция формул). Создан в середине 50-х годов, ориентирован на создание программ проведения расчетов при решении задач, использующих математическое моделирование;

Кобол — COBOL (Common Business Oriented Language — язык, ориентированный на обработку экономической информации). Создан в 1960 г., используется для обработки табличной и другой информации;

Бейсик — BASIC (Beginner's All-purpose Symbolic Instruction Code — язык символьных конструкций для начинающих). Создан в 1965 г., используется для написания программ решения прикладных задач инженерами и популярен среди пользователей, не являющихся профессиональными программистами. Особенно широкое распространение получил с появлением персональных ЭВМ;

ПЛ-1 — PL-1 (Programs Language — язык программирования). Создан в 70-х годах, используется для решения большого класса научных задач и обработки экономической информации;

ЛИСП — LISP (List Processing — обработка списков). Создан в 60-х годах, в настоящее время широко используется при создании систем машинного интеллекта.

Невозможно выделить из существующих языков программирования высокого уровня наилучший для решения любых задач; каждый язык имеет свою область применения, свои достоинства и недостатки. При выборе языка для создания ваших программ необходимо: 1) выяснить, какие системы программирования есть на данной ЭВМ; 2) оценить возможности каждого языка по отношению к решаемой задаче; 3) выучить выбранный язык, освоить методы и приемы составления с его помощью программ.

Здесь упомянуты далеко не все языки: не вспомнили о языках Паскаль, Си, Ада и др., да и речь шла только об универсальных языках программирования. Существует еще много языков, предназначенных для решения какого-то узкого специального класса

задач. Их называют проблемно-ориентированными языками программирования, но и они останутся вне нашего внимания.

Прикладное программирование. Мы уже поняли, что системные программы образуют первый управленческий этаж над аппаратными средствами ЭВМ, откуда организуют их взаимодействие. Но к решению практических задач, например расчета сопротивления крыла самолета, они имеют косвенное отношение. На втором этаже управления работой технических средств расположены прикладные программы, которые, с одной стороны, связаны с конкретными проблемами пользователя, а с другой — используя все возможности, предоставляемые ОС, управляют работой устройств ЭВМ. Прикладные программы, как правило, пишутся на языках программирования высокого уровня. Их можно сравнить с действиями командующего армией, который принимает решения и отдает приказы командирам дивизий, полков, но не командирам батальонов и рот. На основании приказа командарма, командиры дивизий и полков отдают свои приказы батальонам, ротам, а их командиры принимают решения и командуют действиями отдельных подразделений или солдат. Так и в ЭВМ, команды прикладной программы воспринимаются ОС, которая переводит их в команды отдельным устройствам и определяет порядок их включений.

Но в основе всякой прикладной программы лежит задача перевода конкретной проблемы пользователя на язык, понятный ЭВМ. Прикладное программирование — это процесс создания машинных программ, ориентированных на решение конкретной практической задачи, например программы расчета траектории полета космической ракеты, программы определения частоты встречаемости слов в литературном произведении и др.

Легко понять, что уровень «интеллекта» всякой ЭВМ определяется количеством прикладных программ из разных областей знаний: чем их больше, тем выше этот уровень.

Написание прикладных программ — процесс сложный и долгий; недаром отношение стоимости технических средств ЭВМ к стоимости программного обеспечения оценивается в настоящее время величиной порядка 3:17 (рис. 14).

Разработчики первых прикладных программ заметили, что почти во всех математических моделях, исследуемых с помощью ЭВМ, встречаются общие процессы, например вычисление элементарных функций x^h , $\lg x$, $\sin x$ и т. д. Можно было переписывать такие запрограммированные блоки из одной программы в другую, но это требовало затрат времени и приводило к опискам при ручном копировании текстов. Тогда возникло предложение — создать и записать в память ЭВМ стандартные программы элементарных функций, а пользователю разрешить в соответствующем месте своей программы сделать указание, например: «Вычислить $\sin x$ при $x=0,21$, результату присвоить

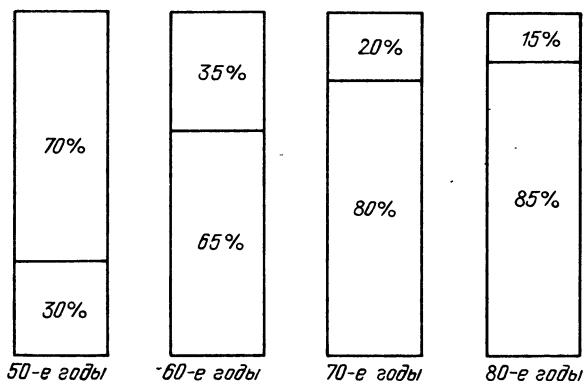


Рис. 14. Изменение относительной стоимости программных и аппаратных средств (вверху — данные для аппаратуры; внизу — для программного обеспечения)

имя y ». ЭВМ воспринимала «sin» как имя стандартной программы, находила в своей памяти ее текст и автоматически включала его в итоговую прикладную программу, которая передавалась в вычислительную систему для исполнения,

В дальнейшем стали разрабатывать и хранить в памяти не только программы вычисления элементарных функций, но и более сложные математические построения. Так появилось понятие подпрограммы. Программисты стали разбивать большие программы на независимые части — модули, создавали и отлаживали эти программные модули, а в тексте основной программы указывали, когда и к какой подпрограмме (модулю) надо обратиться. Такой метод позволил сократить тексты программ, проще стало проверять логику выполнения операций, а в результате повысилась производительность труда программистов и качество выпускаемой ими продукции.

Прошло еще некоторое время, накопился опыт создания прикладных программ и стало очевидным, что надо искать новые пути в программировании. Анализируя математические модели процессов, исследуемых с помощью ЭВМ, и соответствующие им программы, обнаружили, что можно расширить круг стандартных подпрограмм. Оказалось, что почти в 90% моделей встречаются такие математические операции, как сложение двух векторов, скалярное произведение векторов, решение системы линейных алгебраических уравнений и т. д.

Для решения таких типичных задач были разработаны подпрограммы, из которых в ЭВМ создали библиотеку. Подпрограммы, включенные в библиотеку, хранятся в ВЗУ, а специальная диспетчерская программа «библиотекарь» по запросу программиста находит нужную и включает ее в указанное место

основной программы. Библиотека может меняться — исключая старые программы (стирают из памяти) и пополняют фонд новыми, с чем информируют пользователей. В хорошо развитых библиотеках имеются тематические разделы, в которых содержатся сотни программ, и это не удивительно. Вы сами, читатель, можете привести несколько примеров различных методов решения системы линейных алгебраических уравнений: метод исключения неизвестных Гаусса, метод Крамера и т. д. Каждый метод обладает своими достоинствами и недостатками, поэтому в хорошей библиотеке для одной типичной задачи всегда имеется несколько программ, отличающихся друг от друга своими характеристиками.

В настоящее время широкое распространение получили пакеты прикладных программ (ППП). Появление ППП отражает стремление разработчиков программ обеспечить потребности специалистов некоторой предметной области и сделать пакеты программ проблемно-ориентированными. ППП представляет собой комплекс связанных между собой программ, позволяющих решать любую задачу из данной предметной области. Существуют пакеты расчетов поведения ядерного реактора, пакеты решения экономических задач планирования производства, пакеты обучающих программ и т. д. Отличие ППП от библиотеки заключается в том, что программы библиотеки используются независимо друг от друга, а в пакете программы рассчитаны на совместное применение в различных сочетаниях. Стоимость ППП очень высокая и в некоторых случаях в 2—3 раза превышает стоимость самой ЭВМ.

В настоящее время в различных организациях разрабатывается большое количество различных ППП, спектр применения которых достаточно широк и может представлять интерес для сотрудников не только одной организации, но и других, проводящих исследования по смежным проблемам. Очевидно, что разработка ППП, близких по своему назначению, не всегда целесообразна: иногда выгоднее воспользоваться уже созданными и показавшими высокую эффективность.

Как это сделать? Как и где получить информацию о уже существующих программах, их характеристиках и стоимости?

Для этого в нашей стране создано Научно-производственное объединение (НПО) «Центрпрограммсистем», на которое возложены функции головной организации по разработке, производству и сопровождению прикладных программных средств вычислительной техники, предназначенных для многократного межотраслевого использования. Для информации и обеспечения организаций и других заинтересованных лиц программными средствами в НПО ведется банк программных средств (БПС). В этот банк принимаются алгоритмы и программы, созданные в различных институтах и предприятиях нашей страны. Но прежде чем эта продукция попадет в БПС, проводятся ее экспертиза и испытания на эффективность работы и соответствие сопровождающей ее

документации единым требованиям оформления и содержанию продукта.

В настоящее время в БПС НПО «Центрпрограммсистем» сосредоточены алгоритмы и программы по следующим направлениям:

- 1) организация и ведение информационной базы автоматизированных систем управления;
- 2) организация вычислительного процесса;
- 3) автоматизация программирования и проектирования автоматизированных систем управления;
- 4) реализация оптимальных методов планирования и управления;
- 5) решение задач функциональных подсистем автоматизированных систем управления как в промышленной, так и в непромышленной сферах.

Научно-производственное объединение «Центрпрограммсистем» выпускает и рассылает информационные материалы, из которых потенциальные пользователи могут узнать, какие программные средства имеются в БПС, какие возможности они представляют и как их приобрести.

Но тем не менее большое количество программных средств, особенно специально проблемно-ориентированных, имеется в организациях-разработчиках, информация о которых чаще всего становится известной через доклады и выступления на различных конференциях и симпозиумах, а также через публикации в научной печати.

Вернемся к истории ЭВМ и рассмотрим поколения развития вычислительной техники.

ГЛАВА 3



ПОКОЛЕНИЯ ЭВМ

Когда говорят о поколениях, то в первую очередь говорят об историческом портрете ЭВМ. Фотографии, вклеенные в паспорт по истечении определенного срока, показывают, как изменился во времени один и тот же человек. Точно так же поколения ЭВМ представляют серию портретов вычислительной техники на разных этапах ее развития.

Принято считать, что с 1945 г. по настоящее время ЭВМ в своем развитии прошли пять этапов или поколений: первое — 50-е годы; второе — начало 60-х годов; третье — конец 60-х годов; четвертое — 70-е годы; пятое — 80—90-е годы.

3.1. ПЕРВОЕ ПОКОЛЕНИЕ

Основными компонентами (элементной базой) первых ЭВМ были электронные лампы. Десятки тысяч ламп потребляли много электроэнергии, выделяли большое количество тепла и занимали много места. Надежность работы ламповых устройств была низкой. Быстродействие ЭВМ первого поколения (например, ЭВМ М-20) имело порядок 20 тыс. оп/с, а объем оперативной памяти составлял 4К слов по 45 бит. Но даже эти ЭВМ работали в 600 тыс. раз быстрее, чем настольные клавишные вычислительные машины.

Работа машины была организована просто: ЦП приступал к исполнению очередной команды только после окончания выполнения предыдущей. Обмен данными с ВЗУ осуществлялся медленно. Скорость работы ВЗУ (накопителей на перфолентах, перфокартах, позже на МЛ) была меньше скорости работы ЦП в тысячи раз. Оперативная память изготовлялась из блоков ферритовых сердечников.

Программы писались на языке машинных кодов, программист сам распределял ячейки памяти под программу, входные данные и полученные результаты.

Именно в этот период стали разрабатываться методы, облегчающие общение пользователей с ЭВМ: алгоритмические языки, стандартные подпрограммы, библиотеки программ с инструкциями по их применению и т. д.

3.2. ВТОРОЕ ПОКОЛЕНИЕ

Начало 60-х годов характеризуется внедрением новой элементной базы ЭВМ — полупроводников и созданных на их базе транзисторов, которые пришли на смену электронным лампам.

Использование транзисторных элементов позволило уменьшить расход электроэнергии, выделение тепла, сократить размеры отдельных устройств и всей машины. Увеличился срок безотказной работы ЭВМ. Быстродействие достигло $10^4 \dots 10^5$ оп./с, объем

оперативной памяти вырос до 64К слов. Сложнее стала архитектура ЭВМ, появились НМД и дисплеи. Расширилась система команд машины. Удалось распараллелить, совместить по времени выполнение отдельных операций, таких, как работа ЦП и устройства ввода-вывода.

Появились возможности общаться с машиной в мультиграммном режиме и режиме разделения времени. Произошел переход от написания программ на машинном языке к написанию их на алгоритмических языках. Но в то же время продолжается конфликт между медленно работающими устройствами ввода-вывода и быстрым ЦП.

3.3. ТРЕТЬЕ ПОКОЛЕНИЕ

Период 60-х годов характеризуется рождением промышленной технологии создания интегральных схем (ИС) и широким использованием ИС в электронной технике. Удалось на поверхности кремниевой пластинки размером около 1 см² создать электронную схему, содержащую несколько тысяч компонентов.

Применение ИС существенно сократило потребление электроэнергии, упростило системы охлаждения, уменьшило размеры и повысило надежность ЭВМ; ИС составили основу элементной базы вычислительных машин третьего поколения. Быстродействие машин повысилось до $10^6 \dots 10^7$ оп./с, а оперативная память в отдельных ЭВМ расширилась до нескольких мегабайт.

В машинах третьего поколения основным способом общения стал режим разделения времени. Общение с машиной организуется сразу с нескольких терминалов, широко используются дисплейные терминальные устройства. Появились операционные системы мультидоступа, стали интенсивно развиваться языки высокого уровня. С машинами третьего поколения связано еще одно важное событие — пользователь получил возможность при общении с ЭВМ пользоваться как цифровой, так и графической информацией.

3.4. ЧЕТВЕРТОЕ ПОКОЛЕНИЕ

Элементной базой ЭВМ четвертого поколения стали большие интегральные схемы (БИС) и сверхбольшие интегральные схемы (СБИС). В 70-х годах было налажено промышленное производство таких СБИС, у которых на поверхности кристалла кремния располагалось несколько десятков тысяч электронных компонентов. В результате резко сократились размеры машин, быстродействие возросло до 10^8 оп./с, объем оперативной памяти стал измеряться в мегабайтах. Совершенст-

уется периферия ЭВМ, появляются дисковые ЗУ, объем памяти которых измеряется сотнями мегабайт.

Различные ЭВМ стали объединять в многомашинные комплексы и сети. *Многомашинный комплекс*, как правило, соединяет несколько малоудаленных ЭВМ. Такое объединение позволяет повысить надежность работы, организовать обмен между машинами как обрабатывающими программами, так и хранимой информацией. Сети ЭВМ — это, как правило, крупные системы, объединяющие различные вычислительные комплексы и отдельные машины. Каждый пользователь, обращаясь к сети ЭВМ, может получить доступ к данным, находящимся в других ЭВМ или комплексах, что позволяет хранить и использовать в работе такое количество информации, которое невозможно разместить в одном месте. Связь между машинами в сетях может осуществляться через специальную кабельную связь, телефонную сеть и спутниковую связь.

Мы уже отмечали, что БИС и СБИС позволили резко сократить размеры вычислительных машин. Появилась возможность создать ЦП в виде одной БИС или СБИС. Его называли *микропроцессором*. БИС (или СБИС) удалось использовать и для создания миниатюрных устройств сопряжения с ПУ и т. д. Такие БИС (или СБИС), размещенные на одной или нескольких печатных платах, составляют сердце современного *микрокомпьютера*. Постоянное снижение стоимости ИС, а в результате — снижение цен на ЭВМ привели к резкому расширению числа пользователей современной вычислительной техники. Стали выпускать специальные проблемно-ориентированные ЭВМ, нацеленные на решение одного или нескольких конкретных типов задач. Специальные микропроцессоры управляют работой станков с числовым программным управлением; созданные на их основе бортовые ЭВМ управляют полетом самолетов и ракет и др. Микроэлектроника прочно вошла и в наш быт — привычными стали калькуляторы, электронные часы, стиральные машины с программным управлением, электронные игры и т. д.

Успехи микроэлектроники привели к созданию персональных ЭВМ (ПЭВМ). Персональной ЭВМ называют автономную недорогую микропроцессорную вычислительную систему (рис. 15). Персональные ЭВМ получили широкое распространение, и в настоящее время в мире их насчитывается около 10 млн.

С другой стороны, именно возможность создавать СБИС самого разного назначения позволила сконструировать *суперЭВМ*. В нашем языке термин «суперЭВМ» новый, он отражает новые возможности в области конструирования вычислительных машин, которые еще десять-пятнадцать лет назад казались фантастическими. СуперЭВМ — это вычислительные машины рекордной производительности. Современные суперЭВМ работают со скоростью порядка 500 млн. оп./с над 64-разрядными словами; оперативная

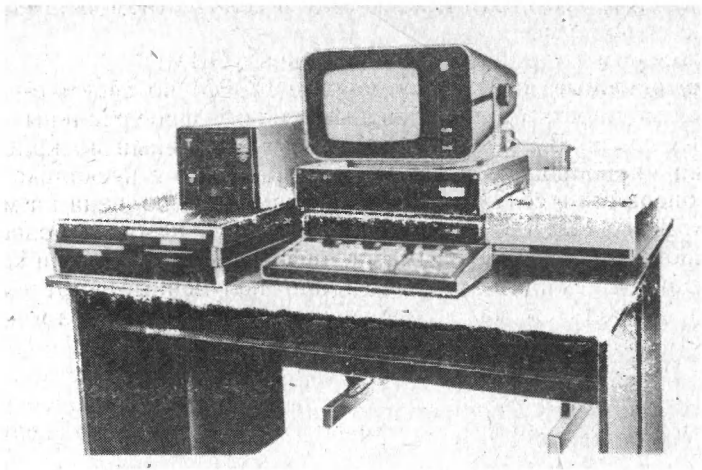


Рис. 15. Дисплейно-вычислительный комплекс ДВК-2М с НГМД и принтером

память этих машин измеряется десятками и даже сотнями мегабайт. Такие ЭВМ стоят дорого, они предназначены для решения больших задач, связанных с изучением атомного ядра, моделированием климата на нашей планете, освоением космического пространства, проектированием сложных технических изделий и т. д.

В этот же период совершенствовалось и программное обеспечение. Интенсивно развивались автоматизированные информационно-поисковые системы, базы данных, стали говорить о базах знаний.

Массовое распространение ПЭВМ привело к созданию таких программных средств (операционные системы, пакеты прикладных программ и т. д.), которые позволили общаться с ЭВМ широкому кругу пользователей: инженерам, экономистам, учителям, домохозяйкам, детям и др. Появились ПЭВМ с сенсорным экраном. Обучение работе и общение с таким компьютером происходит в виде диалога: на экране дисплея высвечивается список предложений (меню), пользователь легким прикосновением к экрану отмечает нужную ему процедуру, и ЭВМ приступает к ее исполнению. Меню может предлагаться в виде текстовых предложений или в виде изображений. Например, операция уничтожения файла изображается мусорным ведром, перевод компьютера в режим выполнения арифметических действий и вычисления элементарных функций изображается калькулятором и т. д. Для знакомства с инструкциями и описаниями команд предлагается большой набор подсказок, которые существенно облегчают общение с ЭВМ массовому неквалифицированному пользователю.

Уровень развития вычислительной техники в настоящее время стал определяться двумя показателями: возможностью создавать

и производить супер-ЭВМ; качеством и количеством выпускаемых персональных ЭВМ.

Поговорим подробнее о персональных ЭВМ.

Персональные ЭВМ. Современные ПЭВМ по своим вычислительным мощностям эквивалентны большим универсальным ЭВМ 60—70-х годов. В основе ПЭВМ лежит собранный на кристалле кремния микропроцессор производительностью в несколько сотен тысяч операций в секунду. Основная память выполнена на микросхемах; ЗУ на МЛ, ГМД расширяют основную память и позволяют организовать обмен программами, информацией с другими ПЭВМ. Как правило, данные, вводимые или полученные в результате работы, выводятся на экран дисплея, что дает возможность осуществлять визуальный контроль и упрощает общение пользователя с ПЭВМ. Если подключить специальное устройство модем (модулятор-демодулятор сигналов), то компьютер способен принимать и передавать данные на большие расстояния по телефонному каналу.

Объединение ПЭВМ в сеть позволяет использовать информацию, хранящуюся в различных системах, что особенно важно при организации учебных классов. В настоящее время ПЭВМ наиболее широко применяются в следующих областях:

- научные исследования и инженерно-конструкторские работы (научная сфера);

- управление хозяйственно-экономической деятельностью (деловая сфера);

- образование;

- бытовая сфера.

В соответствии с задачами, решаемыми в каждой из областей, в ПЭВМ используется разное программное обеспечение. Кроме того, для каждой из областей выпускают и специальные персональные компьютеры. Как пример назовем бытовой компьютер БК-0010, предназначенный для начального знакомства с вычислительной техникой и для применения в домашних условиях (записная книжка, управление приборами, компьютерные игры и т. д.).

В научной сфере ПЭВМ должны уметь накапливать, хранить, искать большие объемы различной информации (банки данных), взаимодействовать с контрольно-измерительной аппаратурой для оперативного анализа результатов экспериментов, воспринимать графическую информацию и т. д. Пользователи научной среды стараются приспособить ПЭВМ для своей узкой области работы, они нуждаются в специальных пакетах прикладных программ обработки результатов данного типа экспериментов, в специальных средствах подключения ПЭВМ к своей аппаратуре и т. д. Если таких средств нет, то пользователь из научной сферы сам разрабатывает необходимое обеспечение. Следовательно, ПЭВМ должны воспринимать такое расширение своих возможностей.

Научные исследования и инженерные расчеты связаны с проведением большого числа вычислений, поэтому необходимо обеспечить высокую скорость вычислений и эффективный обмен информацией. Так как пользователь способен сам создавать программный продукт, то ПЭВМ должна дать ему возможность редактировать на экране тексты, анализировать и диагностировать их, сообщать об обнаруженных ошибках и эффективно выполнять вновь созданные программы.

Для проектно-конструкторских работ ПЭВМ оснащают набором специализированных программ, которые позволяют резко повысить производительность труда инженеров. Часть таких программ удалось реализовать в виде ИС, что усилило проблемную ориентацию ПЭВМ и привело к образованию отдельного класса — автоматизированных рабочих мест (АРМ). Особенно широкое распространение АРМ получили при разработке новой электронной аппаратуры. По оценкам специалистов, уровень использования средств автоматизированного проектирования в электронной промышленности в ближайшие годы возрастет до 80%. Инженер, используя АРМ, может провести разработку и анализ новой идеи на всех этапах — от проверки ее на жизнеспособность до выдачи рабочих чертежей; АРМ позволяет автоматизировать такие работы, как проверка конструкторских проектных норм и правил, выполнение чертежей, ведение статистических работ и т. д.

В деловой сфере требуется, как правило, умение работать с таблицами данных, обрабатывать тексты и графики, создавать информационные базы данных и т. д. Применение ПЭВМ в этой области требует больших объемов оперативной и внешней памяти (ОЗУ — от 0,5М байт и более, ВЗУ — порядка 80М байт на винчестерских дисках). Для подготовки отчетов, докладов, диаграмм и т. д. ПЭВМ снабжают устройствами текстового и графического ввода-вывода информации, что сильно поднимает стоимость машины. Так как пользователи этой категории, как правило, не разрабатывают собственное программное обеспечение, они заказывают и приобретают его за дополнительную плату.

В сфере образования основные задачи ПЭВМ — обеспечить проведение классных занятий по различным дисциплинам, индивидуальное и профессиональное обучение. Для этого персональные ЭВМ должны быть просты и удобны в работе, оснащены легкими в общении программами ведения диалога с машиной. Предназначенные для сферы образования ПЭВМ должны давать пользователю возможность автоматизированного самообучения и предоставлять в его распоряжение библиотеку специализированных учебных программ, чтобы обеспечить интересы обучающихся различных уровней по разным дисциплинам.

В быту. ПЭВМ используются как средства организации досуга и ведения домашнего хозяйства, для них характерна

более низкая стоимость. В системе компьютерного досуга широко используются электронные игры, подключение ПЭВМ к телевизионной сети, выход в телефонную сеть (телефонный автоответчик) и т. д. Телефонный автоответчик обеспечивает передачу информации указанным адресатам и архивное хранение (запись) на ГМД поступивших важных сообщений. Кроме того, ПЭВМ применяют для домашнего обучения и его контроля, для чего в разных странах созданы контрольно-обучающие программы по элементарной математике, языку Бейсик и другим школьным дисциплинам. Комплектация ПЭВМ цветными дисплеями, позволяющими генерировать не менее 16 цветов, увеличивает степень усвоения учебных материалов и оставляет приятное впечатление от общения с электронной машиной.

Суммируя все сказанное, можно формулировать следующее определение персонального компьютера:

ПЭВМ --- это автономная микропроцессорная вычислительная система;

стоимость ее лежит в пределах доступных для индивидуальных покупателей; предусматривается продажа покупателям, которые могут и не иметь опыта работы с вычислительной техникой; вычислительная система позволяет расширять библиотеку программ для применения в различных сферах;

машина способна работать, по крайней мере, с одним языком высокого уровня типа Бейсик, Фортран и т. д.;

операционная система позволяет работать с подключенной дополнительной аппаратурой.

На этом закончим описание ЭВМ четвертого поколения и перейдем к следующему.

3.5. ПЯТОЕ ПОКОЛЕНИЕ

В 80-е годы стало ясно, что использование ЭВМ дает возможность резко повысить производительность труда, автоматизировать производственные процессы, освободить человека от рутинной работы (например, выполнения чертежей, ведения картотек и т. д.). Электронная обработка информации, производство средств вычислительной техники и электроники составили ядро новой отрасли народного хозяйства — информатики. От нее стали зависеть основные факторы, определяющие экономическое и политическое могущество государства.

С каждым годом увеличивается число специалистов, работающих в этой области. Бурному развитию информатики способствует постоянное снижение цен на электронные устройства. Ежегодно в мире общий объем памяти всех ЭВМ увеличивается в 4 раза, а стоимость ЗУ (в пересчете на 1 бит информации) снижается на 40%.

Новые, все более совершенные вычислительные системы уже

сейчас позволяют автоматизировать проектирование сложных технических изделий, подготовку их производства и процесс изготовления, т. е. создавать гибкие автоматизированные производства, различные роботы и заводы-автоматы. Дальнейшее продвижение в этой области, разработка и внедрение новых технологий, использующих последние научные достижения, невозможны без автоматизированного банка научных знаний, накопленных человечеством. Пополнение этого багажа знаний, получение из него информации должны быть организованы просто и доступны широкому пользователю.

Создаваемая в настоящее время электронно-вычислительная техника, позволяющая накапливать и получать знания, используя которые можно будет конструировать «интеллектуальные» роботы, заводы-роботы и т. д., носит название — ЭВМ пятого поколения.

Какими должны быть новые машины, что они должны уметь делать?

ЭВМ пятого поколения должны уметь:

накапливать, хранить, преобразовывать большие массивы информации и оперативно выбирать из них требуемые данные. На основе анализа выбранных данных и их преобразования в новые (знания) специальные программные комплексы (экспертные системы) должны принимать решения и предлагать их пользователю;

общаться с помощью голоса на языке пользователя, воспринимать, обрабатывать текстовую и графическую информацию;

объединять в сети ЭВМ различных классов (микро-, мини-, суперЭВМ и т. д.) для обработки информации, удаленной на значительные расстояния.

ЭВМ пятого поколения должны обладать громадными вычислительными мощностями и иметь низкую стоимость. Такой компьютер было бы правильно представлять не как одну машину, а как вычислительную систему, связывающую в единое целое суперЭВМ (ядро системы) и мощные персональные ЭВМ (периферия). Специальные ЭВМ (машины баз знаний — для обработки, хранения и анализа информации, машины логического вывода — для анализа информации и принятия решений и др.) будут составной частью ядра ЭВМ пятого поколения и определяют их интеллект. Машинный (искусственный) интеллект станет главной особенностью компьютеров пятого поколения. Общение с пользователем будет происходить на естественном языке: программно-аппаратный комплекс опознает речь, преобразует ее в промежуточный код, выяснит смысл вопроса, сгенерирует программу для машины логического вывода, работающей в связке с машинной базой знаний, и построит ответ. Далее ответ преобразуется опять в промежуточную форму и потом на естественный

язык пользователя. Так же будет проходить ввод-вывод изобразительной и графической информации.

Планируется создать компактные ВЗУ объемом в несколько гигабайт, на которые предполагается записать все знания, накопленные человечеством за всю историю.

Создание ЭВМ пятого поколения позволит, в частности:

получить запас «рабочей силы» в виде интеллектуальных роботов и сделать планирование ресурсов рабочей силы практически независимым от демографических факторов;

повысить научно-исследовательский и научно-технический потенциал страны;

автоматизировать процессы управления на всех уровнях путем создания машин, оснащенных «речью» и «слухом», способных к грамматическому и смысловому пониманию текстов;

автоматизировать сферу распределения товаров и другие непромышленные процессы.

Для решения этих грандиозных задач недостаточно только разрабатывать вычислительные системы, наделенные такими «способностями». Необходимо еще обеспечить всеобщую компьютерную грамотность. Поэтому нужны дешевые и надежные мини-ЭВМ, ПЭВМ, оснащенные различными обучающими системами. Компьютер должен стать таким же привычным, какими являются в настоящее время электронные часы.

Какие трудности надо преодолеть на пути создания новой техники? Их много, остановимся только на двух, связанных с повышением скорости работы ЭВМ.

Вспомним, какой принцип был заложен в основу работы вычислительной машины с момента ее появления (см. § 2.1):

программы и данные поступают из ВЗУ в ОЗУ;

из ОЗУ команды поступают в УУ, данные — в АЛУ;

по командам УУ данные АЛУ преобразуются в новые (происходит счет);

из АЛУ результат поступает в ОЗУ;

если он далее не используется, то УУ дает команду передать его ВЗУ и т. д.

По такому строго последовательному методу, предложенному американским ученым, профессором фон Нейманом, и сейчас работает большинство ЭВМ.

Так как мы говорим об электронной технике, то надо помнить, что сигналы передаются по проводникам со скоростью, близкой к скорости света. Следовательно, расстояние около 3 м они проходят за время порядка 10^{-8} с. Таким образом, если мы предположим, что в нашей вычислительной машине, работающей строго последовательно, сигналы от устройства к устройству пробегают в среднем расстояние порядка 3 м, то достичь быстродействия более 10^7 оп./с просто невозможно.

Из приведенного примера видно, что для увеличения быстродействия необходимо:

сократить расстояния, которые пробегают сигналы во время работы — повысить плотность размещения электронных компонентов и устройств;

отказаться от последовательного принципа работы, шире использовать методы параллельной обработки информации.

Решение первой задачи лежит на пути микроминиатюризации электроники. Предполагается, что дальнейшее уплотнение элементов микросхемы на поверхности кремниевой пластины приведет к тому, что толщина напыленного проводника достигнет 0,5...1 мкм, этого же порядка будет и расстояние между соседними проводниками. Такое увеличение плотности требует более сложных технологических процессов изготовления СБИС, разработки новых методов охлаждения микросхем и систем отвода тепла. Разработка новых микросхем невозможна без электронных систем автоматизированного проектирования (САПР), позволяющих применять различные методы моделирования, вносить в проекты необходимые изменения, анализировать полученные решения, оперативно выдавать проектную и технологическую документацию на разрабатываемые электронные устройства.

Большое значение имеет чистота выращиваемых кремниевых кристаллов и качество изготовленных из них пластин. Вкрапления других элементов, микротрещины, возникающие при нарезании пластин, и т. д. — все это приводит к неправильному функционированию микросхем, а следовательно, и всей вычислительной системы. Недаром в число экспериментов, проводимых космонавтами на орбитальной станции, входит и выращивание особо чистых кристаллов.

Перейдем ко второй задаче. Что значит отказаться от последовательного принципа? В машинах второго и третьего поколений уже использовали принципы параллельной обработки: ЦП стал работать независимо от работы устройства ввода-вывода, меньше стала зависимость от ВЗУ и т. д. Но это все касалось параллелизации работы различных частей ЭВМ. Поиски путей повышения производительности вычислительной техники, особенно при создании суперЭВМ, привели к простому, на первый взгляд, решению — создать многопроцессорную ЭВМ. Тогда, казалось бы, сразу несколько процессоров «набросятся» на одну задачу и быстро ее решат и чем больше будет процессоров, тем быстрее будут решаться задачи.

Был предложен матричный процессор, суть которого станет ясна из примера. Возьмем девять универсальных процессоров и построим матрицу 3×3 , соединив их так, как показано на рис. 16.

В такой ЭВМ единое устройство управления так командует процессорами, что все они в одно и то же время выполняют одну и ту же команду, каждый над своими данными, поступающими

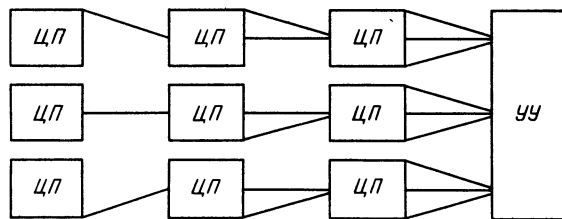


Рис. 16. Схема матричного процессора

из своего ЗУ. Между соседними процессорами предусмотрен обмен данными. Основной принцип «команда одна — данных много» отражен в названии класса таких ЭВМ (SIMD — single instruction, muly datas) и напоминает действия взвода солдат, выполняющих команды командира. Если командир отдал команду «Кругом!», то все солдаты одновременно ее выполняют.

Казалось бы, девять процессоров должны в 9 раз быстрее решать задачи, но... Вспомним метод, предложенный Евклидом для нахождения наибольшего общего делителя двух чисел, например 364 и 195. Сначала надо разделить с остатком 364 на 195:

$$364 = 195 \cdot 1 + 169,$$

далее разделить 195 на полученный остаток:

$$195 = 169 \cdot 1 + 26,$$

потом разделить 169 на новый остаток 26:

$$169 = 26 \cdot 6 + 13,$$

тогда, наконец, предыдущий остаток 26 делится на новый 13 без остатка:

$$26 = 13 \cdot 2 + 0.$$

Последний, ненулевой остаток 13 и является наибольшим общим делителем числа 195 и 364.

Легко видеть, что на каждом последующем шаге мы использовали результат предыдущих вычислений. Следовательно, решать такую задачу параллельно на нескольких процессорах не имеет смысла. Можно привести и другие примеры вычислений, которые организованы строго последовательно, хотя бы вычисление столбиком корня квадратного из положительного числа.

Эти примеры говорят только о том, что существуют задачи, которые плохо решаются на многопроцессорных матричных ЭВМ. Но есть и задачи, обладающие свойством естественного параллелизма, что позволяет эффективно решать их на машинах класса SIMD.

Значительный вклад в повышение производительности ЭВМ внесло использование принципа конвейерной обработки. Конвейерная организация производства известна давно и широко используется в автомобилестроении и др. Идею конвейера продемонстрируем на модельном примере сборочного автомобильного конвейера. В процессе сборки автомобиля выделим пять частей, или сегментов: установка двигателя; установка электрооборудования; сборка салона; установка дверей; монтаж колес.

Каждый этап сборки обеспечивает своя бригада специалистов, которая другими работами не занимается. Если все пять бригад готовят к выпуску автомобиля, отработывая последовательно один за другим пять сегментов, то они смогут приступить к сборке следующего через 5 единиц времени (для простоты мы предположим, что на отработку каждого сегмента тратится одно и то же время τ). При этом 80% рабочего времени каждая бригада простаивает в ожидании работы, а время, необходимое для выпуска N автомобилей, измеряется величиной $5\tau N$.

На конвейере первая бригада после отработки первого сегмента передает свой результат второй, а сама приступает ко второму автомобилю. На следующем этапе к работе над первым автомобилем приступает третья бригада, над вторым — вторая, а над третьем — первая. Через время 5τ первый автомобиль будет готов, второй будет собран через τ единиц времени после первого, а третий — через время τ после второго и т. д. С интервалом времени τ мы будем получать готовый автомобиль, а всего на выпуск N машин будет затрачено время $(5 + (N - 1))\tau$. Сравнивая величины $5\tau N$ и $(5 + (N - 1))\tau$, получаем, что в нашем примере при выпуске более одного автомобиля производительность конвейерной сборки выше последовательной.

Конвейерная организация в ЭВМ может быть использована для всех таких операций, которые можно разбить на подоперации (сегменты), например декодирования команд, выборки следующей команды, выполнения сложения, умножения и т. д. Даже для вычисления выражения вида $a_1b_1 + \dots + a_nb_n$ можно организовать конвейер с двумя сегментами: умножение и сложение. Функциональное устройство «умножитель» будет играть роль первой бригады, а функциональное устройство «сложитель» — роль второй бригады из рассмотренного выше примера.

Если сегменты, составляющие работу, имеют одинаковый объем, а функциональные устройства обрабатывают его за одно и то же время, то при равномерной загрузке конвейера скорость выполнения работы увеличивается в l раз, где l — число сегментов или длина конвейера. Схема конвейерного процессора показана на рис. 17. Конвейерные ЭВМ относятся к классу MISD (multy instructions single date — команд много, данное одно).

Соединение принципов конвейерной и матричной обработки привело к рождению векторно-конвейерных ЭВМ, обладающих

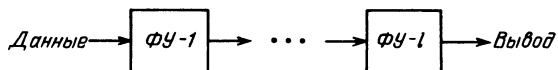
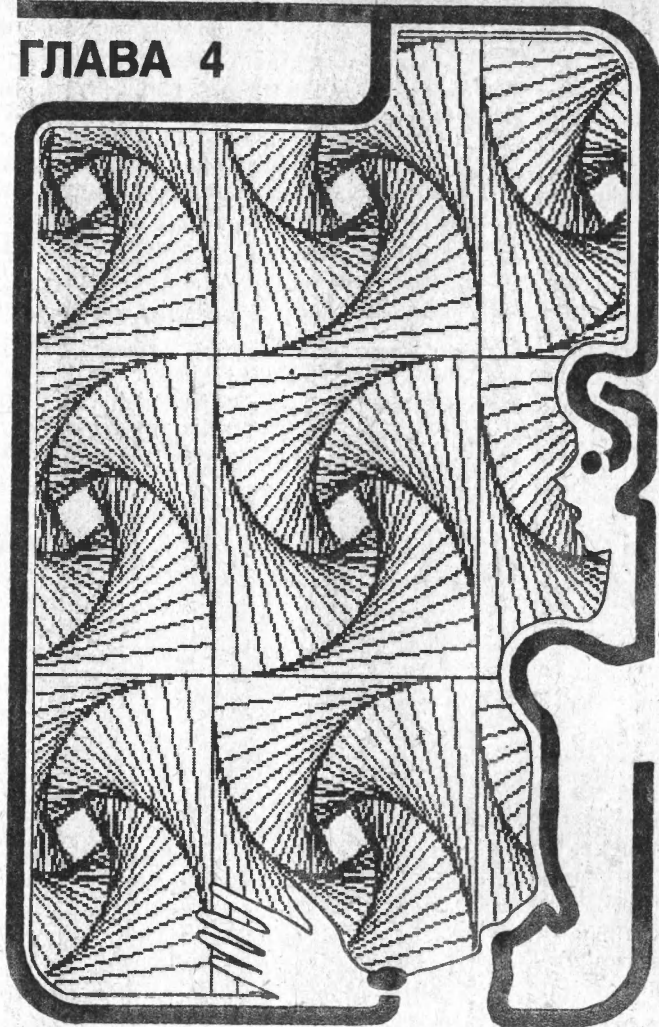


Рис. 17. Схема конвейерного процессора (ФУ— l обозначает l -тое функциональное устройство)

производительностью миллиард и более операций в секунду над 64-разрядными словами. По оценкам специалистов как в нашей стране, так и за рубежом, в течение 10—15 лет для решения научных и технических задач потребуются тысячи суперЭВМ, обладающие производительностью, которая в 100—1000 раз больше производительности существующих машин.

ГЛАВА 4



КАК
ИСПОЛЬЗУЮТСЯ
ЭВМ?

Когда речь шла о портрете ЭВМ, мы говорили, что каждый пользователь имеет свое представление о вычислительной системе, т. е. свой портрет ЭВМ. Но как составляется он, что общего между портретами, авторами которых являются специалисты разных профессий? Общим является метод построения — метод математического моделирования. Что же такое математическая модель? В третьем томе Математической энциклопедии находим следующее определение: «Математическая модель — приближенное описание какого-либо класса явлений внешнего мира, выраженное с помощью математической символики».

Человечество всегда интересовало проблемы создания благоприятных условий жизни, предсказания стихийных бедствий (землетрясений, ураганов и т. д.), поэтому нет ничего удивительного в изучении человеком различных явлений внешнего мира.

Специалисты конкретной предметной области изучают только их интересующие свойства процесса. Например, геологи изучают историю развития Земли — когда, где и какие жили животные, росли растения, как менялся климат. Все это помогает им искать залежи полезных ископаемых. Но геологи в своей профессиональной работе не изучают историю развития человеческого общества на Земле — этим занимаются историки. Уже здесь мы получаем приближенные описания исторического развития нашей планеты. А если перейти к более узкой специализации, то яснее будет наблюдаться приближенный характер получаемых результатов. Практически все исследования окружающего нас мира дают нам неполную, неточную информацию, но это не мешает совершать полеты в космос, познавать тайны атомного ядра, овладевать законами общественного развития и т. д. Необходимо, чтобы построенная приближенная модель наиболее полно отражала свойства изучаемого явления или процесса. Приближенный характер модели может проявляться по-разному. Например, точность показаний приборов, используемых в эксперименте, влияет на точность полученных результатов; расписание полетов самолетов на летний период, составленное без учета метеоусловий (которые трудно предсказать на каждый конкретный день, на полгода вперед), представляет собой приближенную модель работы Аэрофлота и т. д.

Математическая символика (а это не только формулы, графики, числа) является наиболее простым аппаратом для описания свойств окружающего мира, в первую очередь количественных. Математическое описание можно проверить на логическую непротиворечивость, оценить степень приближенности полученных результатов, подвергнуть обработке на ЭВМ и т. д.

Математические модели широко применяются в медицине, биологии, химии, физике, истории и т. д. Основное свойство

правильно построенной математической модели заключается в том, что она позволяет не только описать поведение объекта, но и предсказать, как будет он вести себя в определенных условиях.

Как строится математическая модель? Правильно построить модель — один из самых важных этапов исследования объекта. Как правило, в процессе составления модели участвуют математики и специалисты той отрасли, к которой принадлежит исследуемый объект.

На первом этапе необходимо определить основные свойства объекта и построить их математическое описание. Один и тот же объект может описываться различными моделями, отличающимися, например, степенью детализации описания его свойств. Так как математическая модель строится на некотором упрощенном описании объекта, то результаты, полученные при анализе модели, носят приближенный характер. Степень адекватности модели и объекта определяет и степень точности полученных результатов. Если объект плохо изучен, то степень адекватности определяется через сопоставление имеющихся наблюдений и полученных расчетных данных. Критерием соответствия является практика в самом широком смысле этого слова. В случае необходимости можно уточнить первоначальную математическую модель объекта.

Итак, мы построили математическую модель. Далее требуется найти методы решения построенной математической задачи и подготовить их к обработке на ЭВМ. Вспомним, что нам, как правило, необходимо получить числовое решение, а не аналитическое в виде формул, поэтому надо:

провести математическое исследование модели и, где это возможно, построить аналитические формулы решения;

построить последовательность математических операций, которую необходимо выполнить для получения окончательного численного решения. Такая последовательность операций называется *алгоритмом*.

Алгоритмы могут быть конечными и бесконечными. Если алгоритм бесконечный, то необходимо привлечь дополнительную информацию о точности вычислений, необходимой для решаемой задачи, и остановить выполнение операций алгоритма при ее достижении. Информация о точности вычисляемого результата может быть полезной и для конечных алгоритмов, для уменьшения числа выполняемых операций.

Построив алгоритмы решения, переходим к сопоставлению программ для ЭВМ. Процесс написания программ включает не только создание текстов, но и их отладку (выявление и исправление ошибок).

На следующем этапе происходит обсчет на ЭВМ исследуемой математической модели.

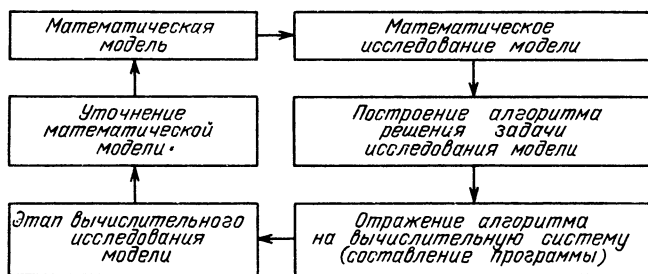


Рис. 18. Схема процесса математического моделирования

Полученные результаты анализируются и, если получена необходимая точность исследования (не обязательно вычислений), то результаты передаются для практического использования. Если необходимо уточнение модели на основании новых данных или анализа полученных расчетов, то строят новую уточненную модель и приступают к ее исследованию. Изложенный выше процесс схематично изображен на рис. 18. Эту схему можно продемонстрировать на различных примерах.

Пример 1. Рассмотрим задачу, которая в математике относится к классу транспортных.

Два завода железобетонных изделий снабжаются цементом из двух складов. В сутки первому заводу необходимо 50 т цемента, второму — 90 т. С первого склада можно вывозить в сутки 60 т, со второго — 80 т цемента. Стоимость доставки тонны цемента из первого склада на первый завод составляет 1,4 руб., на второй — 2 руб. Стоимость доставки тонны цемента со второго склада на первый завод составляет 1,2 руб., на второй — 1,6 руб. Как распределить поставки, чтобы общая сумма расходов на перевозку была минимальной?

Построение математической модели. Какие факторы могут повлиять на организацию перевозок цемента со складов на заводы? Их много: болезни водителей, выход из строя автомобилей, занятых на перевозке цемента, трудности с бензином и т. д., мы не будем их учитывать. Предположим, что на автобазе есть резерв автомобилей, хорошо поставлено медицинское обслуживание, нет трудностей с горюче-смазочным материалом и т. д. Нас интересует только, как распределять цемент между заводами. Введем обозначения:

x_1 — количество цемента, доставляемое в сутки с первого склада на первый завод;

x_2 — количество цемента, доставляемое в сутки с первого склада на второй завод;

x_3 — количество цемента, доставляемое в сутки со второго склада на первый завод;

x_4 — количество цемента, доставляемое в сутки со второго склада на второй завод.

С первого склада можно вывезти 60 т; моделью этого свойства является уравнение

$$x_1 + x_2 = 60.$$

Аналогично для второго склада получаем

$$x_3 + x_4 = 80.$$

Так как поставки должны обеспечить безостановочную работу первого завода, то объем поступлений с обоих складов должен равняться объему суточного потребления цемента:

$$x_1 + x_3 = 50.$$

Для второго завода получаем

$$x_2 + x_4 = 90.$$

Очевидно, что все сформулированные ограничения на объем перевозок должны выполняться одновременно. Следовательно, мы получили систему линейных уравнений:

$$\begin{aligned} x_1 + x_2 &= 60, \\ x_3 + x_4 &= 80, \\ x_1 + x_3 &= 50, \\ x_2 + x_4 &= 90. \end{aligned} \quad (1)$$

Как оценить затраты на перевозки? Зная стоимость одной тонны и объем перевозок, надо перемножить эти величины для каждого завода и соответствующего склада и сложить их:

$$1,4x_1 + 2x_2 + 1,2x_3 + 1,6x_4.$$

Вот как теперь выглядит построенная математическая модель задачи: найти, при каких x_1, x_2, x_3, x_4 линейная функция $f = 1,4x_1 + 2x_2 + 1,2x_3 + 1,6x_4$ принимает наименьшее неотрицательное значение, если на переменные x_1, x_2, x_3, x_4 наложены следующие ограничения: $x_1 + x_2 = 60, x_3 + x_4 = 80, x_1 + x_3 = 50, x_2 + x_4 = 90$.

Исследование модели. Упростим систему ограничений, выразив переменные x_2, x_3, x_4 через x_1 . Получим систему линейных уравнений, эквивалентную (1):

$$\begin{aligned} x_2 &= 60 - x_1, \\ x_3 &= 50 - x_1, \\ x_4 &= 30 + x_1. \end{aligned} \quad (2)$$

Подставив в функцию f выражения переменных x_2, x_3 и x_4 из (2), получим более простой вид для f :

$$f = 228 - 0,2x_1.$$

Таким образом, пришли к эквивалентной математической задаче, которую запишем в следующем виде:

$$\begin{aligned} f &= 228 - 0,2x_1 \rightarrow \min; \\ x_2 &= 60 - x_1, \\ x_3 &= 50 - x_1, \\ x_4 &= 30 + x_1. \end{aligned}$$

Мы уже говорили, что расходы на перевозку измеряются неотрицательной величиной $f \geq 0$. Следовательно, минимальное значение f равно нулю.

Построение алгоритма:

1. Положить $f=0$, найти x_1 .

2. Зная x_1 , вычислить, используя (2), значения x_2, x_3, x_4 .

Вычислительный этап в нашей задаче прост и не требует привлечения ЭВМ:

1. $f=0 \Rightarrow 228 - 0,2x_1 = 0 \Rightarrow x_1 = 1140$.

2. $x_2 = 60 - x_1 \Rightarrow x_2 = -1080$.

3. $x_3 = 50 - x_1 \Rightarrow x_3 = -1090$.

4. $x_4 = 30 + x_1 \Rightarrow x_4 = 1170$.

Получили решение: $f=0, x_1=1140, x_2=-1080, x_3=-1090, x_4=1170$, которое удовлетворяет ограничениям и условию минимальности.

Не надо быть мудрецом, чтобы заметить абсурдность полученного решения — для обеспечения нормальной работы с заводов надо вывозить цемент!

Уточнение модели. Противоречие, полученное в результате исследования, заключается в том, что поставки цемента на заводы получились отрицательными. Значит надо исключить такую возможность и ввести в систему ограничений неравенства $x_1 \geq 0, x_2 \geq 0, x_3 \geq 0, x_4 \geq 0$.

Уточненная модель имеет вид

$$\begin{aligned} f &= 1,4x_1 + 2x_2 + 1,2x_3 + 1,6x_4 \rightarrow \min; \\ x_1 + x_2 &= 60, \\ x_3 + x_4 &= 80, \\ x_1 + x_3 &= 50, \\ x_2 + x_4 &= 90, \\ x_1 \geq 0, x_2 \geq 0, x_3 \geq 0, x_4 \geq 0. \end{aligned}$$

Исследование новой модели приводит к следующему виду:

$$\begin{aligned} f &= 228 - 0,2x_1 \rightarrow \min; \\ x_2 &= 60 - x_1; \\ x_3 &= 50 - x_1, \\ x_4 &= 30 + x_1, \\ x_1 \geq 0, x_2 \geq 0, x_3 \geq 0, x_4 \geq 0. \end{aligned} \tag{3}$$

Отсюда, используя неотрицательность переменных x_1, x_2, x_3, x_4 , приходим к эквивалентной математической задаче:

$$\begin{aligned}
 f &= 228 - 0,2x_1 \rightarrow \min; \\
 0 &\leq x_1 \leq 50, \\
 x_2 &= 60 - x_1, \\
 x_3 &= 50 - x_1, \\
 x_4 &= 30 + x_1.
 \end{aligned}
 \tag{4}$$

Проведем математическое исследование задачи (4). Легко видеть, что если x_1 монотонно возрастает, то функция f монотонно убывает. Следовательно, f достигает минимума при максимальном значении x_1 . Зная максимальное значение x_1 , можно найти x_1, x_3, x_4 ,

Построение алгоритма решения (4):

1. Найти максимум x_1 .
2. Вычислить x_2, x_3, x_4 .
3. Вычислить значение f .

Вычислительный этап приводит нас к следующему результату:

1. $x_1 = 50$.
2. $x_2 = 10, x_3 = 0, x_4 = 80$.
3. $f = 218$.

Решение $\{f = 218; x_1 = 50; x_2 = 10, x_3 = 0, x_4 = 80\}$ удовлетворяет (4) и мы получаем, что минимальные затраты на перевозку будут составлять 218 руб, если из первого склада ежедневно вывозить на первый завод 50 т, на второй — 10 т, а из второго склада вывозить только на второй завод по 80 т ежедневно.

В этом примере развитие моделей происходило вследствие уточнения постановки задачи и сравнения полученных численных расчетов с практикой.

В следующем примере развитие получаемых результатов будет происходить в результате анализа численных расчетов и точности проводимых вычислений.

Пример 2 (рис. 19). В пространстве задана прямоугольная декартова система координат $Ox_1x_2x_3$. Из точки $A(3; -2; 0)$ стартовала ракета со скоростью $\mathbf{a}\{1; -5; 1\}$ (в направлении движения значение скорости равно 27). Планета B имеет координаты $(1803; -2; 1500)$. Определить, в какой точке пространства, через какое время после старта ракета будет иметь наименьшее удаление от планеты B и чему равно это расстояние?

Построение математической модели. Так как нет информации о расположении других планет, режимах работы ракетных двигателей и т. д., будем считать, что другие планеты не оказывают влияния на движение и не расположены на траектории полета, т. е. движение ракеты является равномерным и прямолинейным.

При сделанных выше предположениях уравнение траектории полета имеет вид

$$\mathbf{X} = \mathbf{A} + \mathbf{a}t, \tag{5}$$

где \mathbf{X} — радиус-вектор произвольной точки траектории

$X(x_1, x_2, x_3)$; \mathbf{a} — скорость полета;
 t — время; \mathbf{A} — радиус-вектор
 планеты A .

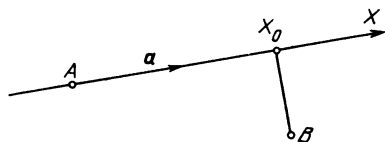


Рис. 19. Траектория полета ракеты

Уравнение (5) представляет собой уравнение прямой линии в пространстве, проходящей через точку A в направлении вектора \mathbf{a} . Кратчайшее расстояние от точки B до прямой (5) совпадает с наименьшим удалением ракеты в полете от планеты B и измеряется длиной перпендикуляра, опущенного из точки B на прямую $\mathbf{X} = \mathbf{A} + \mathbf{a}t$. Обозначим через X_0 основание перпендикуляра, опущенного из B на прямую. Так как точка X_0 лежит на прямой, то существует такое значение параметра $t = t_0$, при котором выполняется равенство

$$\mathbf{X}_0 = \mathbf{A} + \mathbf{a}t_0.$$

Величина t_0 и есть то время, через которое после старта будет достигнуто минимальное удаление, т. е. ракета будет находиться в точке X_0 .

Перепишем уравнение (5) в координатном виде:

$$\begin{aligned} x_1 &= 3 + t, \\ x_2 &= -2 - 5t, \\ x_3 &= t, \quad t \in R. \end{aligned} \quad (6)$$

Расстояние от точки $X(x_1; x_2; x_3)$ до точки B описывается формулой

$$\rho(X, B) = [(x_1 - 1803)^2 + (x_2 + 2)^2 + (x_3 - 1500)^2]^{1/2}. \quad (7)$$

Таким образом, наша математическая модель принимает вид

$$\begin{aligned} \rho(X, B) &\rightarrow \min; \\ x_1 &= 3 + t, \\ x_2 &= -2 - 5t, \\ x_3 &= t, \quad t \in R. \end{aligned}$$

Исследование модели. Так как величина $\rho(X, B) \geq 0$, то достаточно вместо нахождения $\min \rho(X, B)$ найти $\min \rho^2(X, B)$. Подставив ограничения (6) в формулу расстояния (7), получим

$$\rho(X, B) = [((3 + t) - 1803)^2 + ((-2 - 5t) + 2)^2 + (t - 1500)^2]^{1/2},$$

или

$$\rho^2(X, B) = (t - 1800)^2 + 25t^2 + (t - 1500)^2,$$

Раскрыв скобки и приведя подобные члены, приходим к следующей формуле:

$$\rho^2(X, B) = 27t^2 - 6600t + 5\,490\,000. \quad (8)$$

Полученное выражение для $\rho^2(X, B)$ представляет собой квад-

ратный трехчлен от параметра t и найти его минимум не представляет труда. Используя достаточное условие экстремума, получаем уравнение

$$54t - 6600 = 0. \quad (9)$$

Построение алгоритма.

1. Решить уравнение (9), получить t_0 .
2. Подставить t_0 в (8), получить $\rho^2(X, B)$.
3. Подставить t_0 в (6), получить X_0 .

Вычислительный этап (его можно проводить как вручную, так и с помощью компьютера). Решив уравнение (9), получим $t_0 = 6600/54 = 3300/27$. Но число $3300/27$ представляется бесконечной десятичной дробью, поэтому ограничимся двумя знаками после запятой: $t_0 = 122,22$. Подставляя это значение в (8), получаем

$$\rho(X_0, B) \approx 2255,36,$$

а из уравнения (6) вычислим приближенные значения координат точки $X_0(x_1, x_2, x_3)$:

$$x_1 = 125,22; \quad x_2 = -613,1; \quad x_3 = 122,22.$$

И вот решение нашей задачи: через время $t_0 \approx 112,22$ ракета будет находиться на минимальном удалении 2255,36 от планеты B в точке $X_0(125,22; -613,1; 122,22)$.

Уточнение модели. Как хорошо, что в первом примере не было никаких «приблизительно 50 т», а тем более «примерно с первого склада» — везде были конкретные точные числа. Что будем делать? Попробуем положить $t = 122$ и, повторив вычислительный этап, получим $\rho(X, B) = 2255,36$; $X_0(125; -612; 122)$. Рассмотрим, чем отличаются два решения:

$$\{\rho_1(X_0, B) = 2255,36; X_{01}(125,22; -613,1; 122,22); t_{01} = 122,22\}$$

и

$$\{\rho_2(X_0, B) = 2255,36; X_{02}(125; -612; 122); t_{02} = 122\}.$$

Легко проверить, что числа ρ_1 и ρ_2 не различаются, а времена t_{01} и t_{02} попадания в минимально удаленную точку различаются не более чем на 0,2%. Если это не влияет на какие-то работы, о чем ничего не сказано в условии задачи, то мы можем удовлетвориться, например, этой точностью и искать такие решения задачи, чтобы ошибка в вычислениях не превышала $2 \cdot 10^{-2}\%$.

Добавим последнее условие в модель и проведем все необходимые вычисления. Предлагаем читателю проверить, что новая математическая модель

$$\begin{aligned} \rho(X, B) &\rightarrow \min; \\ X &= A + at, \quad t \in R, \\ \text{ошибка вычисления } \rho(X_0, B) &\leq 2 \cdot 10^{-2} \% \end{aligned} \quad (10)$$

имеет бесчисленное множество решений, а значения $t=120$, $t=121$, $t=124$ тоже являются допустимыми для задачи (10) в пределах заданной точности.

Задача. Найдите все допустимые значения t , удовлетворяющие задаче (10).

Бесконечность множества решений не должна пугать, в жизни она встречается достаточно часто. Например, когда вы едете в автомобиле и хотите остановиться как можно ближе к киоску с мороженым, то не измеряете же вы расстояние в миллиметрах или микронах. Поэтому можно заведомо сказать, что если до киоска 10 м, то в интервале 1000 ± 15 см все места остановки машины для вас будут равноудаленными от желаемого, особенно в жару, лакомства.

Как и в этом примере, всегда надо проводить измерения в рамках точности, определяемой конкретными практическими потребностями. Первые два примера характерны тем, что в них изучались математические модели процессов, описываемых числовой информацией. Следующий пример связан с обработкой нечисловой текстовой информации.

Кто из нас, особенно в детстве, не зачитывался рассказами доктора Ватсона о Шерлоке Холмсе. Вспомним хотя бы «Пять зернышек апельсина», «Обряд дома Месгрейвов» и «Пляшущие человечки». В этих произведениях знаменитый сыщик проводит расследование, опираясь на анализ текстовой, а следовательно нечисловой, информации. Вот как рассказывает об этом сам Шерлок Холмс:

«...передо мною эти забавные рисунки, которые могли бы вызвать улыбку, если бы не оказались предвестниками столь страшной трагедии. Я превосходно знаком со всеми видами тайнописи и сам являюсь автором научного труда, в котором проанализировал сто шестьдесят различных шифров, однако я вынужден признаться, что этот шифр для меня совершенная новость. Цель изобретателя этой системы заключалась, очевидно, в том, чтобы сокрыть, что эти значки являются письменами, и выдать их за детские рисунки. Но всякий, кто догадается, что значки эти соответствуют буквам, без особого труда разгадает их, если воспользуется обычными правилами разгадывания шифров».

Как был разгадан шифр великим сыщиком, можно вспомнить, прочитав рассказ «Пляшущие человечки». Не будем осуждать Шерлока Холмса за самоуверенность в обещании без особого труда разгадать закодированный текст, зная только, что символам соответствуют буквы алфавита. Почему? Это будет ясно из наших дальнейших рассуждений.

Пример 3. В пункте А находится экспедиция с радиопередатчиком, который является единственным средством общения с внешним миром. Информация может вводиться в радиопередат-

чик только через аппарат Морзе. Как передавать текстовые отчеты о работе экспедиции по этой связи?

Ответ прост: радист, зная код азбуки Морзе, заменит буквы, цифры в тексте на соответствующие наборы точек и тире, потом передаст эти коды в эфир.

Так как коды Морзе представляют собой наборы из точек и тире, то если в этих кодах мы вместо точки поставим ноль, а вместо тире — единицу, то каждой букве будет соответствовать набор из нулей и единиц (вспомните об этом, когда будете читать § 4.2 о двоичном представлении чисел). Если такой набор рассматривать как некоторое число, то мы пришли к идее замены букв цифрами, т. е. к цифровому кодированию.

Самое простое кодирование состоит в нумерации позиций букв в алфавите и замене буквы ее порядковым номером:

а	б	в	г	д	е	ж	з	и	й	к	л	м
01	02	03	04	05	06	07	08	09	10	11	12	13
н	о	п	р	с	т	у	ф	х	ц	ч	ш	щ
14	15	16	17	18	19	20	21	22	23	24	25	26
ъ	ы	ь	э	ю	я							
27	28	29	30	31	32							

Например, текст «Компьютеры пятого поколения будут хранить большие массивы информации, смогут ее анализировать» при такой замене имеет следующий вид:

```

11 15 13 16 29
31 19 06 17 28
16 32 19 15 04
15 16 15 11 15
12 06 14 09 32
02 20 05 20 19
22 17 01 14 09
19 29 02 15 12
29 25 09 06 13
01 18 18 09 03
28 09 14 21 15
17 13 01 23 09
09 18 13 15 04
20 19 06 06 01
14 01 12 09 08
09 17 15 03 01
19 29

```

В зашифрованном тексте опущены знаки препинания и интервалы между словами, а текст записан последовательно по строкам по пять символов в строке.

Математическая модель этого процесса, кодирования проста и очевидна: надо поставить **то** взаимоднозначное соответствие элементам множества букв алфавита элементы числового множества {01, 02...32} (запись 01, 02, ..., 09 используется только

для единообразия кодирования букв двузначными цифрами). Это соответствие должно удовлетворять следующему условию: если буквы алфавита расположены в принятом порядке а, б, ..., ю, я, то их цифровые коды должны быть упорядочены по возрастанию {01, 02, ..., 31, 32}.

Таким образом, при расшифровке текста надо на место каждой цифры поставить ту букву, порядковым номером которой является эта цифра, и прочитать его.

В исходную задачу введем дополнительное условие: так как переданное в эфир сообщение может принять всякий слушатель, обладающий приемником, настроенным на волну передатчика, то необходимо сделать так, чтобы отдельные сообщения были понятны только тому, кому они адресованы.

Для этого в нашей математической модели достаточно отказать от согласованности порядков элементов в множествах букв и цифр при взаимоднозначном соответствии. Кодирование, при котором порядки букв в алфавите и цифр в натуральном ряду не согласованы, называется перемешивающим.

Таких различных взаимоднозначных соответствий между множествами букв и цифр много. Выберем, например следующее:

а	б	в	г	д	е	ж	з	й	й	к	л	м
12	03	16	14	04	05	25	30	11	31	24	19	07
н	о	п	р	с	т	у	ф	х	ц	ч	ш	щ
27	17	08	22	28	10	18	23	29	02	13	32	30
ъ	ы	ь	э	ю	я							
06	26	21	01	09	15							

При кодировании тот же текст о компьютерах пятого поколения будет выглядеть следующим образом:

24	17	07	08	21
09	10	05	22	26
08	15	10	17	14
17	08	17	24	17
19	05	27	11	15
03	18	04	18	10
29	22	12	27	11
10	21	03	17	19
21	32	09	05	07
12	22	28	11	16
26	11	14	21	17
22	07	12	02	11
11	28	07	17	14
18	10	05	05	12
27	12	19	11	20
11	22	17	16	12
10	21			

Расшифровка данного сообщения без знания взаимоднозначного соответствия множеств букв алфавита и их кодов является

весьма трудоемким занятием. Если просто перебирать всевозможные варианты взаимоднозначных соответствий, а их число равно $1 \cdot 2 \cdot 3 \dots 31 \cdot 32 = 32!$, то потребуется много лет ручного труда. При решении этой переборной задачи большую помощь может оказать ЭВМ, но и с ее помощью на решение уйдет достаточно много времени.

Обратите внимание, что Шерлок Холмс при расшифровке пляшущих человечков использовал два подхода — интуитивный и статистический. В первом случае, учитывая, что письма начинаются, как правило, с обращения к адресату по имени, он стал обладателем кодов двух букв. Во втором случае ..., но лучше предоставим возможность рассказать об этом самому сыщику:

«Коротенькое словечко из двух букв, стоящее перед фамилией, по всей вероятности, имя. Какое же имя может состоять из двух букв? В Америке весьма распространено имя «Аб»».

Так, Шерлок Холмс стал обладателем кодов букв «а» и «б». Примененный им метод можно назвать статистическим, так как выражение «весьма распространено» соответствует по смыслу фразе «очень часто встречается». Использование данных о наиболее часто встречающихся буквах, словах и лежит в основе статистических методов расшифровки, отличающихся от перебора всевозможных взаимоднозначных соответствий между буквами и их кодами. Суть таких методов заключается в следующем: составляют таблицы, содержащие информацию о том, как часто каждая буква алфавита встречается в словах родного языка. Цифры, чаще встречающиеся в исследуемом зашифрованном тексте, заменяют на буквы, имеющие наибольший коэффициент использования. Чем длиннее текст, тем ближе будут значения коэффициентов частот повторяемости цифр в шифрованной записи и букв в различных словах исходного языка.

Но и этим подходом не исчерпываются все методы, применяемые при расшифровке текстов. В настоящее время существует и активно развивается целое математическое направление — теория информации, или теория передачи информации. Развитие работ в этой области тесно связано с исследованием космического пространства, организацией передачи информации, собранной межпланетными космическими станциями, и т. д. Важнейшими здесь являются проблемы выбора такого способа представления (кодирования) собранной информации, чтобы при передаче на Землю расходовалось наименьшее количество энергии и времени, при кодировании, передаче и расшифровке появлялось как можно меньше ошибок, а сами ошибки можно было легко обнаружить и устранить и т. д. Для решения этих задач требуются большие объемы информации, обработка которых использует большие вычислительные мощности, предоставляемые современными ЭВМ.

Аналогичные задачи встречаются и при организации пересылки, преобразования информации в различных устройствах

01 02 03 04 05 06 07 08 09 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31 32

	А	Б	В	Г	Д	Е	Ж	З	И	Й	К	Л	М	Н	О	П	Р	С	Т	У	Ф	Х	Ц	Ч	Ш	Щ	Ъ	Ы	Ь	Э	Ю	Я
А	Э	Ц	Б	Д	Е	Ъ	М	П	Ю	Т	И	А	Ч	Г	Я	В	О	У	Л	З	Ь	Р	Ф	К	Ж	Ы	Н	С	Х	Ш	Й	Ш
Б	Ш	Э	Ц	Б	Д	Е	Ъ	М	П	Ю	Т	И	А	Ч	Г	Я	В	О	У	Л	З	Ь	Р	Ф	К	Ж	Ы	Н	С	Х	Ш	Й
В	Й	Ш	Э	Ц	Б	Д	Е	Ъ	М	П	Ю	Т	И	А	Ч	Г	Я	В	О	У	Л	З	Ь	Р	Ф	К	Ж	Ы	Н	С	Х	Ш
Г	Ш	Й	Ш	Э	Ц	Б	Д	Е	Ъ	М	П	Ю	Т	И	А	Ч	Г	Я	В	О	У	Л	З	Ь	Р	Ф	К	Ж	Ы	Н	С	Х
Д	Х	Ш	Й	Ш	Э	Ц	Б	Д	Е	Ъ	М	П	Ю	Т	И	А	Ч	Г	Я	В	О	У	Л	З	Ь	Р	Ф	К	Ж	Ы	Н	С
Е	С	Х	Ш	Й	Ш	Э	Ц	Б	Д	Е	Ъ	М	П	Ю	Т	И	А	Ч	Г	Я	В	О	У	Л	З	Ь	Р	Ф	К	Ж	Ы	Н
Ж	Н	С	Х	Ш	Й	Ш	Э	Ц	Б	Д	Е	Ъ	М	П	Ю	Т	И	А	Ч	Г	Я	В	О	У	Л	З	Ь	Р	Ф	К	Ж	Ы
З	Ы	Н	С	Х	Ш	Й	Ш	Э	Ц	Б	Д	Е	Ъ	М	П	Ю	Т	И	А	Ч	Г	Я	В	О	У	Л	З	Ь	Р	Ф	К	Ж
И	Ж	Ы	Н	С	Х	Ш	Й	Ш	Э	Ц	Б	Д	Е	Ъ	М	П	Ю	Т	И	А	Ч	Г	Я	В	О	У	Л	З	Ь	Р	Ф	К
Й	К	Ж	Ы	Н	С	Х	Ш	Й	Ш	Э	Ц	Б	Д	Е	Ъ	М	П	Ю	Т	И	А	Ч	Г	Я	В	О	У	Л	З	Ь	Р	Ф
К	Ф	К	Ж	Ы	Н	С	Х	Ш	Й	Ш	Э	Ц	Б	Д	Е	Ъ	М	П	Ю	Т	И	А	Ч	Г	Я	В	О	У	Л	З	Ь	Р
Л	Р	Ф	К	Ж	Ы	Н	С	Х	Ш	Й	Ш	Э	Ц	Б	Д	Е	Ъ	М	П	Ю	Т	И	А	Ч	Г	Я	В	О	У	Л	З	Ь
М	Ь	Р	Ф	К	Ж	Ы	Н	С	Х	Ш	Й	Ш	Э	Ц	Б	Д	Е	Ъ	М	П	Ю	Т	И	А	Ч	Г	Я	В	О	У	Л	З
Н	З	Ь	Р	Ф	К	Ж	Ы	Н	С	Х	Ш	Й	Ш	Э	Ц	Б	Д	Е	Ъ	М	П	Ю	Т	И	А	Ч	Г	Я	В	О	У	Л
О	Л	З	Ь	Р	Ф	К	Ж	Ы	Н	С	Х	Ш	Й	Ш	Э	Ц	Б	Д	Е	Ъ	М	П	Ю	Т	И	А	Ч	Г	Я	В	О	У
П	У	Л	З	Ь	Р	Ф	К	Ж	Ы	Н	С	Х	Ш	Й	Ш	Э	Ц	Б	Д	Е	Ъ	М	П	Ю	Т	И	А	Ч	Г	Я	В	О
Р	О	У	Л	Х	Ь	Р	Ф	К	Ж	Ы	Н	С	Х	Ш	Й	Ш	Э	Ц	Б	Д	Е	Ъ	М	П	Ю	Т	И	А	Ч	Г	Я	В
С	В	О	У	Л	З	Ь	Р	Ф	К	Ж	Ы	Н	С	Х	Ш	Й	Ш	Э	Ц	Б	Д	Е	Ъ	М	П	Ю	Т	И	А	Ч	Г	Я
Т	Я	В	О	У	Л	З	Ь	Р	Ф	К	Ж	Ы	Н	С	Х	Ш	Й	Ш	Э	Ц	Б	Д	Е	Ъ	М	П	Ю	Т	И	А	Ч	Г
У	Г	Я	В	О	У	Л	З	Ь	Р	Ф	К	Ж	Ы	Н	С	Х	Ш	Й	Ш	Э	Ц	Б	Д	Е	Ъ	М	П	Ю	Т	И	А	Ч
Ф	Ч	Г	Я	В	О	У	Л	З	Ь	Р	Ф	К	Ж	Ы	Н	С	Х	Ш	Й	Ш	Э	Ц	Б	Д	Е	Ъ	М	П	Ю	Т	И	А
Х	А	Ч	Г	Я	В	О	У	Л	З	Ь	Р	Ф	К	Ж	Ы	Н	С	Х	Ш	Й	Ш	Э	Ц	Б	Д	Е	Ъ	М	П	Ю	Т	И
Ц	И	А	Ч	Г	Я	В	О	У	Л	З	Ь	Р	Ф	К	Ж	Ы	Н	С	Х	Ш	Й	Ш	Э	Ц	Б	Д	Е	Ъ	М	П	Ю	Т
Ч	Т	И	А	Ч	Г	Я	В	О	У	Л	З	Ь	Р	Ф	К	Ж	Ы	Н	С	Х	Ш	Й	Ш	Э	Ц	Б	Д	Е	Ъ	М	П	Ю
Ш	Ю	Т	И	А	Ч	Г	Я	В	О	У	Л	З	Ь	Р	Ф	К	Ж	Ы	Н	С	Х	Ш	Й	Ш	Э	Ц	Б	Д	Е	Ъ	М	П
Щ	П	Ю	Т	И	А	Ч	Г	Я	В	О	У	Л	З	Ь	Р	Ф	К	Ж	Ы	Н	С	Х	Ш	Й	Ш	Э	Ц	Б	Д	Е	Ъ	М
Ъ	М	П	Ю	Т	И	А	Ч	Г	Я	В	О	У	О	З	Ь	Р	Ф	К	Ж	Ы	Н	С	Х	Ш	Й	Ш	Э	Ц	Б	Д	Е	Ъ
Ы	Ъ	М	П	Ю	Т	И	А	Ч	Г	Я	В	О	У	Л	З	Ь	Р	Ф	К	Ж	Ы	Н	С	Х	Ш	Й	Ш	Э	Ц	Б	Д	Е
Ь	Е	Ъ	М	П	Ю	Т	И	А	Ч	Г	Я	В	О	У	Л	З	Ь	Р	Ф	К	Ж	Ы	Н	С	Х	Ш	Й	Ш	Э	Ц	Б	Д
Э	Д	Е	Ъ	М	П	Ю	Т	И	А	Ч	Г	Я	В	О	У	Л	З	Ь	Р	Ф	К	Ж	Ы	Н	С	Х	Ш	Й	Ш	Э	Ц	Б
Ю	Б	Д	Е	Ъ	М	П	Ю	Т	И	А	Ч	Г	Я	В	О	У	Л	З	Ь	Р	Ф	К	Ж	Ы	Н	С	Х	Ш	Й	Ш	Э	Ц
Я	Ц	Б	Д	Е	Ъ	М	П	Ю	Т	И	А	Ч	Г	Я	В	О	У	Л	З	Ь	Р	Ф	К	Ж	Ы	Н	С	Х	Ш	Й	Ш	Э

ЭВМ. Желаящие ближе познакомиться с этими проблемами могут обратиться к литературе по теории информации, теории кодирования и подробнее узнать суть используемых математических моделей и методов их анализа. А мы в заключение покажем, какими сложными могут быть соответствия между множествами букв алфавита и их цифровыми кодами.

Метод кодирования Виженера с использованием перемешанного алфавита. Для составления таблицы Виженера (табл. 1) воспользуемся перемешанным алфавитом (11).

В первых двух строках таблицы записаны буквы алфавита в принятом порядке и номера их позиций. В третьей строке записан перемешанный алфавит (11). Каждая последующая строка получена из предыдущей путем перестановки последней буквы на первое место и сдвига всех остальных букв на одну позицию вправо. В крайнем левом столбце записан алфавит в принятом естественном (неперемешанном) порядке. Осталось выбрать ключевое слово. В память о великом разгадывателе запутанных историй возьмем его фамилию ХОЛМС в качестве ключевого слова.

Процесс кодирования (построение отображения множества букв алфавита в множество цифр $\{0, 1, \dots, 32\}$) происходит следующим образом:

запишем кодируемую фразу, а под ее словами ключевое слово:

КОМПЬЮТЕРЫ	ПЯТОГО	ПОКОЛЕНИЯ	БУДУТ	ХРАНИТЬ	БОЛЬШИЕ
ХОЛМСХОЛМС	ХОЛМСХ	ОЛМСХОЛМС	ХОЛМС	ХОЛМСХО	ЛМСХОЛМ

МАССИВЫ	ИНФОРМАЦИИ	СМОГУТ	ЕЕ	АНАЛИЗИРОВАТЬ	
СХОЛМСХ	ОЛМСХОЛМСХ	ОЛМСХО	ЛМ	СХОЛМСХОЛМСХО	(12)

Процесс кодирования происходит следующим образом: В (12) в первом столбце первого слова расположены буквы К и Х. В квадрате Виженера во второй строке найдем букву К, а в крайнем левом столбце — букву Х. На пересечении столбца К и строки Х находится буква Р, запишем ее, далее из второго столбца (12) по буквам О и О найдем букву Э и т. д. В итоге получим

КОМПЬЮТЕРЫ	ПЯТОГО	ПОКОЛЕНИЯ	БУДУТ	ХРАНИТЬ	БОЛЬШИЕ
ХОЛМСХОЛМС	ХОЛМСХ	ОЛМСХОЛМС	ХОЛМС	ХОЛМСХО	ЛМСХОЛМ
РЕЦДАТЕНЕИ	НУПБЛЫ	ЦДИЦФКБХЯ	ТБЫПЦ	ЭБРЦКШЯ	ФБНПИШЬ

МАССИВЫ	ИНФОРМАЦИИ	СМОГУТ	ЕЕ	АНАЛИЗИРОВАТЬ	
СХОЛМСХ	ОЛМСХОЛМСХ	ОЛМСХО	ЛМ	СХОЛМСХОЛМСХО	(13)
САДМХУМ	НБМШСЯРИКЗ	ДЦБЛИЕ	НЫ	ВЖЛЭХФЗБДФШЦЯ	

Теперь, используя первую и вторую строки квадрата Виженера, запишем построенную третью строчку из (13) в числовых кодах (как и ранее, опустим знаки препинания и интервалы между словами, а текст запишем по строкам по пять цифр в каждой строке):

17,	30,	23,	05,	01
19,	06,	14,	06,	09
14,	20,	16,	02,	12
28,	23,	05,	10,	26
21,	11,	02,	22,	32
19,	27,	28,	16,	23
30,	02,	17,	23,	11
26,	32,	21,	02,	14
16,	09,	26,	28,	18
01,	05,	13,	22,	20
13,	14,	02,	13,	26
18,	10,	17,	09,	11
08,	05,	23,	02,	12
10,	06,	14,	28,	03
07,	12,	30,	22,	21
08,	29,	05,	21,	03
26,	32			

Интересно, смог бы знаменитый Холмс расшифровать приведенный цифровой текст, не зная ни ключевого слова, ни квадрата Виженера? Сомнительно. Современные криптографы, а так называют специалистов по кодированию-декодированию текстов, вооруженные методами современной теории информации и кодирования и использующие мощные вычислительные машины, такую задачу научились решать.

Читатель, имеющий опыт программирования и желание попробовать свои силы в этой области, может решить с помощью ЭВМ следующую задачу: составить программу для ЭВМ, которая, зная квадрат Виженера, по заданному ключевому слову шифрует и расшифровывает полученные сообщения. Попробуйте!

4.2. ПРЕДСТАВЛЕНИЕ ЧИСЕЛ В ЭВМ И СИСТЕМЫ СЧИСЛЕНИЯ

Рассмотренные примеры математических моделей подсказывают нам, что человек применяет их (модели) для создания некоторого промежуточного, между собой и компьютером, описания процессов и явлений окружающего мира. Правильная математическая модель помогает не только описать последовательность, логику процессов, но и найти количественные (числовые) характеристики интересующих нас явлений. А компьютеры и родились в свое время именно для обработки числовой информации, вот почему с появлением ЭВМ стремительно стали развиваться количественные методы.

Человек воспринимает и описывает окружающий его мир по-разному: качественно (например: «Впереди плыл большой белый пароход») или с помощью чисел — количественно (например: «Корабль водоизмещением 5000 т плыл со скоростью 15 узлов»). Мы все это понимаем, можем представить и, если необходимо, использовать. А каким видит мир ЭВМ, ведь инфор-

мация к ней поступает из математических моделей, приборов и т. д.? Как она различает числа, закодированную цифрами информацию и т. д.? Эти вопросы и будут рассмотрены в данном и следующих параграфах.

Начнем с описания различных систем счисления.

Прежде всего посмотрим, что представляет собой *десятичная система счисления*, которой мы пользуемся с детства. Например, целое положительное число в ней представляется в виде суммы единиц, десятков, сотен, тысяч и т. д. Действительно, число 245 можно представить в виде

$$2 \cdot 10^2 + 4 \cdot 10^1 + 5 \cdot 10^0,$$

где цифра 2 показывает количество сотен, 4 — количество десятков и последняя цифра, 5 — количество единиц. Таким образом, запись числа в виде 245 представляет собой перечисление коэффициентов в разложении этого числа по степени 10.

Аналогично любое число вида

$$a_n \cdot 10^n + a_{n-1} \cdot 10^{n-1} + \dots + a_1 \cdot 10^1 + a_0 \cdot 10^0 + a_{-1} \cdot 10^{-1} + a_{-2} \cdot 10^{-2} + \dots \quad (14)$$

можно записать, перечислив его коэффициенты a_i в разложении по степеням числа 10 в привычном нам виде: $a_n a_{n-1} \dots a_1 a_0 a_{-1} a_{-2} \dots$. Число 10 называется основанием десятичной системы счисления. Обратите внимание, что i — номер коэффициента a_i — совпадает с показателем степени числа 10 в разложении (14). Так как десять единиц младшего разряда составляют одну единицу старшего разряда, то каждый из коэффициентов a_i представляет собой некоторую цифру из набора $\{0, 1, 2, 3, 4, 5, 6, 7, 8, 9\}$. Этот набор цифр составляет базис десятичной системы счисления. Заметьте, что число цифр в базисном наборе равно 10 и совпадает с основанием рассматриваемой системы счисления.

Десятичная система счисления получила широкое распространение по той простой причине, что в древние времена первоначальным аппаратом для проведения счета выступали десять пальцев рук. Однако она не является единственной. В качестве основания системы счисления можно взять, например, любое целое $p > 1$, а для базисной системы коэффициентов a_i использовать набор из p различных цифр $\{0, 1, \dots, p-1\}$. Произвольное число в системе счисления с основанием p имеет вид

$$a_n \cdot p^n + a_{n-1} \cdot p^{n-1} + \dots + a_1 \cdot p^1 + a_0 \cdot p^0 + a_{-1} p^{-1} + a_{-2} p^{-2} + \dots,$$

где

$$a_i \in \{0, 1, \dots, p-1\}, i \in \mathbb{Z}.$$

Кратко это число можно записать, как мы это уже делали выше, перечислив коэффициенты при степенях числа p , отделив запятой коэффициенты с неотрицательными номерами от коэффициентов с отрицательными номерами.

Например, десятичное число 18,25 в системе счисления с основанием $p=4$ можно записать так: $1 \cdot 4^2 + 0 \cdot 4^1 + 2 \cdot 4^0 + 1 \cdot 4^{-1} = 102,1_4$ (индекс 4 в правой части равенства говорит о том, что перед нами число, записанное в системе счисления с основанием 4).

Минимальным значением основания системы счисления, $p > 1$ является число 2. В *двоичной системе* базисная система коэффициентов содержит всего две цифры, 0 и 1.

Так, например, десятичное число 51,5 в двоичном представлении имеет вид

$$51,5 = 1 \cdot 2^5 + 1 \cdot 2^4 + 0 \cdot 2^3 + 0 \cdot 2^2 + 1 \cdot 2^1 + 1 \cdot 2^0 + 1 \cdot 2^{-1} = 110011,1_2.$$

Теперь уже в качестве легкого упражнения вы можете проверить, что обычные десятичные числа 0, 1, 2, 3, 4, 5, 6, 7, 8, 9 имеют в двоичной системе следующий вид: 0000, 0001, 0010, 0011, 0100, 0101, 0110, 0111, 1000, 1001 (вспомните, пожалуйста, § 2.1).

Двоичная система счисления получила широкое распространение с появлением ЭВМ. Дело в том, что, как мы уже знаем, любое число в этой системе представляется сочетанием нулей и единиц. Это позволяет достаточно просто организовать хранение и переработку информации, представленной в двоичном виде. Например, для того чтобы различать цифры в такой системе, достаточно иметь устройство, обладающее двумя устойчивыми состояниями, одно из которых соответствует единице, другое — нулю. (Для сравнения аналогичная задача в десятичной системе счисления требует устройства с десятью различными устойчивыми состояниями.) Другим важным достоинством двоичной системы счисления является простота вычислений. Отметим, кстати, что выполнение арифметических действий над числами в двоичной системе счисления производится по тем же правилам, что и в десятичной. При этом пользуются соответствующими таблицами (рис. 20—22).

Правило сложения:

$$\begin{aligned} 0 + 0 &= 0, \\ 0 + 1 &= 1, \\ 1 + 0 &= 1, \\ 1 + 1 &= 10. \end{aligned}$$

Пример. Сложить два числа, 10101_2 и 1011_2 :

$$\begin{array}{r} 10101 \\ + 1011 \\ \hline 100000 \end{array}$$

Проверим, соответствует ли результат сложению этих чисел в десятичной системе:

$$10101_2 = 1 \cdot 2^4 + 0 \cdot 2^3 + 1 \cdot 2^2 + 0 \cdot 2^1 + 1 \cdot 2^0 = 21_{10},$$

$$1011_2 = 1 \cdot 2^3 + 0 \cdot 2^2 + 1 \cdot 2^1 + 1 \cdot 2^0 = 11_{10},$$

$$100000_2 = 1 \cdot 2^5 + 0 \cdot 2^4 + 0 \cdot 2^3 + 0 \cdot 2^2 + 0 \cdot 2^1 + 0 \cdot 2^0 = 32_{10}.$$

+	0	1
0	0	1
1	1	10

Рис. 20. Таблица сложения

-	0	1
0	0	1
1	-1	0

Рис. 21. Таблица вычитания

\times	0	1
0	0	0
1	0	1

Рис. 22. Таблица умножения

Сумма чисел, расположенных справа в первых двух строках, равна $21_{10} + 11_{10} = 32_{10}$ и совпадает с числами 100000_2 , что и требовалось доказать.

Правило вычитания:

$$\begin{aligned} 0 - 0 &= 0, \\ 1 - 0 &= 1, \\ 1 - 1 &= 0, \\ 0 - 1 &= -1. \end{aligned}$$

Пример. Из числа 110001_2 вычесть 111_2 :

$$\begin{array}{r} 110001 \\ - 111 \\ \hline 101010 \end{array}$$

Убедимся в правильности полученного результата:

$$110001_2 = 1 \cdot 2^5 + 1 \cdot 2^4 + 0 \cdot 2^3 + 0 \cdot 2^2 + 0 \cdot 2^1 + 0 \cdot 2^0 = 49_{10},$$

$$111_2 = 1 \cdot 2^2 + 1 \cdot 2^1 + 1 \cdot 2^0 = 7_{10},$$

$$101010 = 1 \cdot 2^5 + 0 \cdot 2^4 + 1 \cdot 2^3 + 0 \cdot 2^2 + 1 \cdot 2^1 + 0 \cdot 2^0 = 42_{10}.$$

Очевидно, что разность $49_{10} - 7_{10} = 42_{10}$ совпадает с числом 101010 .

Правило умножения:

$$\begin{aligned} 0 \times 0 &= 0, \\ 1 \times 0 &= 0, \\ 0 \times 1 &= 0, \\ 1 \times 1 &= 1. \end{aligned}$$

Пример. Умножить два числа, 10101_2 и 1011_2 :

$$\begin{array}{r} \times 10101 \\ 1011 \\ \hline 10101 \\ 10101 \\ 10101 \\ 10101 \\ \hline 11100111 \end{array}$$

Произведем проверку: $10101_2 = 21_{10}$, $1011_2 = 11_{10}$ и $21_{10} \times 11_{10} = 231_{10}$. С другой стороны: $11100111_2 = 1 \cdot 2^7 + 1 \cdot 2^6 + 1 \cdot 2^5 + 0 \cdot 2^4 + 0 \cdot 2^3 + 1 \cdot 2^2 + 1 \cdot 2^1 + 1 \cdot 2^0 = 231_{10}$, следовательно, получили верный результат.

В основе операции деления лежит процесс сравнения делителя с остатком, полученным на каждом шаге алгоритма.

Пример. Разделить число 10101_2 на 111_2 :

$$\begin{array}{r} 10101 \overline{) 111} \\ \underline{111} \\ 111 \\ \underline{111} \\ 0 \end{array}$$

Проверим результат: $10101_2 = 21_{10}$, $111_2 = 7_{10}$. В десятичной системе счисления $21:7=3$. Действительно, $11 = 1 \cdot 2^1 + 1 \cdot 2^0 = 3_{10}$.

Отметим недостаток, характерный для двоичной системы счислений — значительный рост числа разрядов, необходимых для изображения чисел. Например, двузначное десятичное число 22_{10} в двоичной системе счисления имеет вид 10110_2 , т. е. для его изображения требуется уже 5 разрядов. Вместе с тем перечисленные выше достоинства двоичной системы делают этот недостаток не столь существенным.

Интерес представляют также восьмеричная и шестнадцатеричная системы.

В случае *восьмеричной системы счисления* основанием системы служит число 8, а базис системы составляет набор цифр $\{0, 1, 2, 3, 4, 5, 6, 7\}$.

Например:

$$(765,4)_8 = 7 \cdot 8^2 + 6 \cdot 8^1 + 5 \cdot 8^0 + 4 \cdot 8^{-1} = 501,5_{10}.$$

При записи чисел в *шестнадцатеричной системе* необходимо использовать 16 цифр, составляющих базис системы. Однако только десять цифр из шестнадцати имеют общепринятое обозначение 0—9. Для записи остальных базисных чисел (10, 11, 12, 13, 14, 15) обычно используют символы A, B, C, D, E, F либо $\overline{10}$, $\overline{11}$, $\overline{12}$, $\overline{13}$, $\overline{14}$, $\overline{15}$.

Таким образом, запись $3AF_{16}$ означает:

$$(3AF)_{16} = (3 \overline{10} \overline{15})_{16} = 3 \cdot 16^2 + 10 \cdot 16^1 + 15 \cdot 16^0 = 256 + 160 + 15 = 431_{10}.$$

Арифметические операции над числами в восьмеричной и шестнадцатеричной системах, как, впрочем, и в любой другой системе, выполняются аналогично соответствующим операциям в рассмотренных выше десятичной и двоичной системах. Заметим только, что в каждом случае используется своя таблица сложения и умножения. Не будем останавливаться на подробном описании выполнения арифметических действий в восьмеричной и шестнадцатеричной системах счисления, а приведем лишь таблицы сло-

Таблица 2. Таблица сложения

+	0	1	2	3	4	5	6	7
0	0	1	2	3	4	5	6	7
1	1	2	3	4	5	6	7	10
2	2	3	4	5	6	7	10	11
3	3	4	5	6	7	10	11	12
4	4	5	6	7	10	11	12	13
5	5	6	7	10	11	12	13	14
6	6	7	10	11	12	13	14	15
7	7	10	11	12	13	14	15	16

Таблица 3. Таблица умножения

×	0	1	2	3	4	5	6	7
0	0	0	0	0	0	0	0	0
1	0	1	2	3	4	5	6	7
2	0	2	4	6	10	12	14	16
3	0	3	6	11	14	17	22	25
4	0	4	10	14	20	24	30	34
5	0	5	12	17	24	31	36	43
6	0	6	14	22	30	36	44	52
7	0	7	16	25	34	43	52	61

жения и умножения в восьмеричной системе (табл. 2, 3).

Предлагаем читателю попробовать составить таблицы деления и умножения для систем с другими основаниями, например троичной и шестнадцатеричной.

Мы уже отмечали, что большинство устройств ЭВМ работают с числами, представленными в двоичной системе счисления, однако нам с вами задавать исходные данные и анализировать результаты вычислений, представленные в виде последовательностей нулей и единиц, неудобно: мы привыкли использовать десятичную систему счисления. Поэтому нужен простой способ записи десятичных чисел с помощью символов двоичной системы счисления, т. е. нулей и единиц. Такую возможность обеспечивает двоично-десятичная система счисления. В ней каждая цифра десятичного представления числа записывается двоичными знаками. Для того чтобы запись числа в двоично-десятичной системе счисления была однозначной, для изображения каждой десятичной цифры отводится одно и то же число двоичных разрядов, а именно четыре. (Это следует из того, что наибольшее базисное число десятичной системы счисления 9 требует для своего изображения четыре символа 1001.) Ниже приведена запись в двоично-десятичной системе счисления чисел от 0 до 9 составляющих базис десятичной системы счисления:

$$\begin{array}{ll}
 0_{10} = 0000_{2-10}, & 5_{10} = 0101_{2-10}, \\
 1_{10} = 0001_{2-10}, & 6_{10} = 0110_{2-10}, \\
 2_{10} = 0010_{2-10}, & 7_{10} = 0111_{2-10}, \\
 3_{10} = 0011_{2-10}, & 8_{10} = 1000_{2-10}, \\
 4_{10} = 0100_{2-10}, & 9_{10} = 1001_{2-10}.
 \end{array}$$

Например, десятичное число 9024,19 в двоично-десятичной системе счисления имеет вид

$$9024,19_{10} = 1001000000100100, 00011001_{2-10},$$

а число $153,78_{10}$ может быть представлено следующим образом:

$$000101010011, 01111000_{2-10}.$$

Нули, стоящие в крайних позициях, как правило, опускают. Тогда приходим к виду

$$101010011, 01111_{2-10} = 153,78_{10}.$$

Весьма прост и обратный переход от двоично-десятичной системы счисления к десятичной. Для этого необходимо двоично-десятичное представление числа разбить на группы по четыре символа (тетрады) влево и вправо от запятой, а затем каждую тетраду заменить десятичной цифрой в соответствии с таблицей. Например, последовательность двоичных знаков, разделенных запятой, $10000011, 00011001_{2-10}$ соответствует числу $83,19_{10}$.

Следует иметь в виду тот факт, что запись числа в двоично-десятичной системе не совпадает в общем случае с записью того же числа в двоичной системе. В этом легко убедиться на следующем примере. Рассмотрим набор двоичных цифр $10101,1$. Легко видеть, что такая последовательность представляет собой двоично-десятичную запись десятичного числа $15,8$, а если теперь эту же последовательность двоичных цифр интерпретировать как число, записанное в двоичной системе счисления, то получим

$$1 \cdot 2^4 + 0 \cdot 2^3 + 1 \cdot 2^2 + 0 \cdot 2^1 + 1 \cdot 2^0 + 1 \cdot 2^{-1} = 16 + 4 + 1 + 1/2 = 21,5_{10}.$$

Очевидно, что полученные результаты различны.

Из других смешанных систем счисления наибольший интерес представляет двоично-восьмеричная система. Базис восьмеричной системы составляют числа от 0 до 7. Следовательно, для изображения любой базисной цифры достаточно трех двоичных символов. Группы из трех разрядов для изображения восьмеричных цифр принято называть триадами.

Выпишем изображение базисных цифр в двоично-восьмеричной системе:

$$\begin{array}{ll} 0_8 = 000_{2-8}, & 4_8 = 100_{2-8}, \\ 1_8 = 001_{2-8}, & 5_8 = 101_{2-8}, \\ 2_8 = 010_{2-8}, & 6_8 = 110_{2-8}, \\ 3_8 = 011_{2-8}, & 7_8 = 111_{2-8}. \end{array}$$

Интересно отметить, что представление восьмеричных чисел от 0 до 7 в двоичной системе счисления тождественно совпадает с их представлением в двоично-восьмеричной системе, и это справедливо не только для базисных цифр. Рассмотрим, например, число 724_8 .

Очевидно, что

$$724_8 = 111010100_{2-8}.$$

С другой стороны, мы знаем, что

$$724_8 = 7 \cdot 8^2 + 2 \cdot 8^1 + 4 \cdot 8^0 = 468_{10}.$$

Легко убедиться в том, что

$$468_{10} = 1 \cdot 2^8 + 1 \cdot 2^7 + 1 \cdot 2^6 + 0 \cdot 2^5 + 1 \cdot 2^4 + 0 \cdot 2^3 + 1 \cdot 2^2 + 0 \cdot 2^1 + 0 \cdot 2^0 = 111010100_2.$$

Очевидно, что

$$111010100_{2-8} = 724_8 = 468_{10} = 111010100_2,$$

т. е. двоично-восьмеричная и двоичная запись одного и того же числа совпали. Это свойство позволяет рассматривать восьмеричную запись числа как сокращенную запись того же числа в двоичной системе счисления.

Перевод целых и дробных чисел из одной системы счисления в другую. Как мы уже отмечали, человек привык работать в десятичной системе, а ЭВМ ориентирована на двоичную систему, поэтому общение человека с машиной невозможно без создания простых и надежных алгоритмов перевода чисел из одной системы в другую и наоборот.

Итак, как осуществляется перевод из десятичной системы счисления в двоичную? Мы знаем, что запись произвольного числа

$$x = \alpha_k \cdot 2^k + \alpha_{k-1} \cdot 2^{k-1} + \dots + \alpha_1 \cdot 2^1 + \alpha_0 \cdot 2^0 + \alpha_{-1} \cdot 2^{-1} + \dots + \alpha_{-2} \cdot 2^{-2} + \dots$$

в двоичной системе счисления представляет собой последовательность цифр $\alpha_k \alpha_{k-1} \dots \alpha_1 \alpha_0, \alpha_{-1} \alpha_{-2}$, каждая из которых принимает значения 0 либо 1. Таким образом, для записи числа x в двоичной системе счисления необходимо уметь определять значения коэффициентов разложения x по степеням числа 2 $\alpha_k, \dots, \alpha_1, \alpha_0, \alpha_{-1}, \dots$.

Рассмотрим отдельно перевод целых чисел и правильных дробей.

Пусть x — первое число, тогда в его разложении отсутствуют коэффициенты с отрицательными индексами, т. е. его можно представить в виде

$$x = \alpha_k \cdot 2^k + \dots + \alpha_1 \cdot 2^1 + \alpha_0 \cdot 2^0.$$

Разделим число x на 2. Частное от деления будет равно $\alpha_k \cdot 2^{k-1} + \dots + \alpha_1$, а остаток равен α_0 . Полученное частное опять разделим на 2, остаток от деления будет равен α_1 .

Если теперь продолжить этот процесс деления, то на $(k+1)$ -м шаге получим набор цифр $\alpha_0, \alpha_1, \alpha_2, \dots, \alpha_k$, которые входят

в двоичное представление числа x и совпадают с остатками при последовательном делении x на 2. Обратите внимание, что мы получим цифры α_i в порядке, обратном порядку расположения в двоичном представлении числа x :

$$x = (\alpha_k \alpha_{k-1} \dots \alpha_1 \alpha_0)_2.$$

Примеры.

1. Перевести число 11_{10} в двоичную систему счисления. Разделим 11 на 2. В качестве частного имеем 5, а остаток равен 1. Следовательно, в двоичной записи числа 11 последняя цифра равна 1. Разделим теперь 5 на 2 и получим соответственно 2 и 1. Таким образом, получили вторую цифру в двоичной записи числа 11, она также равна 1. Наконец, при делении 2 на 2 в качестве остатка получим число 0, которое и является третьей цифрой в двоичной записи числа 11. Частное равно единице, которая в целых числах на 2 не делится. Оно и является последней, четвертой цифрой в записи десятичного числа 11 в двоичной системе счисления. Таким образом,

$$11_{10} = 1011_2.$$

Рассмотренную выше последовательность действий (алгоритм) перевода целого числа из десятичной системы счисления в двоичную удобнее изобразить так:

$$\begin{array}{r|l} 11 & 2 \\ \hline (1) & 5 \\ \hline & (1) & 2 \\ & \hline & & 0 & 1 \end{array}$$

Собирая остатки от последовательного деления в направлении, указанном стрелкой, получим $11_{10} = 1011_2$.

2. Перевести число 49_{10} в двоичную систему счисления:

$$\begin{array}{r|l} 49 & 2 \\ \hline (1) & 24 \\ \hline & (0) & 12 \\ & \hline & & (0) & 6 \\ & \hline & & & (0) & 3 \\ & \hline & & & & (1) & 2 \\ & \hline & & & & & 0 & 1 \end{array}$$

Итак, $49_{10} = 1010001_2$.

Пусть теперь x — десятичная дробь, тогда в разложении отсутствуют коэффициенты с положительными индексами:

$$x = \alpha_{-1} \cdot 2^{-1} + \alpha_{-2} \cdot 2^{-2} + \dots \quad (15)$$

В двоичной системе счисления $x = (0, \alpha_{-1} \alpha_{-2} \dots)_2$.

Таким образом, необходимо найти коэффициенты $\alpha_{-1}, \alpha_{-2}, \dots$, входящие в запись числа x в двоичной системе счисления.

Умножим правую и левую части выражения (15) на 2. В результате в правой части получим

$$\alpha_{-1} + \alpha_{-2} \cdot 2^{-1} + \alpha_{-3} 2^{-2} + \dots \quad (16)$$

Целая часть здесь равна α_{-1} , она и даст нам старший коэффициент в разложении числа x по степеням 2.

Рассмотрим дробную часть (16): $x_1 = \alpha_{-2} \cdot 2^{-1} + \alpha_{-3} \cdot 2^{-2} + \dots$. Умножим x_1 на 2, целая часть числа $2x_1$ равна α_{-2} . Цифра α_{-2} и представляет собой второй коэффициент после запятой в двоичном представлении числа x . Этот процесс необходимо продолжать до тех пор, пока в правой части не получим нуль.

Примеры.

1. Перевести число $0,5625_{10}$ в двоичную систему счисления. Вычисления при этом удобно проводить по следующей схеме:

	0,	5625
		2
↓	1	1250
		2
	0	2500
		2
	0	5000
		2
	1	0000

Выписав цифры, стоящие слева от вертикальной черты в последовательности, определяемой стрелкой, получим

$$0,5625_{10} = 0,1001_2.$$

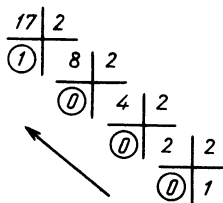
2. Перевести в двоичную систему счисления число $0,7_{10}$:

	0,	7
		2
↓	1	4
		2
	0	8
		2
	1	6
		2
	1	2
		2
	0	4
		2
	0	8
		2
	1	6
	

Очевидно, что процесс перевода числа $0,7_{10}$ может продолжаться бесконечно, давая все новые и новые знаки в его изображении в двоичной системе. Действительно, мы можем за четыре шага получить число $0,1011_2$, а за семь шагов число $0,1011001_2$, которое является более точным представлением числа $0,7_{10}$ в двоичной системе счисления, и т. д. Такой бесконечный процесс обрывают на некотором шаге, когда считают, что получена требуемая точность представления числа.

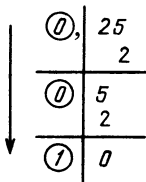
Перевод смешанных чисел, содержащих целую и дробную части, осуществляется в два этапа. Отдельно переводится целая часть, отдельно — дробная. В итоговой записи полученного числа, как обычно, целая часть отделяется от дробной запятой.

Пример. Перевести число $17,25_{10}$ в двоичную систему счисления. Для целой части получим



Отсюда $17_{10} = 10001_2$.

Для дробной части

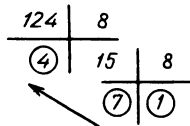


$0,25_{10} = 0,01_2$.

Таким образом, $17,25_{10} = 1001,01_2$.

Мы рассмотрели правила перевода из десятичной системы счисления в двоичную. Последовательность действий (алгоритм) останется той же, если вместо двоичной будет выступать какая-либо другая система счисления.

Пример. Перевести число $124,25_{10}$ в восьмеричную систему. Для целой части получим



Отсюда $124_{10} = 174_8$.

Переведем дробную часть числа:

$$\begin{array}{r|l} \textcircled{0}, & 25 \\ & 8 \\ \hline \textcircled{2} & 00 \end{array}$$

Следовательно, $0,25_{10} = 0,2_8$.

Таким образом,

$$124,25_{10} = 174,2_8.$$

При рассмотрении смешанных систем счисления мы отмечали, что в восьмеричной системе счисления наблюдается совпадение двоичной и двоично-восьмеричной записей одного и того же числа. Поэтому удобно при переводе числа из десятичной системы счисления в двоичную поступить так. Сначала перевести его в восьмеричную систему счисления, а затем каждую восьмеричную цифру записать двоичными знаками. Очевидно, что это потребует меньшего количества вычислений.

Пример. Перевести число 100_{10} в двоичную систему счисления:

$$\begin{array}{r|l} 100 & 8 \\ \hline \textcircled{4} & 12 & 8 \\ & \textcircled{4} & 7 \end{array}$$

Отсюда $100_{10} = 144_8$. Записав теперь последовательно каждый разряд числа 124_8 двоичными знаками, получим число 1010100_2 , которое и является двоичным представлением числа 100_{10} . В этом легко убедиться, если вычислить значение многочлена

$$1 \cdot 2^6 + 0 \cdot 2^5 + 1 \cdot 2^4 + 0 \cdot 2^3 + 1 \cdot 2^2 + 0 \cdot 2^1 + 0 \cdot 2^0.$$

Рассмотренные выше правила перевода чисел из одной системы счисления в другую достаточно громоздки и требуют большого количества вычислений, если речь идет о больших числах или требуется высокая точность их представления. Но этого не следует пугаться. Дело в том, что современные ЭВМ сами выполняют всю работу по специальным программам, незаметно для пользователя.

4.3. КАК ВИДИТ ОКРУЖАЮЩИЙ МИР КОМПЬЮТЕР (ПРЕДСТАВЛЕНИЕ ДАННЫХ В ЭВМ)

Данными принято называть информацию, вводимую в компьютер и выводимую из него.

Введенные в ЭВМ данные помещаются в запоминающее устройство. Следует отметить, что независимо от содержания информации, типа ее носителя и вида устройства ввода, которое «читает» с носителя данные, они записываются в память в одной и той же форме, а именно в виде последовательностей нулей и единиц.

Как уже говорилось в § 2.1, память машины разбита на ячейки.

В ячейке может храниться любая последовательность двоичных цифр фиксированной длины. Эту последовательность принято называть машинным словом. Длина машинного слова определяется конструкцией ЭВМ. Будем считать, что ячейка ЭВМ хранит машинные слова, состоящие из 24 двоичных символов.

Двоичную цифру, стоящую на каком-либо фиксированном месте ($0 \leq i \leq 23$) слова, называют i -м разрядом ячейки.

Что же представляют собой машинные слова?

Это могут быть числа, участвующие в вычислениях, символы, над которыми производятся какие-либо действия, а также инструкции самой ЭВМ, составляющие программу решения некоторой задачи.

В задачах вычислительного характера ЭВМ выполняет операции над словами, которые воспринимаются машиной как числа. Число для ЭВМ — это слово специального вида. Для задания, например, вещественного числа необходимо указать его знак, а также целую и дробную части. Поэтому ЭВМ сначала определяет в машинном слове эти характеристики, а потом правильно расшифровывает последовательность нулей и единиц. В современных ЭВМ используются две формы представления чисел: с фиксированной и плавающей запятой.

Представление чисел с фиксированной запятой. Ячейка памяти ЭВМ, содержащая число с фиксированной запятой, имеет знаковый и цифровые разряды. Разряды ячейки нумеруются, как правило, слева направо, причем каждому разряду ячейки соответствует один и тот же разряд числа.

Если мы говорим, что ЭВМ умеет выполнять действия над числами с фиксированной запятой, то это означает, что на этапе конструирования машины было определено, сколько и какие разряды машинного слова отведены под изображение целой и дробной частей числа. Другими словами, в представлении числа зафиксировано место запятой, отделяющей целую часть от дробной.

Например, если для изображения целой части числа отведено 10 двоичных знаков, а дробная часть изображается 13 двоичными знаками (один символ отводится для изображения знака числа, например знаку «+» соответствует 0, знаку «-» — единица), то число $10,25_{10} = 1010,01_2$ запишется в виде машинного слова, представленного на рис. 23.

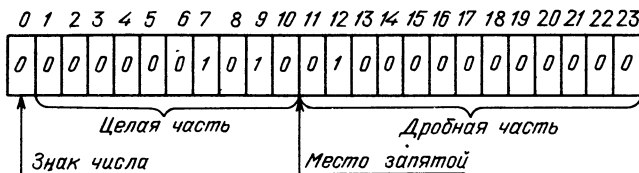


Рис. 23. Формат числа с фиксированной запятой

Можно условиться, что под изображение целой части числа отводятся разряды с 1 по 23 включительно, тогда в ячейке памяти будут храниться только целые числа. Легко видеть, что максимальному по модулю целому числу соответствует машинное слово, содержащее единицы в разрядах 1—23.

Выполнение арифметических операций над числами, представленными в форме с фиксированной запятой, реализуется достаточно просто, так как запятая, отделяющая дробную и целую части, находится у операндов на одном и том же месте. Следовательно, для получения результата достаточно осуществить поразрядно действия, определяемые видом выполняемой операции.

Простота реализации операций — главное достоинство представления чисел в форме с фиксированной запятой. Однако такая форма представления чисел имеет и существенный недостаток, заключающийся в ограниченном диапазоне чисел, с которыми ЭВМ может работать.

Действительно, если ячейка ЭВМ имеет 24 разряда и запятая, отделяющая целую часть от дробной, находится после 10-го разряда, то максимальное по абсолютной величине число, которое можно записать в ячейку памяти, имеет вид $111111111,111111111111_2$.

Можно убедиться, что оно не больше, чем 1026_{10} .

Наименьшее же по абсолютной величине отличное от нуля число равно $0000000000,0000000000001_2$. Очевидно, что оно соответствует десятичному числу $1 \cdot 2^{-13}$.

Таким образом, в ячейку памяти ЭВМ с фиксированной запятой можно записать любое число a , такое, что

$$1 \cdot 2^{-13} \leq a < 1026.$$

Как видно, диапазон изменения числа a не так уж велик, что создает значительные трудности при решении многих вычислительных задач, так как любое число, меньшее по абсолютной величине, чем левая граница допустимого диапазона, представляется в ЭВМ нулями. Если же в процессе работы машины понадобится поместить в ячейку памяти число, большее, чем правая граница диапазона, то в результате в ячейке окажется число, не имеющее с ним ничего общего. На практике, конечно, придумали методы борьбы с этим недостатком, но это требует значительных усилий со стороны программиста и анализа выполнения программы ЭВМ с целью обнаружения арифметических ошибок.

Представление чисел с плавающей запятой. Такое представление основано на том, что любое число $a \neq 0$ в системе счисления с основанием Q имеет вид

$$a = (\pm M)Q^{\pm p},$$

где M — правильная положительная дробь, называемая мантиссой числа a ; $\pm p$ — целое число, называемое порядком числа a . Представление числа в таком виде не является единственным. Например, десятичное число 125 можно записать следующим образом:

$$125_{10} = 12,5 \cdot 10^1 = 1,25 \cdot 10^2 = 0,125 \cdot 10^3 = 0,0125 \cdot 10^4 = \dots$$

Для выделения одного-единственного вида записи числа с плавающей запятой наложим следующее ограничение. Пусть в первом разряде мантиссы стоит отличная от нуля цифра, т. е. $1/a \leq M < 1$. Такое представление числа a называется нормализованным. Очевидно, что число $a = 125$ в нормализованном виде выглядит так:

$$a = 0,125 \cdot 10^3.$$

Здесь мантисса равна 0,125, а порядок равен 3. Все другие записи этого числа являются ненормализованными.

В двоичной нормализованной форме число a имеет вид $a = (\pm M)10^{\pm p}$, где M — двоичная мантисса, причем $1/2 \leq M < 1$, $10_2 = 2_{10}$ — основание системы: p — двоичный порядок.

Примеры.

1. $7_{10} = 10^{111} \cdot 0,111$ ($M = 0,111$; $p = 111$).
2. $-3_{10} = -2^2 \cdot 3/4 = -10^{10} \cdot 0,11$ ($M = 0,11$; $p = 10$).
3. $(3/16)_{10} = 2^{-2} \cdot 3/4 = 10^{-10} \cdot 0,11$ ($M = 0,11$; $p = -10$).

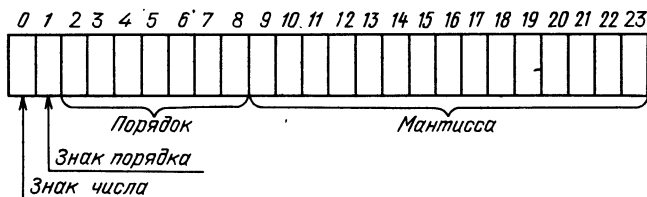


Рис. 24. Формат числа с плавающей запятой

Машинное слово длиной в 24 разряда в такой записи имеет вид, представленный на рис. 24. Нулевой разряд отведен для изображения знака числа, первый — знака порядка (знаку «+» соответствует 0, знаку «-» соответствует 1). Разряды содержат двоичный порядок числа, а разряды с 9 до 23 отведены для изображения двоичной мантиссы числа.

Так, например, число $7_{10} = 10^{111} \cdot 0,111$ запишется в ячейке ЭВМ так:

0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23
0	0	0	0	0	0	1	1	1	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0

Число $-3_{10} = -10^{10} \cdot 0,11$ будет представлено следующим образом:

0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23
1	0	0	0	0	0	0	1	0	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0

При такой форме записи чисел наименьшее по модулю отличное от нуля нормализованное число, представимое в ячейке памяти, равно

$$10^{-1111111} \cdot 0,1 = \frac{1}{2} \cdot 2^{-127} = 2^{-128} (M=0,1; p=-1111111).$$

Максимальное нормализованное число, представимое в ячейке ЭВМ, равно

$$10^{1111111} \cdot 0,1111111111111111 = \left(\frac{1}{2}\right)^{15} \cdot 2^{127} < 2^{113}.$$

Очевидно, что режим с плавающей запятой обеспечивает более широкий диапазон представления чисел в ЭВМ. Заметим, что если в результате каких-либо действий получим число, порядок которого больше, чем 1111111, то такое число воспринимается как ноль.

В качестве недостатков такого способа представления чисел по сравнению с представлением чисел с фиксированной запятой следует отметить увеличение количества символов для изображения числа (для изображения мантиссы, порядка и знака порядка, знака числа), а также в связи с этим усложнение процесса выполнения арифметических операций.

Так, сложение двух чисел происходит следующим образом. Сначала выравнивают порядки слагаемых, а затем складывают мантиссы чисел, после чего осуществляется нормализация результата.

Пример. Сложить два числа: $a_1 = 5,5_{10}$ и $a_2 = 3_{10}$. Так как $a_1 = 5,5_{10} = 101,1_2 = 10^{11} \cdot 0,1011$, $a_2 = 3_{10} = 11_2 = 10^{10} \cdot 0,11$, то порядок первого числа равен $p_1 = 11$, а второго $p_2 = 10$. Выравним порядки. Для второго слагаемого справедливо равенство $10^{10} \cdot 0,11 = 10^{11} \cdot 0,011$, откуда возьмем представление второго слагаемого с порядком $p_1 = p_2 = 11$.

Сложим теперь мантиссы чисел:

$$M_1 + M_2 = 0,1011 + 0,011 = 1,0001.$$

Таким образом,

$$a_1 + a_2 = 10^{11} \cdot 0,1011 + 10^{11} \cdot 0,011 = 10^{11} \cdot 1,0001.$$

Очевидно, что полученное в результате сложения число не является нормализованным, так как

$$M = 1,0001 > 1.$$

Нормализуем его и получим окончательный результат:

$$10^{100} \cdot 0,10001 = 8,5_{10}.$$

В правильности полученного результата легко убедиться, выполнив операцию сложения в привычной десятичной системе счисления.

Аналогичным образом осуществляется и операция вычитания, только в этом случае мантисса результата получается после вычитания мантиссы вычитаемого из мантиссы уменьшаемого.

Для выполнения операции умножения двух чисел их мантиссы перемножаются, а порядки складываются.

Пример. Умножить два числа: $a_1 = 7_{10}$ и $a_2 = -3_{10}$. Заменяем десятичные числа их двоичным эквивалентом:

$$7_{10} = 10^{11} \cdot 0,111 \text{ и } -3_{10} = -10^{10} \cdot 0,11,$$

откуда

$$M_1 = 0,111, p_1 = 11, M_2 = 0,11, p_2 = 10.$$

Мантисса произведения равна

$$M_1 M_2 = 0,11 \cdot 0,111 = 1,0101.$$

Соответственно порядок произведения

$$p_1 + p_2 = 11 + 10 = 101.$$

Таким образом, после нормализации и с учетом знаков сомножителей будем иметь

$$a_1 a_2 = 10^{110} \cdot 0,10101 = -21_{10}.$$

При выполнении операции деления мантиссу делимого делят на мантиссу делителя, а из порядка делимого вычитают порядок делителя.

Мы рассмотрели два способа представления чисел, отметили достоинства и недостатки каждого из них. Следует подчеркнуть тот факт, что в современных универсальных ЭВМ применяются, как правило, обе эти формы представления чисел.

Наряду с числами в качестве данных могут выступать и всевозможные символы (ведь мы уже отмечали, что машина умеет не только быстро считать, но и выполнять, например, всевозможные операции по обработке текстовой, графической информации).

Существует три типа символов: это цифры от 0 до 9, буквы, например, русского алфавита от А до Я или латинского от А до Z или какого-либо иного алфавита, а также специальные знаки (знаки арифметических операций, скобки, кавычки и т. д.). Каждому символу ставится в соответствие вполне определенный код — целое число. Код каждого символа представляет собой две цифры в шестнадцатеричной системе счисления (наиболее компактной из рассмотренных нами), например:

$$A \rightarrow C1, B \rightarrow C2, V \rightarrow C3 \text{ и т. д.}$$

Это означает, что символ А имеет код C1, а его двоичным представлением является 11000001.

Таблица соответствия символов кодам хранится в памяти ЭВМ,

а операции сопоставления символов и цифровых кодов осуществляются автоматически по специальным программам.

Так как машинное слово в рассматриваемом нами случае состоит из 24 разрядов, то в ячейку памяти можно записать один из 2^{24} различных символов. Такое количество различных символов вряд ли можно получить, если даже объединить все алфавиты всех известных в мире языков. На практике обычно для изображения одного символа отводят 8 разрядов машинного слова, так называемый байт. В этом случае мы можем иметь, $2^8 = 256$ различных символов. Этого вполне достаточно для решения практических задач. В одной ячейке машинного слова хранится при этом 3 символа.

В памяти ЭВМ хранятся не только числа, символы, но и тексты программ решения задач. Программа состоит из набора отдельных команд. Команда определяет действия ЭВМ на каждом шаге ее работы. Следовательно, она должна содержать информацию о том, над какими объектами (операндами) и какие операции должна выполнить машина, что нужно сделать с полученным результатом, а также определить следующую для исполнения команду.

В ЭВМ вся эта информация представлена машинным словом — последовательностью двоичных цифр, расположенных в ячейке памяти.

Для того чтобы ЭВМ могла из последовательности цифр извлечь необходимую для работы информацию, в ее конструкции предусмотрена возможность анализа определенных групп цифр в такой записи. Каждая группа цифр несет соответствующую информацию: одна указывает вид операции, другая дает информацию об операндах данной операции (например, указывает их адреса в памяти ЭВМ).

Каждой операции, которую может выполнять данная ЭВМ, присваивается так называемый код операции — некоторое целое число. Соответствие операции и кода устанавливается с помощью специальной таблицы, хранимой в памяти ЭВМ, причем делается это автоматически самой машиной.

Для изображения кода операции в машинном слове отводятся определенные разряды, например с 0 до 7, как это представлено на рис. 25. Остальные разряды, с 8 по 23, составляют так называемую адресную часть, в которой находятся адреса операндов,

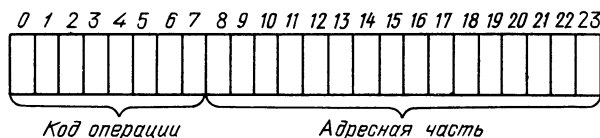


Рис. 25. Формат команды

участвующих в операции. В процессе работы ЭВМ извлекает слово из ячейки памяти, расшифровывает его и включает выполнение соответствующей команды.

Следует еще раз подчеркнуть, что информация, независимо от ее содержания, представляется в памяти в виде машинных слов, т. е. последовательностей нулей и единиц. По виду машинного слова невозможно определить, что это — число или команда. Более того, одна и та же последовательность цифр может выступать и как число, и как некоторая команда. Все зависит от того, куда данное машинное слово попадает в процессе работы ЭВМ. Если оно попадает в арифметическое устройство, то рассматривается как операнд некоторой операции, а если в устройство управления машины, то расшифровывается как соответствующая команда ЭВМ.

4.4. АЛГОРИТМЫ И АЛГОРИТМИЧЕСКИЕ ЯЗЫКИ

Каждый из нас в своей повседневной жизни сталкивается с огромным количеством разнообразных по своей сложности задач. Многие из них настолько привычны и естественны, что мы не задумываемся над способом их решения. Как правило, нам они хорошо известны и, более того, сформулированы в виде специальных инструкций (например, как пользоваться бытовыми электроприборами и т. д.). Для того, чтобы подогреть чайник на газовой плите, достаточно последовательно выполнить следующие действия: налить в чайник воды, зажечь спичку, поднести ее к конфорке, повернуть вентиль конфорки, убедившись, что газ загорелся, поставить чайник на плиту. Если теперь попытаться сформулировать эту последовательность действий в виде инструкции, то она могла бы выглядеть, например, так:

1. Наполнить чайник водой.
2. Зажечь спичку.
3. Поднести спичку к конфорке.
4. Открыть вентиль.
5. Убедившись, что газ загорелся, поставить на плиту чайник.

Мы сформулировали последовательность действий (инструкцию), следуя которой любой человек сможет на газовой плите вскипятить воду в чайнике.

Рассмотрим еще одну часто встречающуюся ситуацию: пешеходу нужно перейти улицу с двухсторонним движением. Когда он пользуется переходом на перекрестке улиц, где движение регулируется светофором, действия пешехода предельно просты. Нужно убедиться в том, что на светофоре горит зеленый свет, и перейти улицу. Гораздо более сложную последовательность действий необходимо совершить, когда нет светофора. В этом случае пешеход должен вести себя, например, следуя-

щим образом. Посмотреть налево и, убедившись в том, что опасности столкновения с автомобилем нет, пройти до середины улицы. Очевидно, что все автомобили, следующие слева направо относительно пешехода, будут проезжать за его спиной и не представляют для него опасности. Находясь на осевой линии дороги, пешеходу теперь следует посмотреть направо. Если автомобилей вблизи нет, то дорогу можно переходить.

Более четко действия пешехода можно записать так:

1. Посмотреть налево, убедиться в отсутствии опасности.
2. Пройти до середины дороги.
3. Посмотреть направо, убедиться в отсутствии опасности.
4. Пересечь оставшуюся часть дороги.

Еще пример. При выполнении домашнего задания по математике школьнику необходимо вычислить длину окружности L для заданного значения радиуса R . При этом он пользуется известной формулой $L = 2\pi R$, которая определяет такую последовательность арифметических действий:

1. Перемножить числа 2 и $\pi = 3,14$.
2. Полученный результат умножить на значение R .

Вы, читатель, сами можете придумать огромное количество примеров, где для достижения цели определен набор правил или инструкций, приводящих к определенному результату. Такие наборы правил называют алгоритмами. А именно, алгоритм — это описание таких действий, последовательно выполняя которые можно получить решение задачи. Каждое отдельное действие называют шагом алгоритма (например, «зажечь спичку», «дойти до середины дороги», «перемножить числа 2 и π » и т. д.). Последовательность шагов алгоритма строго фиксирована. При этом исполнитель должен уметь выполнять каждый шаг алгоритма.

Каждый алгоритм подразумевает наличие исходных данных и ожидаемого результата. Например, для алгоритма вычисления значения длины окружности L в качестве исходных данных выступают константы 2, π и значение радиуса R . Результатом является число L , равное длине окружности при заданном значении R . Многие алгоритмы обладают тем свойством, что остаются правильными для различных наборов исходных данных. Например, алгоритм вычисления значения длины окружности не зависит от конкретного значения радиуса R . При различных R мы будем получать различные значения L , но последовательность действий при этом будет совершенно одинаковой.

Алгоритм перехода улиц остается одинаковым как для каждого человека, так и для перекрестков улиц в различных районах и даже в других городах. Такое свойство алгоритма использовать в качестве исходных данных различные наборы называется *массовостью*. Это свойство позволяет один и тот же алгоритм использовать для решения задачи при различных начальных данных. Следует иметь в виду, что для каждого алгоритма

существует некоторое множество объектов, которые допустимы в качестве исходных данных алгоритма. Например, в алгоритме вычисления длины окружности в качестве исходного данного для радиуса окружности может быть взято любое действительное положительное число.

Если мы рассмотрим алгоритм деления действительных чисел, то, как известно, в качестве делимого может выступать любое число, а в качестве делителя любое число, кроме нуля, так как операция деления на нуль не определена.

Всегда ли можно получить результат, применяя алгоритм к некоторому набору из множеств допустимых исходных данных? Оказывается, что не всегда. Действительно, рассмотрим алгоритм деления двух чисел и в качестве делимого возьмем число 5, а в качестве делителя — число 3. Очевидно, что оба они являются допустимыми для нашего алгоритма. Однако совершенно очевидно, что процесс деления никогда не закончится, так как результатом будет являться бесконечная десятичная дробь 1,6666... Об алгоритмах такого рода говорят, что они потенциально осуществимы, но состоят из бесконечного числа шагов.

В связи с этим надо говорить о *конечности* алгоритма. Смысл этого понятия заключается в том, чтобы алгоритм давал ответ через конечное число шагов и за конечное время.

Вновь вернемся к алгоритму деления и прервем его выполнение на каком-то шаге. Очевидно, что в результате получим число, которое, вообще говоря, не является частным от деления 5 на 3. Однако полученную таким образом величину приближенно можно принять в качестве результата. Более того, увеличивая число шагов, можно достичь любой наперед заданной точности получаемого результата. Вообще говоря, в данном примере мы используем несколько иной алгоритм деления, который предусматривает обрыв алгоритмического процесса при достижении требуемой точности результата. Во многих практических задачах требуется не потенциальная, а реальная выполнимость алгоритма, поэтому свойство конечности приобретает особенно важный смысл.

Следующей важной характеристикой алгоритма является *понятность*. Данное свойство означает, что исполнителю ясно, каким образом выполняется алгоритм, причем любой исполнитель однозначно понимает смысл последовательности действий, составляющих алгоритм. Например, если в качестве исполнителей алгоритма вычисления длины окружности выступают два школьника, то предполагается, что каждый из них умеет выполнять операцию умножения чисел. Тогда последовательность действий, задающая алгоритм, для обоих школьников будет одинаково понятна.

Еще одним важным свойством алгоритма является *однозначность*. Смысл заключается в том, что если алгоритм многократно

применять к одному и тому же набору исходных данных, то всегда будет получаться один и тот же результат. Из свойства однозначности вытекает независимость решения задачи от индивидуальных особенностей исполнителя.

С учетом выше рассмотренных свойств алгоритма определим алгоритм как систему правил, которая сформулирована на языке, понятном исполнителю, определяет процесс перехода от допустимых исходных данных к искомому результату и обладает свойствами массовости, конечности, понятности, однозначности.

Для одной и той же задачи может существовать несколько различных алгоритмов ее решения. Например, корни квадратного уравнения

$$x^2 + bx + c = 0$$

можно найти по формуле

$$x_{1,2} = -b/2 \pm \sqrt{(b/2)^2 - c},$$

а можно и по теореме Виета, которая устанавливает, как известно, связь между корнями уравнения x_1 , x_2 и его коэффициентами в виде

$$x_1 x_2 = c, \quad x_1 + x_2 = -b.$$

А если у задачи существует несколько алгоритмов ее решения, то какому из них следует отдать предпочтение? Четких и однозначных критериев для сравнения алгоритмов нет, но вполне естественно выбрать тот из них, который требует для своего выполнения меньше усилий со стороны исполнителя и быстрее приводит к искомому решению. С точки зрения ЭВМ наилучшими являются алгоритмы, не требующие большого объема памяти (как оперативной, так и ВЗУ) для хранения исходных данных, программных текстов, промежуточных и окончательных результатов, и исполнение которых занимает наименьшее время. Для вычислительных алгоритмов в качестве характеристики выступает также точность получаемого результата.

Довольно часто решаемая задача может быть сведена к некоторому набору более простых и, возможно, хорошо изученных подзадач. Например, для построения на плоскости графика функции $y = ax^2 + bx + c$ при заданных коэффициентах a , b , c надо отыскать точки пересечения кривой с осью x . Для определения этих точек достаточно найти корни уравнения $ax^2 + bx + c = 0$, а эта задача, в свою очередь, требует вычисления дискриминанта $D = b^2 - 4ac$, что невозможно без знания алгоритмов выполнения арифметических операций. Если для каждой из подзадач будут построены соответствующие алгоритмы, то алгоритм решения первоначально поставленной задачи можно теперь сконструировать из алгоритмов решения подзадач. Такой подход к построению алгоритмов из алгоритмов решения подзадач получил

широкое распространение. Его успех определяется наличием ряда достоинств.

Во-первых, конструируя алгоритм, можно работать с более крупными понятиями и объектами, что позволяет не затушевывать деталями структуру алгоритма, а оставлять ее ясной, прозрачной.

Во-вторых, алгоритм основной задачи может несколько раз обращаться к решению одной и той же подзадачи. Например, при отыскании корней квадратного уравнения по формулам $x_1 = (-b + \sqrt{b^2 - 4ac}) / 2a$ и $x_2 = (-b - \sqrt{b^2 - 4ac}) / 2a$ дискриминант $D = b^2 - 4ac$ необходимо вычислять для нахождения как x_1 , так и x_2 . При детализации шагов алгоритма можно было бы дважды переписывать последовательность действий для вычисления значения D . Но зачем?

В-третьих, повышается качество алгоритмов, так как в этом случае разработку алгоритмов решения подзадач можно поручить специалистам, обладающим высокой квалификацией в области, к которой относится подзадача.

В каждой области науки и техники существует и накапливается свой фонд алгоритмов, в этом можно убедиться, если заглянуть во всевозможные справочники. Пополнение этого фонда — задача специалистов каждой отдельной области человеческих знаний.

Каким образом нужно представить алгоритм, чтобы исполнитель, в частности ЭВМ, мог его использовать?

Очевидно, что в первую очередь нужно записать алгоритм решения задачи на языке, понятном исполнителю. Причем сделать это необходимо с учетом свойств и особенностей исполнителя, а также характера всех объектов, с которым работает алгоритм. Степень детализации описания алгоритма должна быть вполне достаточной для того, чтобы исполнитель понимал его однозначно.

Какие средства при этом можно использовать?

Для описания алгоритмических процессов можно, конечно, использовать обычный разговорный язык (как, впрочем, мы уже поступали, описывая простейшие алгоритмы). Однако такая запись для сложных алгоритмов оказывается, как правило, очень громоздкой, а главное — недостаточно четкой и строгой.

Для ЭВМ алгоритм можно описывать на внутреннем языке машины, т. е. в виде последовательности команд, тем более, что такая запись является конечной целью программирования. Этот способ использовался на первых ЭВМ. В этом случае нужно алгоритм решения задачи записать в виде последовательности элементарных операций (сложение, вычитание, умножение, деление и т. д.). Легко видеть, что если задача достаточно сложная, то и запись алгоритма ее решения будет громоздкой, труднообозримой и может содержать значительное число ошибок. Да и обмениваться алгоритмами в этом случае довольно трудно,

ведь каждая ЭВМ имеет ряд присущих только ей особенностей, которые в значительной мере влияют на возможность эффективного использования алгоритмов.

Таким образом, нужны языки записи алгоритмов, т. е. некоторые формализованные средства, правила представления алгоритмов, которые позволили бы наглядно, с использованием общепринятых символов записывать алгоритмы. Язык этот должен быть достаточно гибким, чтобы описывать сложные алгоритмы, используя при этом минимум изобразительных средств, позволять человеку отчетливо, но в то же время сжато выразить свою мысль. Такой язык должен быть удобным средством обмена информацией между людьми, а также выполнять функции посредника между человеком — составителем алгоритма — и ЭВМ — его исполнителем.

Для задания алгоритмов используются следующие способы:

1. Описание алгоритма на профессиональном языке той области знаний, которой принадлежит задача: с использованием математической символики, терминов и понятий, характерных для конкретной области знаний.

2. Графическое представление алгоритма в виде совокупности его фрагментов с указанием связей между ними.

3. Описание алгоритма на языке программирования. Текст, представляющий запись алгоритма на языке программирования, называют программой.

Рассмотрим еще раз алгоритм вычисления длины окружности $L = 2\pi R$ по заданному значению радиуса R . Запишем его так:

1. Ввести числа 2, π , R .
2. Перемножить 2 и π , результат обозначить A .
3. Перемножить A и R , результат обозначить L .
4. Остановить процесс.

Будем считать, что пункты алгоритма выполняются последовательно в порядке возрастания их номеров. Каждый пункт определяет вполне конкретные действия исполнителя. На первом шаге для исполнителя определяются те исходные данные, с которыми ему предстоит работать: константы 2, π и конкретное значение величины R .

На втором шаге алгоритма вычисляется произведение чисел 2 и π , которые входят в формулу для L . Так как это значение на следующем шаге алгоритма выступает в качестве сомножителя, его нужно сохранить. С этой целью вводится переменная A , значение которой будет равно 2π . На третьем шаге алгоритма значение A перемножается с R , полученный результат обозначается переменной L . Пункт 4 служит указанием исполнителю об окончании алгоритмического процесса.

При записи данного алгоритма в п. 2 мы воспользовались следующим приемом. Ввели буквенное обозначение для промежуточной величины, которое использовали затем при описании

вычисления на очередном шаге алгоритма. Такое действие (которое называют присваиванием) принято обозначать так: «:=». Запись $A := 2 \times \pi$ означает, что необходимо вычислить произведение 2π и обозначить его через A . Следовательно, пункты 2 и 3 алгоритма можно записать так:

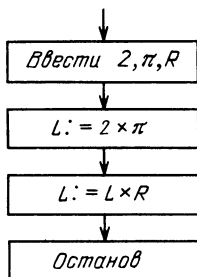
2. $A := 2 \times \pi$
3. $L := A \times R$

Можно сократить число переменных для обозначения промежуточных результатов и записать алгоритм следующим образом:

1. Ввести $2, \pi, R$.
2. $L := 2 \times \pi$.
3. $L := L \times R$.
4. Остановить процесс.

Необычным в этой записи является то, что в п.3 переменная L находится одновременно в левой и правой частях. Такая запись допустима, так как после выполнения п. 2 переменная L имеет значение $2 \times \pi$, а в п. 3 сначала вычисляется значение L , а затем уже это значение обозначается буквой L . Таким образом, переменная L получает новое значение, а старое значение L , полученное в п. 2, утрачивается.

Отметим одну важную особенность рассмотренного алгоритма. Здесь все действия выполняются последовательно одно за другим не более одного раза. Алгоритмы такого типа носят название линейных. На языке блок-схем они обычно изображаются в виде прямоугольников, соединенных стрелками, которые задают последовательность выполнения операций, определяемых каждым прямоугольником-блоком. Для нашего алгоритма будем иметь



Но линейных алгоритмов в природе не так много. Они связаны, как правило, с решением достаточно простых задач. Большое значение имеют так называемые разветвляющиеся алгоритмы. Они содержат условия, которые определяют последовательность действий при их выполнении. Рассмотрим, например, задачу

вычисления a — абсолютного значения действительного числа a . Известно, что для нахождения a достаточно сравнить a с нулем. Если a — неотрицательное число, то $|a| = a$, а если a — отрицательное число, то $|a| = -a$.

Запишем алгоритм решения этой задачи:

1. Ввести a .
2. Сравнить a с нулем:
если $a \geq 0$, то перейти к п. 4,
если $a < 0$, то перейти к п. 3.
3. $|a| := -a$, перейти к п. 5.
4. $|a| := a$.
5. Остановить процесс.

В п. 2 алгоритма число a сравнивается с нулем и определяется, является ли число a отрицательным или положительным. Если $a \geq 0$, то выполняется п. 4, где переменной $|a|$ присваивается значение самого числа a . Затем выполняется п. 5, который завершает выполнение алгоритма. Отметим, что при $a \geq 0$ п. 3 не выполняется вообще. Если $a < 0$, тогда после п. 2 выполняется п. 3, переменная $|a|$ получает значение $-a$, после чего процесс останавливается. В этом случае не выполнялся п. 4 алгоритма.

Мы использовали инструкцию вида «если..., то ...», благодаря чему оказалось возможным выполнить различные действия в зависимости от знака числа. Такие инструкции называются условными, они и позволяют строить разветвления в алгоритме.

Общий вид инструкции, реализующей разветвление, таков: «Если выполняется некоторое условие P , то делай A , иначе делай B ». Здесь A и B — последовательности действий, причем A выполняется в случае, когда выполнено условие P , если же P не выполнено, то вместо A выполняется B .

В качестве условия может быть использовано любое утверждение, относительно которого в каждом конкретном случае исполнитель может сказать, выполнено оно или нет.

Перепишем наш алгоритм с использованием новых инструкций:

1. Ввести a .
2. Если $a < 0$, то $|a| := -a$, иначе $|a| := a$.
3. Остановить процесс.

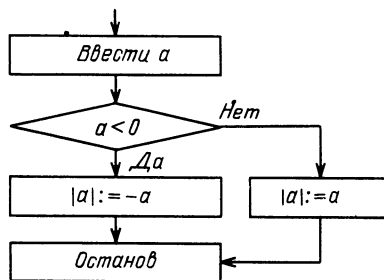
Существует и такая форма записи инструкции условия: «Если P , то A ». Она означает следующее: если условие P соблюдается, то выполняются действия, определяемые A ; если P не соблюдено, то действия, определяемые A , пропускаются, а выполняются те, что стоят в алгоритме вслед за A .

Для нашего примера получим:

1. Ввести a .
2. Если $a < 0$, то $|a| := -a$, перейти к п. 4.
3. $|a| := a$.
4. Остановить процесс.

На языке блок-схем действия с условиями обычно заключаются

в ромб, из которого выходят две стрелки. Одна соответствует тому, что условие выполнено (ее, как правило, помечают символом «да»), другая — тому, что условие не выполнено (ее помечают символом «нет»). На языке блок-схем алгоритм решения задачи имеет следующий вид:



Известно много алгоритмов, у которых некоторые шаги алгоритмического процесса повторяются многократно. Про такие алгоритмы говорят, что они содержат цикл.

Рассмотрим, например, алгоритм подсчета значения произведения десяти чисел $a_1, a_2, a_3, a_4, a_5, a_6, a_7, a_8, a_9, a_{10}$.

1. Ввести a_1, a_2, \dots, a_{10} .

2. $S := a_1$.

3. $S := S \times a_2$.

4. $S := S \times a_3$.

...

11. $S := S \times a_{10}$.

12. Остановить процесс.

Здесь шаги 3—11 выполняют одинаковые действия, а именно перемножают значение S , полученное на предыдущем шаге, с очередным числом a_i , i изменяется при этом от 2 до 10. Как только для операции умножения в качестве сомножителя будет взято число a_{10} (это произойдет на 11-м шаге алгоритма), мы получим окончательный результат.

Для записи такого рода алгоритмов используются циклические инструкции. Выглядят они так: «Пока P , делать Q ». Здесь P — некоторое условие, Q — последовательность действий, которая выполняется столько раз, сколько необходимо для того, чтобы условие P перестало выполняться.

Наш алгоритм с использованием циклической инструкции будет выглядеть так:

1. Ввести $a_1, a_2, a_3, a_4, a_5, a_6, a_7, a_8, a_9, a_{10}$.

2. $S := a_1$.

3. $i := 2$.

4. $S := S \times a_i$.

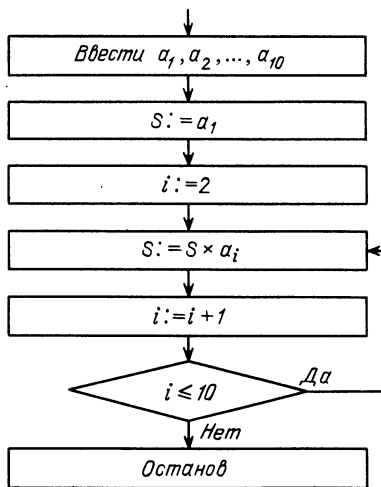
5. $i := i + 1$.

6. Если $i \leq 10$, то перейти к п. 4.

7. Остановить процесс.

Поясним данную запись. В п. 2 S принимает значение a_1 — первого сомножителя. Индекс i используется для идентификации очередного сомножителя a_i . В п. 3 i получает начальное значение, равное 2. В п. 4 перемножаются первые два сомножителя, после чего переменная S получает значение $a_1 a_2$. Теперь необходимо выбрать следующий сомножитель, т. е. a_3 . Для этого переменной i в п. 5 присваивается значение $i + 1$, и она становится равной 3. В п. 6 i сравнивается с 10, тем самым проверяется, все ли сомножители участвовали в операции умножения. Так как $3 < 10$, то осуществляется переход к п. 4, после выполнения которого S будет равно $a_1 a_2 a_3$. Затем i снова увеличивается на единицу, сравнивается с числом 10 и т. д. Пункты 4—6 будут повторяться до тех пор, пока после очередного увеличения i на единицу, оно не станет равным 11. Тем самым нарушится условие $i \leq 10$. К этому моменту значение S и будет равно произведению всех десяти сомножителей a_i .

В виде блок-схемы данный алгоритм будет выглядеть так:



Мы рассмотрели три типа структур алгоритмов, а именно линейную, разветвляющуюся и циклическую. Следует отметить, что любой алгоритм является либо линейным, либо разветвляющимся, либо циклическим, либо комбинацией этих трех структур.

Программирование для ЭВМ — это перевод алгоритма с описательного языка или языка блок-схем на язык программирования для данной ЭВМ. В конкретных языках каждой из рассмотренных структур соответствуют вполне определенные операторы. Например, линейный алгоритм реализуется набором операторов

присваивания, для разветвляющихся алгоритмов определены так называемые условные операторы, для организации циклических процессов используются операторы цикла.

Для каждого конкретного языка фиксируется свой набор операторов, которые указывают на выполнение определенных действий, при этом фиксируются правила записи каждого из операторов. Это позволяет однозначно понимать записи для каждого этапа алгоритмического процесса.

4.5. ЭКРАННЫЙ РЕДАКТОР «SCREEN»

Как происходит процесс написания программ для ЭВМ? Предположим, что вы, читатель, познакомились с каким-нибудь языком программирования и решили создать первую программу. А может и того проще — решили использовать персональный компьютер в качестве записной книжки — собрать фамилии, имена своих друзей, их адреса и телефоны. Текст программы, список друзей написаны на листе бумаги, а они должны попасть в ЭВМ. Но как, каким образом?

В настоящее время такую информацию вводят в память машины в основном через дисплей. Специальные программы — редакторы — позволяют не только набирать текст на экране дисплея, но и редактировать его, т. е. искать и исправлять опечатки, переставлять местами абзацы и т. д.

Так как широкое распространение в стране получили ЭВМ серии ДВК (дисплейно-вычислительный комплекс), то ниже мы опишем правила работы с экранным редактором «Screen», позволяющим достаточно удобно создавать различные тексты, в том числе и программные.

Напомним (см. § 2.2), что каждый отдельный текст информации, подготовленный к обработке на ЭВМ и записанный в одном из запоминающих устройств (диск, лента, дискета и т. д.), принято называть файлом. Каждый файл имеет имя и тип.

При работе с дисплеем в определенных режимах на экране появляется изображение маленького прямоугольника, которое называется курсором. Курсор может передвигаться по экрану в горизонтальном (вправо, влево) и вертикальном (вверх, вниз) направлениях.

Итак, вы пришли в класс, где установлен ДВК-2М (см. рис. 15), заняли место и приготовились к работе. Дежурный по классу включил машину, на дисковод (см. рис. 8) с номером 0 установил дискету с операционной системой (ОС) ДВК, редактором «Screen» и загрузил ОС.

На дисковом с номером 1 установите свою дискету. Для простоты будем считать, что обе дискеты имеют диаметр 133 мм. Когда ОС загружена и ЭВМ готова к работе, информация о готовности поступает в виде точки на экране дисплея (отсутствие точки на

экране означает, что произошел сбой при загрузке ОС и надо обратиться к дежурному по классу).

Наберите на клавиатуре команду

RUN «SCREEN»

и нажмите на клавишу «ВК».

Через несколько секунд редактор обратится к вам с вопросом, который вы прочитаете на экране дисплея:

ВХОДНОЙ ФАЙЛ

Как понимать вопрос?

1. Будете ли вы создавать новый файл или будете продолжать работать над ранее записанным на дискету файлом?

2. Если будете продолжать работу, то где, на каком дисковом диске находится дискета с этим файлом, какие у него имя и тип?

Как отвечать на этот вопрос?

1. Если создаете новый файл, то просто нажмите клавишу «ВК».

2. Если будете продолжать работу, то сначала надо указать имя дискового, а затем через двоеточие — имя и тип файла. Например:

MX1:PEK.TXT

Эта запись означает, что дискета диаметром 133 мм «М» (если дискета имеет диаметр 203 мм, то ее обозначают «Д») установлена на дисковом «Х» с номером 1, на этой дискете записан файл PEK.TXT, над которым будет продолжена работа.

Для дискеты диаметром 203 мм соответствующий ответ выглядит так:

DX1:PEK.TXT

Во всех случаях после сообщения редактору адреса входного файла надо нажать клавишу «ВК». На этом диалог не заканчивается. Редактор опять задает вопрос, который вы читаете на экране дисплея:

ВЫХОДНОЙ ФАЙЛ

Как понимать вопрос? Куда, на какую дискету записать после окончания сеанса результат вашей работы и как назвать (имя, тип) итоговый файл?

Как отвечать на вопрос? Наберите на клавиатуре номер дискового (МХК или ДХК, где К — номер дискового), через двоеточие укажите имя и тип создаваемого файла. Например:

MX1:TOP.TXT или DX1:TOP.TXT

Нажатием клавиши «ВК» сообщите редактору, что ответ готов.

После того как указаны имя и тип входного файла, редактор задаст еще один последний вопрос. На экране дисплея вы прочтете:

СКОЛЬКО СТРОК ПРОПУСТИТЬ

Как понимать вопрос? Если вы после перерыва возвращаетесь к написанию некоторого текста, то, как правило, обращаетесь к незаконченной странице, к тому месту, где прервали свою работу, а предыдущие страницы откладываете в сторону. Так и здесь редактор предлагает свои услуги: «Не хотите ли вы «перелистнуть» в выходном файле несколько, пусть N , первых строк, а на экран вывести текст с $(N+1)$ -й строки?» Но тогда, и не забывая об этом, первые N строк станут недоступны для редактирования и на экране не высветятся.

Если вы работаете с очень большими файлами, то из-за ограниченности памяти, обеспечивающей редакторскую работу дисплея (примерно 250 строк), редактирование текста осуществляется частями. Разбиение на части можно осуществлять указанием числа ранее отредактированных строк.

Как отвечать на вопрос?

1. Надо набрать на клавиатуре число N строк, которые можно скопировать в выходной файл, и нажать клавишу «ВК».

2. Если $N=0$, то достаточно нажать только клавишу «ВК».

Получив ответы на все свои вопросы, редактор очистит экран дисплея. Далее если продолжается работа над ранее созданным файлом и указано число N строк, которые надо пропустить, то на экране дисплея появится текст входного файла, начиная с $(N+1)$ -й строки. Если же файл необходимо создать заново (т. е. когда на вопрос редактора об имени входного файла вы нажали клавишу «ВК»), то после очистки экрана в левом верхнем углу появится курсор — сообщение о готовности к работе.

Для набора текста на русском языке перед вводом первой буквы нажмите клавишу «РУС» в левом нижнем углу клавиатуры, а перед вводом латинского текста нажмите клавишу «ЛАТ» в правой нижней части клавиатуры (рис. 26). Для записи прописных (больших) букв нажмите клавишу «ВР» (на клавиатуре слева внизу), при переходе на строчные (мелкие) буквы нажмите клавишу «НР» (на клавиатуре справа внизу). В отсутствие указаний редактор пишет латинские тексты.

Окончив работу, надо записать созданный текст на дискету. Для этого надо сообщить редактору о конце сеанса следующим образом:

1. Сначала нажать клавишу «↓» в правой части клавиатуры.

2. Потом нажать клавишу «0» — окончание.

Нажимать надо последовательно, а не одновременно. Редактор «Screen», получив такое сообщение, запишет информацию по

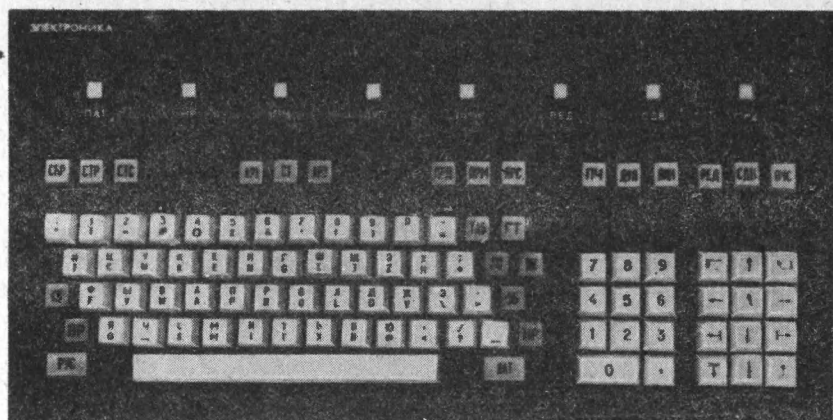














Рис. 26. Клавиатура персональной ЭВМ ДВК-2М

указанному выходному адресу и сообщит вам об этом точкой на очищенном от текста экране.

Выше мы построили алгоритм набора текста файла. А как редактировать готовый текст — исправлять опечатки, вставлять пропущенные слова, предложения и т. д.? Для этого нам надо предварительно познакомиться с правой частью клавиатуры (клавиши со стрелками).

Для упрощения описаний клавиш со стрелками введем следующие обозначения:

	1	2	3
4			
3			
2			
1			

Далее будем говорить «клавиша $N_1.N_2$ » или просто « $N_1.N_2$ », понимая под этим клавишу, расположенную в N_1 -й строке и N_2 -м столбце.

Клавиши управления курсором

Влево «3.1»	При нажатии этой клавиши: 1) курсор перемещается влево; 2) если курсор расположен в левой верхней позиции экрана, то он переходит в правую нижнюю, если там есть ранее записанный текст; если текста нет, то исчезает; 3) из первой позиции строки курсор перемещается в последнюю позицию предшествующей строки
Вправо «3.3»	При нажатии этой клавиши курсор перемещается: 1) вправо; 2) из последней позиции строки в первую позицию следующей за ней строки; 3) из самой правой нижней позиции экрана в левую верхнюю позицию
Вверх «4.2»	При нажатии клавиши: 1) курсор двигается вверх по экрану; 2) если курсор находится в самой первой строке текста, то раздается звуковой сигнал, положение курсора и изображение на экране не меняются; 3) если курсор находится в самой верхней строке экрана, то текст опускается по экрану вниз, при этом на месте верхней строки появляется изображение ранее невидимой, предшествующей ей строки
Вниз «2.2»	При нажатии клавиши: 1) курсор перемещается вниз по экрану; 2) если курсор находится в последней строке текста, то раздается звуковой сигнал и ничего не изменяется; 3) если курсор находится в самой нижней строке экрана, то текст поднимается по экрану вверх, а на месте нижней строки появляется ранее невидимая, следующая за ней строка
Домой «3.2»	При нажатии клавиши курсор перемещается в левую верхнюю позицию экрана
Перевод строки «4.1» и «4.3»	Клавиши работают аналогично клавишам возврата каретки пишущей машинки, устанавливают курсор в первую позицию следующей строки
Сдвиг текста в строке «2.1»	При нажатии клавиши символ, помеченный курсором, пропадает, а на его место перемещается следующий символ. Весь текст строки, следующий за курсором, передвигается на позицию влево, в последнюю позицию строки помещается пробел
Раздвижка текста в строке «2.3»	При нажатии клавиши символ, помеченный курсором, заменяется на пробел, а весь текст строки, начиная с символа над курсором, перемещается на позицию вправо. Последний символ строки пропадает
Вставка строк «1.1»	При нажатии клавиши весь текст, начиная со строки, помеченной курсором, опускается на строку вниз. На освободившемся месте появляется новая чистая строка. Положение курсора не изменяется
Удаление строк «1.3»	При нажатии клавиши строка, помеченная курсором, пропадает. Ее место занимает следующая за ней строка, а весь текст сдвигается вверх

Неоднократное нажатие или удержание перечисленных выше клавиш, кроме «3.2», вызывает многократное повторение соответствующих операций.

Мы обошли вниманием только одну клавишу «1.2» — ее можно назвать «командной». Выше были описаны локальные процедуры над текстом или его фрагментами, а нам уже встречалась команда редактору «Окончить работу», начинающая с «1.2».

Именно клавиша «1.2» подсказывает редактору, что далее последует команда. Ниже приведены команды, которые понимает редактор «Screen» (запись «А», «В» означает, что сначала надо нажать клавишу «А» и отпустить, а потом нажать клавишу «В»):

«1.2», «Н»	Выдать на экран начало текста («Начало»)
«1.2», «К»	Выдать на экран конец текста («Конец»)
«1.2», «Р»	Дублировать предыдущую строку («Размножить»)
«1.2», «П»	Переместить строки («Переместить»)
«1.2», «Д»	Дублировать группу строк («Дублировать»)
«1.2», «У»	Удалить строку («Удалить»)
«1.2», «И»	Контекстный поиск («Искать»)
«1.2», «Е»	Продолжение контекстного поиска («Еще»)
«1.2», «З»	Контекстная замена («Заменить»)
«1.2», «С»	Сохранить предыдущую часть текста и выдать на экран следующую часть («Сохранить»)
«1.2», «О»	Окончание работы с редактором («Окончить»)

Редактирование текста. Какие операции включает в себя процесс редактирования написанного текста? Попробуем некоторые из них перечислить:

1. Удаление и замена отдельных символов.
2. Удаление и вставка слов.
3. Удаление и вставка строк текста и целых абзацев.
4. Поиск и перемещение отдельных частей текста.

Как осуществляются эти операции в «Screen»?

Удаление и замена отдельных символов. Если необходимо удалить символ, то установите при помощи клавиш «3.1», «3.3», «2.2.» и «4.2» курсор в строку, в которой расположен удаляемый символ, в следующую за ним позицию. При нажатии клавиши «ЗБ» курсор перемещается на одну позицию влево и символ заменяется на пробел. Если необходимо удалить несколько рядом расположенных символов, то описанная выше операция последовательно повторяется.

Если надо заменить один символ на другой, то подведите курсор под заменяемый символ и нажмите на новую клавишу клавиатуры. Аналогично, если необходимо заменить одно слово на другое, то подведите курсор под первый символ заменяемого слова и наберите на клавиатуре новое. Если при этом останутся лишние символы (старое слово было длиннее нового), то их можно удалить, либо последовательно нажимая клавишу «Пробел», либо используя клавишу «ЗБ» вышеописанным способом.

Удаление и вставка слов. Если надо удалить слово без его

замены другим, то подведите под правый символ слова курсор и нажмите на клавишу «2.1». Вы увидите, как часть строки, расположенная вправо от курсора, сдвинется на одну позицию влево, в конце строки появится пробел, а первый символ слова заменится на следующий за ним. Повторите эту операцию нужное число раз и слово исчезнет.

Для того чтобы вставить слово в строку, сначала надо раздвинуть текст в строке, а потом в освободившееся место вписать нужное слово. Как раздвинуть текст? Установите курсор в позицию, которую надо освободить, и нажмите клавишу «2.3». Вся строка, расположенная вправо от курсора, сдвигается на одну позицию; над курсором появится пробел, а последний символ строки пропадает. Повторяя эту операцию требуемое число раз, можно освободить место в строке для вставки пропущенного слова. Жаль только, что может пропасть текст при передвижении строки вправо. Чтобы избежать этого, можно воспользоваться операцией размножения строки.

Размножение и удаление строк. Размножение связано с операцией вставки новых строк. Ведь новые строки надо куда-то вставлять, следовательно, освобождать для них в тексте место, т. е. раздвигать строки. Для раздвижения строк курсор необходимо поместить в строку, перед которой требуется освободить место. Другими словами, курсор должен быть в строке, которую надо опустить. После этого нажмите клавишу «1.1». Тогда весь текст, начиная с помеченной строки, сдвинется вниз, а в тексте появится новая чистая строчка. В случае необходимости операции можно повторить, еще раз, нажав клавишу «1.1».

Если надо размножить строку и уже подготовлено место, то поместите курсор в строку, следующую за размножаемой, и нажмите последовательно клавиши «1.2» и «Р» (команда дублирования).

Для удаления строки установите курсор в эту строку и нажмите клавишу «1.3». Весь лежащий ниже текст поднимется вверх на одну строку, а помеченная строка пропадает.

Последовательное неоднократное повторение вышеописанных операций позволяет вставлять, удалять (сдвигать) и размножать сразу несколько строк.

Поиск и перемещение частей текста (работа с контекстами). У нас появился новый термин — контекст. Так как это слово прочно вошло в язык информатики, раскроем его содержание, следуя энциклопедическому словарю, изданному в 1980 г.: «Контекст — относительно законченный отрывок текста, общий смысл которого позволяет уточнить значения отдельных входящих в него слов и выражений».

Для чего используются контексты при обработке информации? Если требуется выделить для каких-то целей некоторую часть текста, записанного на бумагу, то можно взять цветной карандаш

и либо обвести весь кусок текста, либо подчеркнуть начальную и последнюю строки, либо отметить первое и последнее слова выделяемого текста. А как поступить, если текст высвечивается на экране дисплея и нам надо выделить какую-то его часть? Для этой цели и используются контексты.

Наш редактор под контекстом понимает номера первой и последней строк текста (например, контекст начала — 5-я строка, контекст конца — 59-я строка) либо часть предложений из первой и соответственно последней строк (например, в предыдущем предложении начало можно пометить словом «для», а конец — «контексты»). Разрешается контекст указывать несколькими словами, но главное надо помнить, что, во-первых, редактор воспринимает только выражения длиной не более 80 символов и, во-вторых, контекст начала и контекст конца должны обозначаться разными символами.

Контекстный поиск. Предположим, что необходимо найти ту часть текста, которая начинается или содержит некоторое слово, например «компьютер». Для этого необходимо нажать последовательно клавиши «1.2» и «И» («Искать»). Редактор очистит экран и на экране дисплея в первой строке появится следующее:

УКАЖИТЕ КОНТЕКСТ ДЛЯ ПОИСКА

Как отвечать? На клавиатуре надо набрать контекст (номер строки или слово, в нашем примере слово «компьютер») и нажать клавишу «ВК». Через несколько секунд на экране появится часть исходного текста, содержащая указанный контекст.

Если надо продолжить поиск дальше, то нажмите последовательно клавиши «1.2» и «Е» («Еще»), и поиск продолжится до следующей части текста, содержащей указанный контекст.

Перестановка текста. Для перестановки части текста в другое место существует команда «Переместить». Сначала поместите курсор в ту строку, куда должна попасть первая строка перемещаемого текста. После этого наберите на клавиатуре последовательно «1.2», «П», и редактор обратится к вам со словами, которые появятся на экране дисплея:

УКАЖИТЕ КОНТЕКСТ НАЧАЛА

Как отвечать? Наберите на клавиатуре контекст начала и нажмите клавишу «ВК». На экране появится следующее:

УКАЖИТЕ КОНТЕКСТ КОНЦА

Как отвечать? Наберите на клавиатуре контекст конца перемещаемого текста и нажмите клавишу «ВК».

Редактор найдет указанную часть текста и переставит ее на указанное курсором место, а на экране высветится текст, начинающийся с

нающийся с первой строки перестановки.

Контекстная замена. Вспомните, сколько сил уходит на написание домашних сочинений. Например, вот текст сочинения, завтра его надо сдать, а сейчас можно перечитать написанное. И, как правило, перечитывая, всегда находим такую часть текста, которую хочется изменить. Как мы это делаем? Пишем новый вариант неудавшейся части, затем, используя ножницы и клей, заменяем старый текст на новый и переписываем весь текст набело.

А как реализовать такую операцию при редактировании текста на экране дисплея? Для этого редактору надо дать команду подготовиться к контекстной замене. Нажмите последовательно клавиши «1.2» и «3» («Заменить»). После очищения экрана редактор обратится к вам с вопросами, которые будут появляться на экране дисплея. Первая группа вопросов относится к определению той части текста, в которой надо произвести замену:

УКАЖИТЕ КОНТЕКСТ НАЧАЛА

Как отвечать? Наберите на клавиатуре контекст начала заменяемой части текста и нажмите клавишу «ВК».

На экране появится второй вопрос:

УКАЖИТЕ КОНТЕКСТ КОНЦА

Как отвечать? Наберите на клавиатуре контекст конца заменяемой части текста и нажмите клавишу «ВК».

Следующая группа вопросов должна определить, что и чем будет заменяться. После указания контекста конца на экране дисплея появится вопрос:

УКАЖИТЕ КОНТЕКСТ ЗАМЕНЯЕМЫЙ

Как отвечать? Наберите на клавиатуре контекст заменяемой части текста и нажмите клавишу «ВК». Помните, что редактор «Screen» при замене понимает только контекст длиной не более 80 символов.

Получив информацию о том, что надо заменить, необходимо знать, чем заменять. Поэтому на экране появится новый вопрос:

УКАЖИТЕ КОНТЕКСТ ЗАМЕНИТЕЛЬ

Как отвечать на вопрос? Наберите на клавиатуре тот текст, которым надо заменить старый, и нажмите клавишу «ВК». Опять не забывайте, что длина заменителя не должна превышать 80 символов.

Последний вопрос редактора

ВХОЖДЕНИЕ

надо понимать так: что делать, если в указанной вами части текста заменяемый контекст встретится несколько раз — надо ли сделать замену только в первом найденном предложении или во всех остальных тоже?

Как отвечать на вопрос? Наберите на клавиатуре 1, если надо заменить только первое вхождение контекста, или любое другое целое число, если необходима замена всех вхождений, и завершите ответ нажатием клавиши «ВК».

Редактор приступит к просмотру текста, поиску заменяемых контекстов и их замены, а после окончания работы сообщит, сколько замен произведено, а в скольких случаях замену произвести не удалось.

Почему иногда редактор не может произвести замену? Дело в том, что на экране в строке помещается 80 символов, и если длина контекста заменителя вместе с незаменяемым текстом строки превысит 80 символов, то редактор проводить замену не будет, о чем он и сообщит после окончания работы.

Этим мы закончим описание основных услуг, предоставляемых экраным редактором «Screen» для написания и редактирования программ на языках высокого уровня, различной текстовой информации (статьи, сочинения и др.) и т. д. Поработайте с редактором и вы почувствуете, какие удобства дает ЭВМ для обработки текстов, а в процессе работы перед вами раскроются и другие возможности, о которых мы сейчас не говорили.

ЗАКЛЮЧЕНИЕ

Вот и заканчивается наш рассказ о началах современной информатики. Многое осталось в стороне, но нас утешает мудрое высказывание Козьмы Прутков: «Нельзя объять необъятное», что без всякого лукавства можно отнести к информатике, которая представляет действительно необъятную область человеческой деятельности.

Но если нельзя охватить все, то общее представление об этой области науки иметь нужно и возможно. С годами будет меняться лицо вычислительной техники, появятся новые методы использования электроники, но освоение новых методов и машин будет встречать непреодолимые трудности, если не будут освоены основные положения компьютерной науки — информатики.

Мы не смогли описать достаточно подробно многие темы, например, как устроены и используются суперЭВМ, персональные компьютеры, автоматизированные рабочие места (АРМ), автоматизированные информационно-поисковые системы и т. д. Для желающих подробно ознакомиться с этими темами мы рекомендуем приведенный в конце книги список литературы.

Много интересных проблем обсуждается на страницах журнала «Информатика и образование», который издается с 1986 г.

В 1988 г. стал выпускаться периодический сборник «В мире персональных компьютеров». Безусловно полезна будет читателю серия брошюр, выходящая в издательстве «Знание» — «Вычислительная техника и ее применение».

Во всех крупных городах нашей страны создаются центры информатики, оснащенные современными персональными ЭВМ. Так, например, в Москве открыт Московский городской центр информатики (Смоленский бул., 4). В этом центре работает несколько залов: лекционно-демонстрационный, профессиональных пользователей, индивидуального обучения, игровой. Они оснащены бытовыми компьютерами, комплексом учебной вычислительной техники «Корвет», персональными ЭВМ: ЕС-1840, «Электроника-85», «Роботрон 1715». При необходимости на прокат можно получить различные программные средства.

Вступайте в мир информатики, дорогой читатель. Здесь ждет вас много интересного, незнакомого и удивительного!

Мы старались помочь понять этот мир, немного приоткрыть в него дверь, а как это удалось, судить вам, наш читатель.

ПРИЛОЖЕНИЕ

Единицы объема запоминающих устройств

Бит	Наименьшая единица информации, разряд в двоичной системе исчисления
Байт	Группа данных из 8 бит
Слова	Группа данных, кратная 8 бит
Двойное слово	Группа данных, кратная 16 бит
Килобайт (К байт)	1024 байт
Мегабайт (М байт)	1024 К байт
Гигабайт (Г байт)	1024 М байт

Единицы времени, используемые в информатике

Миллисекунда (1 мс)	10^{-3} с
Микросекунда (1 мкс)	10^{-6} с
Наносекунда (1 нс)	10^{-9} с
Пикосекунда (1 пс)	10^{-12} с

Т а б л и ц а П1. Основные характеристики накопителей информации

Тип накопителя	Размер	Объем хранимой информации, М байт	Средняя скорость считывания-записи, К бит/с
МЛ	Длина 720 мм, ширина 13 мм	5	20
ГМД	Диаметр, мм: 203	0,5...1,6	800
	133	0,2...1,2	250
	89	1,2...3,0	250
МД (типа «Винчестер»)	Диаметр 133 мм	5...300	5000
ОД	Диаметр 133 мм	200	5000

Типичные характеристики дисплеев на основе электронно-лучевой трубки

Размер по диагонали, см	30,5; 33; 35,6
Объем текстовой информации на экране	25 строк по 80 символов, 25 строк по 40 символов
Разрешающая способность в графическом режиме, точек	1024 × 2048
Количество цветов, выводимых на экран одновременно (для цветных дисплеев)	4—16
Количество цветовых гамм (палетт)	2—256

Т а б л и ц а П2. Основные типы принтеров для персональных ЭВМ

Тип принтера	Разрешающая способность, точка/мм	Скорость печати, знак/с
Точечно-матричный	3...8	30...90
Термографический	3...7	60...180
Струйный	3...5	20...150
Лазерный	10...12	200...300

Таблица ПЗ. Характеристики персональных ЭВМ

Характеристика	Сфера применения			
	научная	деловая	обучение	бытовая
Средний объем ОЗУ, К байт	640...1200	256	256	64
Средний объем ВЗУ, М байт	10...40	0,5...20	0,5...10	0,1...1
Основные применения	Табличные вычисления, пакеты прикладных программ; обработка графической и текстовой информации; базы данных; АРМ; подключение к контрольно-измерительной аппаратуре	Обработка таблиц, графической и текстовой информации; базы данных; подключение к телефонной сети	Автоматизированные обучающие системы, профессиональная подготовка	Игры, обработка текстов, ведение домашнего хозяйства, подключение к телефонной и телевизионной сети, контрольно-обучающие системы

Таблица П4. Пять поколений вычислительных систем

Поколение	Аппаратные средства	Программные средства	Основной вид обрабатываемых данных
Первое (50-е годы)	Электронные лампы	Двойное кодирование	Двоичные числа
Второе (начало 60-х годов)	Транзисторы	Ассемблер	Числа
Третье (конец 60-х годов)	Интегральные схемы	Языки высокого уровня: Паскаль, Фортран, ПЛ-1, Алгол	Числа и некоторые виды текстовой информации
Четвертое (70-е годы)	БИС, СБИС	Простые информационные системы	Числа, тексты, таблицы
Пятое (середина 80-х годов)	Аппаратная реализация традиционных функций программных средств	Информационные системы высокого уровня для принятия решений	Числа, текст, изображения

Примеры советских ЭВМ четырех поколений

Поколение

Первое	МЭСМ, М-20, БЭСМ-1, «Стрела-1»
Второе	БЭСМ-2, -3, -3М, «Минск-22, -32», серия «Урал», М-220, серия «Мир»
Третье	БЭСМ-6, «Урал-11, -12, -14», серия ЕС, ряд 1, 2
Четвертое	М-10, «Эльбрус-2», ЕС-1045, -1065, СМ-3, -4, -1800, ПС-2000 (матричная ЭВМ); персональ- ные компьютеры: ДВК-2М, БК-0010, «Электроника-85», «Искра- 226, -1030», ЕС-1841, «Корвет»

Т а б л и ц а П5. Представление чисел в различных системах счисления

Десятичная	Двоичная	Восьмеричная	Шестнадцатеричная
0	0	0	0
1	1	1	1
2	10	2	2
3	11	3	3
4	100	4	4
5	101	5	5
6	110	6	6
7	111	7	7
8	1000	10	8
9	1001	11	9
10	1010	12	A
11	1011	13	B
12	1100	14	C
13	1101	15	D
14	1110	16	E
15	1111	17	F
16	10000	20	10

- Кибернетика.** Становление информатики: Сб. статей.—М.: Наука, 1986.—192 с.
- Персональные компьютеры.** Информатика для всех: Сб. статей.—М.: Наука, 1987.—149 с.
- Перегудов М. А., Халамайзер А. Я.** Бок о бок с компьютером.—М.: Высш. шк., 1987.—192 с.
- Мичи Д., Джонстон Р.** Компьютер — творец: Пер. с англ.—М.: Мир. 1987.—255 с.
- Лоберг Р., Лутц Т.** Домашний компьютер: Пер. с нем.—М.: Дет. лит., 1987.—95 с.
- Поляков В. Т.** Посвящение в радиоэлектронику.—М.: Радио и связь, 1988.—352 с.
- Финк Л. М.** Папа, мама, я и микрокалькулятор.—М.: Радио и связь, 1988.—272 с.
- Зеленко Г. В., Попов В. В., Попов С. Н.** Домашний компьютер.—М.: Радио и связь, 1989.
- Свириденко С. С.** Современные информационные технологии.—М.: Радио и связь, 1989.
- Геворкян Г. Х., Семенов В. Н.** Бейсик — это просто.—М.: Радио и связь, 1989.

Адрес	Машинное слово, используемое для указания определенной области памяти ЭВМ, в которой находится элемент данных
Алгоритм	Набор предписаний, однозначно определяющих содержание и последовательность выполнения операций для решения задач
База данных	Информация, упорядоченная в виде набора элементов (записей) одинаковой структуры. Для обработки записей используются специальные программы, позволяющие их упорядочить, делать выборки по указанному правилу. Обычно базой данных называют информацию и программы ее обработки
Байт	Единица измерения информации, которая используется для указания объема памяти ЭВМ и равна восьми битам
Бит	Двоичный разряд, элементарная единица информации, принимающая значения 0 или 1
Внешние накопители	Устройства для постоянного хранения информации (программ и данных): магнитные ленты, гибкие и жесткие магнитные диски
Данные	Информация, представленная в форме, воспринимаемой для обработки ЭВМ или человеком
Дисплей	Устройство визуального отображения информации. На экран дисплея выводятся тексты и графические изображения
Интерактивный (диалоговый) режим	Режим взаимодействия пользователя с ЭВМ, при котором каждый запрос с оконечного устройства (дисплея) вызывает ответное действие вычислительной машины
Интерфейс	Средство сопряжения устройств вычислительной техники
Каталог файлов	Логический раздел на внешнем накопителе, объединяющий группу файлов и хранящий информацию об имени, объеме, времени создания файла
Клавиатура	Устройство ввода текстов, чисел и управляющей информации в память ЭВМ
Курсор	Мигающий или выделенный другим способом значок на экране дисплея, обычно указывающий позицию, в которой отображается очередной вводимый с клавиатуры символ
Массив	Многомерная структура данных, в которой элементы упорядочены таким образом, что их описание однозначно определяет положение каждого элемента
Математическое обеспечение	Совокупность программных средств, обеспечивающих эффективное функционирование ЭВМ
Машинное слово	Последовательность битов или знаков, трактуемая в процессе обмена или обработки как единый элемент данных
Оперативная память, или оперативное запоминающее устройство (ОЗУ)	Устройство, в котором размещаются во время выполнения программы. При выключении ЭВМ содержимое ОЗУ не сохраняется
Операционная система (ОС)	Комплекс управляющих программ, обеспечивающих автоматическое управление вычислительными про-

Периферийные устройства	цессами и ресурсами вычислительной системы. Устройства, выполняющие внешние функции машинной обработки информации — подготовку ее к виду, удобному для ввода в ЭВМ или вывода из машины, хранения, передачи и т. п.
Постоянное запоминающее устройство (ПЗУ)	Устройство, в котором постоянно сохраняется информация. Чтение из ПЗУ осуществляется с более низкой скоростью, чем из ОЗУ; запись новой информации в ПЗУ невозможна; при выключении ЭВМ содержимое ПЗУ сохраняется
Пакет прикладных программ	Комплекс взаимосвязанных программ для решения задач определенного плана
Принтер	Устройство для печатания на бумаге информации, передаваемой из памяти компьютера
Программа	Последовательность действий (команд, операторов), описанная на специальном языке и предназначенная для выполнения ЭВМ
Процессор	Электронное устройство, обеспечивающее выполнение арифметических, логических и управляющих операций, заданных программой в машинном коде
Регистр	Накопитель на электронных переключающих элементах, емкость которого обычно равна одному машинному слову и предназначена для оперативного хранения информации
Стек	Структура данных, при которой очередное обращение в целях записи или выборки всегда осуществляется в начале (вершину)
Таймер	Счетчик, содержимое которого изменяется с поступлением сигнала времени
Текстовый редактор	Программа для подготовки и обработки текстовой информации, позволяющая вводить символы (буквы, цифры и другие знаки) с клавиатуры и осуществлять различные действия по изменению (редактированию) текстов
Транслятор	Обслуживающая программа, преобразующая текст, написанный программистом на языке программирования, в машинный код, который может быть исполнен компьютером
Файл	Последовательность записей, размещаемая на внешних запоминающих устройствах и рассматриваемая в процессе пересылки как единое целое
Язык программирования высокого уровня	Язык, средства которого допускают описание решаемой задачи в наглядном виде

ОГЛАВЛЕНИЕ

Предисловие	3
Глава 1. Становление информатики	4
1.1. Как родилась информатика	5
1.2. Как появились ЭВМ	10
Глава 2. Портрет ЭВМ	16
2.1. Аппаратные средства	17
2.2. Программные средства (программное обеспечение)	28
Глава 3. Поколения ЭВМ	38
3.1. Первое поколение	39
3.2. Второе поколение	39
3.3. Третье поколение	40
3.4. Четвертое поколение	40
3.5. Пятое поколение	45
Глава 4. Как используются ЭВМ?	52
4.1. Математические модели и ЭВМ	53
4.2. Представление чисел в ЭВМ и системы счисления	67
4.3. Как видит окружающий мир компьютер (представление данных в ЭВМ)	78
4.4. Алгоритм и алгоритмические языки	85
4.5. Экранный редактор «Screeп»	95
Заключение	104
Приложение	106
Список рекомендуемой литературы	109
Краткий словарь терминов	110

Научно-популярное издание

Межиздательская серия «Научно-популярная библиотека школьника»

Решетников Валерий Николаевич
Сотников Александр Николаевич

ИНФОРМАТИКА — ЧТО ЭТО?

Заведующий редакцией Ю. Г. Ивашов
Редактор Л. Е. Кочарьянц
Обложка художника Т. А. Доброхотовой-Майковой
Художественный редактор А. В. Проценко
Технический редактор Т. Г. Родина
Корректор Л. А. Буданцева

ИБ № 1770

Сдано в набор 04.01.89. Подписано в печать 11.05.89. Т-09965. Формат 60×88¹/₁₆. Бумага оберточная.
Гарнитура литературная. Печать офсетная. Усл. печ. л. 6,86. Усл. кр.-отт. 7,35. Уч.-изд. л. 6,97.
Тираж 236 000 экз. (4-й завод 150 001—200 000 экз.). Изд. № 22310. Зак. № 1816. Цена 30 к.

Издательство «Радио и связь». 101000 Москва, Почтамт, а/я 693

Московская типография № 4 «Союзполиграфпрома» при Государственном комитете СССР по делам издательств, полиграфии и книжной торговли. Москва, И-41, Б. Переяславская, 46

Как возникли ЭВМ? Из чего состоит компьютер? Что такое программное обеспечение? Как используются ЭВМ? Как в электронной машине представляются числа?

На эти вопросы читатель найдет ответы в этой книге. Он узнает, что такое математическая модель и как "видит" окружающий мир ЭВМ. Познакомится с компьютерами разных поколений и заглянет в завтрашний день информатики. И даже, может быть, почувствует себя на мгновение сидящим у экрана дисплея. Авторы помогут читателю совершить увлекательную экскурсию в мир информатики — новой области знания, переживающей в наши дни бурное развитие.



«Радио и связь»