Учебное пособие
для педагогических
институтов

В.М.Заварыкин
В.Г.Житомирский
М.П.Лапчик

ТЕХНИКА ВЫЧИСЛЕНИЙ И АЛГОРИТМИЗАЦИЯ

В.М.Заварыкин
В.Г.Житомирский
М.П.Лапчик

ТЕХНИКА ВЫЧИСЛЕНИЙ И АЛГОРИТМИЗАЦИЯ

Допущено Министерством просвещения СССР
в качестве учебного пособия
для студентов педагогических институтов
по физико-математическим специальностям

МОСКВА «ПРОСВЕЩЕНИЕ» 1987

ББК 22.12
З-12

Рецензенты:

кафедра вычислительной математики и программирования
Минского педагогического института им. А. М. Горького
(зав. кафедрой доцент, канд. физ.-мат. наук *А. И. Павловский)*;
доцент, канд. физ.-мат. наук *А. Р. Есаян*
(Тульский государственный педагогический институт им. Л. Н. Толстого).

Заварыкин В. М. и др.

З-12 Техника вычислений и алгоритмизация: Вводный курс: Учеб. пособие для студентов пед. ин-тов по физ.-мат. спец./ В. М. Заварыкин, В. Г. Житомирский, М. П. Лапчик.— М.: Просвещение, 1987.— 160 с.: ил.

Пособие является первой (вводной) частью учебно-методического комплекса дисциплин, обеспечивающих необходимый уровень подготовки будущих учителей к использованию микропроцессорной техники в школе. Рассмотрены приемы вычислений на калькуляторах и методы оценки точности результатов, приведены начальные сведения об алгоритмических структурах и программировании на языке Бейсик.

З $\frac{4309000000-557}{103(03)-87}$ 22—87

ПРЕДИСЛОВИЕ

Использование в средней школе микропроцессорной техники предъявляет новые требования к учителю, который должен хорошо знать возможности новых вычислительных средств, а также правила их использования при решении учебных задач. Учебные планы физико-математических факультетов, ведущих подготовку учителей для преподавания школьного курса «Основы информатики и вычислительной техники», предусматривают цикл учебных дисциплин по изучению и использованию современной вычислительной техники на протяжении всех лет обучения в вузе. Первым в этом цикле дисциплин является курс «Техника вычислений и алгоритмизация».

Цель этого курса — закрепить знания по основам информатики и вычислительной техники, полученные ранее в школе, и развить навыки ее практического использования при решении учебных задач; дать представление о методах оценки точности вычислений с помощью калькуляторов и микроЭВМ; показать различные типы образцов микропроцессорной техники. Одной из важных задач вводного курса является объяснение возможности эффективного использования современных вычислительных средств (калькуляторов, микроЭВМ) при изучении дисциплин учебного плана: математического анализа, алгебры, геометрии, физики, практикумов по решению математических и физических задач. Курс техники вычислений и алгоритмизации помимо этого должен служить основой для последующего использования микропроцессорной техники во внеучебной, кружковой, научно-исследовательской работе со студентами.

Настоящее пособие адресовано в первую очередь студентам первых курсов физико-математических фа-

культетов, получающим квалификацию учителя математики (физики), информатики и вычислительной техники. При соответствующей ориентации практических заданий это пособие может быть использовано также и на химико-биологических, естественно-географических, индустриально-педагогических факультетах, факультетах учителей начальных классов, т. е. там, где подготовка учителей в той или иной мере предполагает использование микропроцессорной техники для вычислений.

Изучение курса сопровождается коротким лабораторным практикумом. Для выполнения практических работ требуется, как минимум, комплект калькуляторов различных типов: арифметических, инженерных, программируемых (из расчета по меньшей мере 15—20 шт. каждого типа на поток студентов в 100 человек). С наилучшим эффектом курс может быть поставлен на базе кабинетов вычислительной техники, оснащенных достаточным количеством персональных микроЭВМ типа «Электроника Д3-28», «Искра-226», ДВК-1, ДВК-2М, «Агат» и т. п.

Глава 1
ТЕХНИКА ВЫЧИСЛЕНИЙ НА МИКРОКАЛЬКУЛЯТОРАХ

1.1. Общие сведения о микрокалькуляторах

Человечество всегда нуждалось в доступных массовых и эффективных средствах ручного счета. История показывает, что путь создания таких приспособлений был долгим и многотрудным — абак, русские счеты, математические таблицы, логарифмическая линейка, механические арифмометры — вот далеко не полный перечень вычислительных средств и инструментов, в разное время создаваемых и используемых человеком. Уровень каждого из этих вычислительных приспособлений является прямым отражением научно-технического потенциала своего времени.

В современную эпоху в деле создания средств для ручных вычислений произошел резкий качественный скачок: разработан и в настоящее время получил широкое распространение новый тип вычислительных устройств — электронные малогабаритные клавишные вычислительные машины, называемые *микрокалькуляторами* (МК). Массовое производство МК в нашей стране относится к концу 60-х — началу 70-х годов настоящего столетия. В короткий срок электронные калькуляторы благодаря своим потребительским свойствам и богатым функциональным возможностям (малые габариты, высокое быстродействие и точность вычислений, широкий диапазон выполняемых операций, простота эксплуатации) проникли во все сферы деятельности людей, а их производство стало крупной отраслью промышленности. В настоящее время микрокалькуляторы широко используются в учебном процессе средней и высшей школы.

По функциональным возможностям выпускаемые в нашей стране микрокалькуляторы разделяют на три типа: арифметические, инженерные (непрограммируемые) и программируемые.

Арифметические МК — это наиболее простые в эксплуатации микрокалькуляторы, предназначенные для выполнения обычных арифметических расчетов. Основное назначение этих приборов — выполнение четырех арифметических действий. К этой группе МК относятся отечественные микрокалькуляторы «Электроника» моделей Б3-23, Б3-26, Б3-30, С3-22, МК-44, МК-55, МК-57 и др.

Инженерными МК (или МК для научно-технических расчетов) называют микрокалькуляторы, которые обладают возможностью не только выполнять арифметические операции, но и вычислять значения основных элементарных функций. К ним относятся следующие модели «Электроники»: Б3-18, Б3-19, Б3-22, Б3-15, Б3-36, МК-41, МК-45, МК-51 и др.

Программируемые МК обладают всеми основными возможностями микрокалькуляторов указанных выше групп, но, кроме этого,

Рис. 1

они могут автоматически выполнять целую серию заранее составленных команд (программу), введенную в память МК с помощью клавиатуры. По своему характеру программируемые МК, несмотря на свои миниатюрные размеры, принципиально не отличаются от больших программно-управляемых электронных вычислительных машин (ЭВМ). К ним относятся отечественные модели «Электроники»: Б3-21, Б3-34, МК-46, МК-54, МК-56, МК-64.

Микрокалькулятор представляет собой портативный вычислительный прибор, имеющий два-три десятка клавиш (кнопок) и световой индикатор (дисплей) для чтения результатов. В состав МК входит сложное счетно-решающее устройство, содержащее десятки тысяч функциональных элементов и соединений. Числа в процессе вычислений размещаются в особых устройствах памяти — *регистрах*. Схема основных взаимосвязей между клавиатурой, индикатором и дополнительными регистрами памяти изображена на рисунке 1.

На индикаторе всегда изображено то число, которое в данный момент хранится в регистре индикации (X). Число, набираемое на клавиатуре, попадает в регистр X и тут же высвечивается на индикаторе. Регистр X выполняет роль одного из операционных регистров, роль другого операционного регистра у многих моделей МК выполняет особый рабочий регистр Y. Числа в регистр Y попадают из регистра X (например, после нажатия операционных клавишей сложения, умножения и др.). Память МК включает также и дополнительные регистры памяти (П), которых у программируемых МК может быть более десятка, а у простейших и инженерных калькуляторов, как правило, от одного до трех. Одно из назначений дополнительных регистров — хранить промежуточные результаты вычислений, что позволяет производить довольно сложные вычисления без выписывания результатов промежуточных действий на бумаге.

Контрольные вопросы

1. Чем различаются основные типы микрокалькуляторов: арифметические, инженерные и программируемые?
2. Где хранятся числа в МК в процессе вычислений?
3. Как попадают числа в регистр X? в регистр Y?
4. Содержимое какого регистра всегда отражается на индикаторе?

1.2. Способы представления чисел

Микрокалькуляторы оперируют числами, вводимыми в память и изображаемыми на индикаторе в десятичной системе счисления. Будем считать для определенности, что МК может работать с 8-разрядными десятичными числами. Если учесть еще один разряд для изображения знака числа, то на бумаге разрядную сетку МК можно изобразить так, как показано на рисунке 2. В знаковом разряде индицируется только знак «—», при изображении положительных чисел этот разряд остается свободным. Для отделения целой части числа от дробной используется десятичная запятая (или точка). В большинстве случаев запятая при изображении числа на индикаторе размещается в том же разряде, что и цифра, за которой запятая следует*. Так, например, число 2,037 на индикаторе МК, разрядная сетка которого изображена на рисунке 2, будет изображаться так, как показано на рисунке 3**.

В микрокалькуляторах используются в основном два способа представления чисел: с *естественным размещением запятой* и в форме с так называемой *плавающей запятой*. Естественная форма представления чисел наиболее распространена в «карманных» МК. В этом случае запятая в изображении числа всегда присутствует явно (см. рис. 3), может располагаться в любом цифровом разряде сетки, а ее место определяется или при вводе числа с клавиатуры, или в результате выполнения операции. Легко видеть, что в МК с 8-разрядной сеткой в форме с естественным размещением запятой могут быть представлены лишь числа от $\pm 10^{-7}$ до $\pm(10^{8}-1)$. Этот диапазон вполне обеспечивает возможность решения многих практических задач. Если в результате выполнения операции получится число, по модулю меньшее, чем 10^{-7}, то на индикаторе МК с естественной формой представления чисел высветится нуль, значащие разряды, не вошедшие в разрядную сетку МК, пропадают. Такие числа называют *машинным нулем*. Если в результате получится число, по модулю

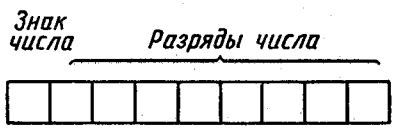


Рис. 2

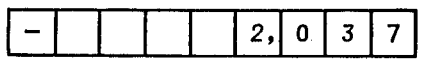


Рис. 3

*В некоторых моделях МК запятая при изображении на индикаторе занимает полный разряд, что уменьшает на единицу количество умещаемых в разрядной сетке знаков дробного числа.

**Иногда число на индикаторе МК занимает не правые, а левые разряды сетки, однако это не имеет в данном случае принципиального значения.

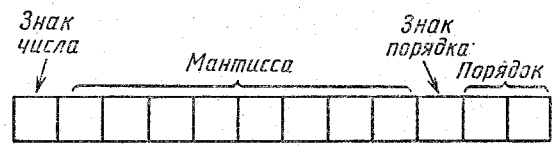


Рис. 4

большее, чем 10^8-1, загорится сигнал переполнения — специальный знак в первом слева (знаковом) разряде индикатора.

Наиболее совершенные модели МК используют представление чисел в форме с плавающей запятой. В этом случае число автоматически представляется в виде $M \cdot 10^p$, где M — мантисса, p — порядок числа, т. е. показатель степени, в которую нужно возвести 10, чтобы, умножив результат на мантиссу, получить данное число. Для изображения знака и величины порядка в разрядной сетке таких МК имеются специальные разряды — один знаковый и два для изображения величины порядка (рис. 4). Значения мантиссы в МК удовлетворяют условию $1 \leqslant M < 10$. Так, например, число 0,000327 при представлении в МК с плавающей запятой будет преобразовано к виду $3{,}27 \cdot 10^{-4}$. Максимальная величина порядка выражается двузначным числом 99. Это дает огромный диапазон представления чисел: от $\pm 10^{99}$ до $\pm 9{,}9999999 \cdot 10^{99}$. К числу микрокалькуляторов с плавающей формой представления чисел относятся ориентированный на среднюю школу калькулятор МКШ-2, МК-41 и др.

Контрольные вопросы

1. Какие два способа представления чисел используются в микрокалькуляторах?
2. Каков диапазон представления чисел с естественным размещением запятой в МК с 8-разрядной сеткой? Может ли такой микрокалькулятор хранить величину заряда электрона ($e = 1{,}6022 \cdot 10^{-19}$)?
3. Какие числа называют машинным нулем?

1.3. Вычисления на арифметических микрокалькуляторах

Технические особенности эксплуатации конкретных микрокалькуляторов (обеспечение электропитанием, подготовка к работе и т. п.) не сложны и описываются в руководствах по эксплуатации. К тому же общие принципы вычислений с помощью МК одинаковы: дело сводится к последовательному нажатию клавиш с цифрами и знаками операций и считыванию результатов, которые практически мгновенно высвечиваются на индикаторе. Различные МК во многом схожи и по внешнему виду. Клавиатура МК арифметического типа модели «Электроника

Б3-26» показана на рисунке 5. Клавиатура МК состоит из цифровых и операционных клавиш, которые обычно окрашиваются в разные цвета.

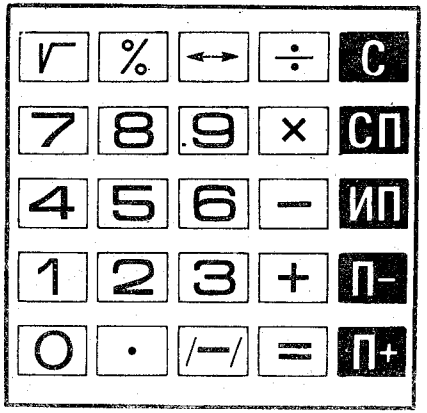


Рис. 5

Различные микрокалькуляторы простейшего (арифметического) типа обладают множеством схожих функциональных возможностей, круг которых все-таки ограничен. К МК этого класса принадлежит и целый ряд портативных калькуляторов, допускающих автономное электропитание, и группа арифметических МК настольного типа с питанием от сети переменного тока (табл. 1.5). Вычислительные возможности простейших МК рассмотрим применительно к общему случаю. При необходимости для иллюстрации отдельных операций будут использоваться конкретные модели калькуляторов.

Ввод чисел. При вводе чисел используются цифровые клавиши, клавиша с десятичной запятой (точкой), а при вводе отрицательных чисел — еще и клавиша [/—/]. Так, при вводе числа 34,085 последовательно нажимаются клавиши:

[3] [4] [,] [0] [8] [5].

Последовательность появления знаков на индикаторе при выполнении этого упражнения приведена в таблице 1.1.

Таблица 1.1

Ввод	Индикатор (X)
3	3
4	34
,	34,
0	34,0
8	34,08
5	34,085

Для присвоения числу на индикаторе знака «минус» или для изменения знака этого числа на противоположный нажимается клавиша [/—/]. Как уже отмечалось выше, отображаемое на индикаторе МК число одновременно размещается в операционном регистре X. Гашение числа на индикаторе (как и в регистре Y) осуществляется однократным нажатием клавиши сброса [C]. Многие МК имеют клавишу [↔], нажатие которой производит обмен содержимым между регистрами X и Y: однократное нажатие этой клавиши меняет содержимое регистров X и Y местами, при повторном ее нажатии содержимое этих регистров восстанавливается в первоначальном виде. Для гашения содержимого регистра Y в некоторых МК (например, в Б3-23) требуется двукратное нажатие клавиши сброса [C]: первым нажатием гасится содержимое индикатора (т. е. X), вторым — нуль индикатора записывается в Y. Следует знать, что содержимое всех регистров гасится при отключении МК от сети. Впредь будем предполагать, что перед началом счета операционные регистры X и Y очищены.

Выполнение арифметических действий. Основное назначение арифметического МК — это выполнение операций сложения, вычитания, умножения и деления. Для выполнения каждого из этих арифметических действий над двумя десятичными числами нужно:

1. Ввести первое число (оно высвечивается на индикаторе).
2. Нажать одну из клавиш [+] [−] [×] [÷].
3. Ввести второе число (при этом первое число с индикатора исчезает, на нем высвечивается второе число).
4. Нажать клавишу [=] (на индикаторе высвечивается результат операции).

Пример 1.3.1. Для вычисления суммы 13,7 + 0,9 нужно выполнить следующую последовательность действий:

[1] [3] [,] [7] [+] [0] [,] [9] [=].

В будущем при выписывании последовательности клавиш для решения на МК вычислительных задач не будем особо выделять клавиши ввода чисел. Это сделает вычислительные программы для МК более компактными и наглядными. Так, программа выполнения приведенного выше задания будет иметь вид:

13,7 [+] 0,9 [=].

После нажатия клавиши [=] на индикаторе высветится ответ: 14,6. В выполнении двухместных операций, к которым относятся арифметические операции, участвуют оба операционных регистра

МК: регистр X и регистр Y. Для лучшего понимания возможностей микрокалькуляторов полезно знать последовательность пересылок чисел-операндов в ходе выполнения операции. При решении задачи на выполнение любой из арифметических операций над двумя числами в МК обычно происходит следующее. После ввода первого числа оно размещается в регистре X (и соответственно высвечивается на индикаторе). При нажатии одной из клавиш [+] [−] [×] [÷] содержимое регистра X пересылается в регистр Y, причем содержимое X остается без изменения. При введении второго числа оно размещается в регистре X, а его прежнее содержимое стирается. При нажатии на клавишу [=] микрокалькулятор выполняет подготовленную операцию и записывает ответ в регистр X (индикатор), а в регистр Y засылается второе число*. Зная порядок перемещения чисел в памяти МК в ходе выполнения операции, опытный вычислитель может использовать эти числа в последующих вычислениях без дополнительного ввода их с клавиатуры, чем достигается экономия времени и повышается эффективность вычислений. Содержимое операционных регистров X и Y при выполнении программы 13,7 [+] 0,9 [=] на МК Б3-26 приведено в таблице 1.2. Пользуясь клавишей [↔], можно пронаблюдать за содержимым регистров X и Y в процессе счета.

Таблица 1.2

Ввод	X	Y
13,7	13,7	0
[+]	13,7	13,7
0,9	0,9	13,7
[=]	14,6	0,9

Выполнение цепочки операций. Как показывает рассмотренный пример, выполнение ранее установленной операции обеспечивается нажатием клавиши [=], — в этом собственно и состоит ее назначение. Однако в МК этим свойством обычно обладают и операционные клавиши [+], [−], [×], [÷]; если вместо кла-

*У некоторых моделей МК в регистр Y при этом засылается, как и в X, результат выполненной операции.

виши [=] в рассмотренном примере нажать одну из операционных клавишей, то на индикаторе высветится тот же результат, что и в предыдущем случае. Более того, в результате МК будет подготовлен к выполнению над полученным результатом арифметической операции, клавиша которой была нажата последней. Это свойство клавишей [+], [−], [×], [÷] (его называют их *вторым назначением*) позволяет производить цепочку арифметических действий, не выписывая промежуточных результатов и экономя число нажатий клавиш.

Пример 1.3.2. Пусть результат рассмотренного выше действия сложения нужно умножить на 8,31, т. е. выполнить вычисление: $(13{,}7+0{,}9)\times 8{,}31$. Задача решается по программе:

13,7 [+] 0,9 [×] 8,31 [=].

В этом случае нажатие клавиши [×] производит установленную ранее операцию сложения и подготавливает МК к умножению. Это умножение и осуществляется затем нажатием клавиши [=]. Последовательность изменения содержимого регистров X и Y в ходе выполнения этого упражнения воспроизведена в таблице 1.3.

Таблица 1.3

Ввод	X	Y
13,7	13,7	0
[+]	13,7	13,7
0,9	0,9	13,7
[×]	14,6	0,9
8,31	8,31	14,6
[=]	121,326	8,31

Аналогичным образом могут вычисляться значения многих арифметических выражений, в которых порядок выполнения действий соответствует порядку их написания. К таким относятся, например, выражения типа: $(a\pm b)\times c$, $(a\pm b)/c\pm d$ и др.

А что делать, если нужно вычислить выражение, порядок действий в котором не совпадает с порядком их написания, а некоммутативность последней предусмотренной операции не позволяет преобразовать выражение к одному из указанных выше вариантов? К ним относятся, например, следующие три простейших выражения, включающие некоммутативные операции вычитания и деления: $a/(b\pm c)$; $a-b\times c$; $a-b/c$. Что касается двух последних выражений, то они допускают желаемое преобразование: $-(b\times c)+a$, $-(b/c)+a$. Теперь задача может быть решена с использованием клавиши [/−/], нажатие которой изменяет знак числа в X на противоположный. Возьмем для примера выражение: $17-2\times 5=-(2\times 5)+17$. Вычисление обеспечивается программой:

2 [×] 5 [+] [/−/] 17 [=].

Прежде чем выполнить сложение с числом 17, здесь устанавливается знак «−» перед числом на индикаторе.

Однако в каждом из трех указанных случаев легче избежать выписывания промежуточных результатов на бумаге, если воспользоваться клавишей [↔].

Пример 1.3.3. Пусть надо вычислить значение выражения:

$$51{,}8/(9{,}6-0{,}35).$$

Сначала нужно выполнить вычитание, а потом деление. Записи промежуточного результата можно избежать, если перед выполнением команды деления поменять местами содержимое регистров X и Y. Вычисление обеспечивается программой:

9,6 [−] 0,35 [÷] 51,8 [↔] [=].

(Ответ: 5,6.)

Порядок вычислений виден из таблицы 1.4.

Таблица 1.4

Ввод	X	Y
9,6	9,6	0
[−]	9,6	9,6
0,35	0,35	9,6
[÷]	9,25	0,35
51,8	51,8	9,25
[↔]	9,25	51,8
[=]	5,6	9,25

Значительно упрощается выполнение цепочных вычислений в МК, имеющих дополнительные регистры памяти (П). При на-

личии одного дополнительного регистра, как, например, в Б3-26, на клавиатуре имеются клавиши: [СП] — гашение содержимого регистра П; [П+] — прибавление числа, хранящегося в регистре X (индикаторе), к содержимому регистра П; [П−] — вычитание из содержимого регистра П числа, записанного в регистре X; [ИП] — вызов числа из регистра П в регистр X. Этот несложный аппарат позволяет избегать выписывания промежуточных результатов при выполнении сложных цепочных вычислений.

Пример 1.3.4. Вычислим значение выражения $(a+b)\times(c+d)$ с использованием дополнительного регистра памяти. Программа имеет вид:

[СП] a [+] b [=] [П+] c [+] d [×] [ИП] [=].

Некоторые из более поздних моделей МК арифметического типа имеют несколько дополнительных регистров памяти, использование которых предоставляет вычислителю большие возможности для хранения результатов промежуточных действий или даже позволяет автоматически накапливать их. Рассмотрим для примера некоторые особенности настольного микрокалькулятора арифметического типа «Электроника МК-44», располагающего тремя дополнительными регистрами памяти. Клавиатура этого МК изображена на рисунке 6.

Работа с дополнительными регистрами памяти П1, П2 и П3 обеспечивается двумя левыми вертикальными рядами клавиш. Запись содержимого регистра X в один из трех дополнительных регистров памяти производится клавишей [x→П] с последующим нажатием одной из цифровых клавиш [1], [2] или [3], смотря по тому, в какой из регистров — П1, П2 или П3 — надлежит записать содержимое регистра X. Аналогично при вызове содержимого одного из трех регистров П в регистр X после

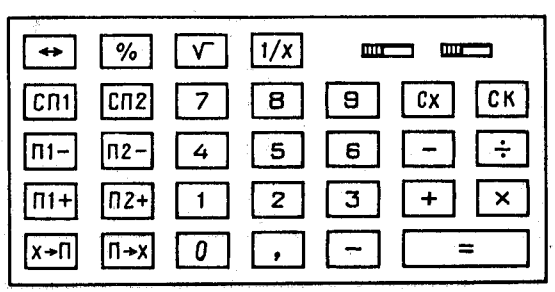


Рис. 6

нажатия клавиши [П→x] нажимается одна из клавиш [1], [2], [3], определяющая номер регистра П, из которого вызывается информация. Гашение содержимого любого из трех дополнительных регистров П можно произвести пересылкой нуля из регистра X с помощью клавиш [С], [x→П] и одной из клавиш [1], [2], [3]. Для гашения содержимого регистров П1 и П2 имеются специальные клавиши [СП1] и [СП2]. Нажатием клавиш [П1+] и [П2+] содержимое регистра X добавляется к содержимым регистров П1 и П2, а с помощью клавиш [П1−] и [П2−] содержимое регистра X вычитается соответственно из содержимого дополнительных регистров П1 и П2.

Пример 1.3.5. Вычислить значение выражения:

$$a+bc-d/e.$$

Используем регистр П1. Программа может иметь вид:

a [x→П] [1] b [×] c [=] [П1+] d [÷] e [=] [П1−] [П→x] [1].

В МК-44 предусмотрен также режим *автоматического накопления,* который устанавливается специальным переключателем. В этом режиме при каждом нажатии клавиши [=] получаемый результат (содержимое регистра X) автоматически прибавляется к содержимому третьего дополнительного регистра памяти.

Пример 1.3.6. В режиме автоматического накопления значение выражения вида $ax+by+cz$ может быть получено по программе (сначала производится очистка регистра П3):

[С] [x→П] [3] a [×] x [=] b [×] y [=] c [×] z [=] [П→x] [3].

Работа с константами. Представление о вычислительных возможностях простейших МК арифметического типа будет неполным, если вычислитель не знаком с особым режимом работы МК, называемым *режимом констант.* Режим констант вводится автоматически в большинстве моделей МК, а сущность его состоит в том, что при выполнении двухместной арифметической операции после нажатия клавиши [=] МК запоминает выполненную при этом операцию и число, попавшее в регистр Y (оно становится константой). В обычных условиях это не означает ничего особенного, однако если вслед за этим по-

Таблица 1.5

Арифметические микрокалькуляторы	С автономным источником питания						Настольного типа								
Функциональные возможности	Б3-14	Б3-23	Б3-24Г	МК-57	Б3-26	Б3-30	Б3-39	МК-33	МК-40	МК-53	Б3-05М	МК-59	С3-22	МК-42	МК-44
4 арифметических действия	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+
Действия с константой	+	+	+	+	+	+	+	+	+	+	+		+	+	+
Обратная величина числа	+			+	+			+			+		+	+	+
Извлечение квадратного корня	+			+	+	+	+			+					+
Процент от числа	+	+		+	+	+	+	+	+	+	+	+	+	+	+
Накопление в отдельном регистре памяти			+	+	+			+	+	+	+	+	+	+	+
Вычитание из содержимого регистра памяти				+	+			+	+	+	+			+	+
Накопление итоговых сумм процентных результатов									+						+
Обмен содержимым регистров X и Y				+	+			+				+			+
Изменение знака числа				+	+			+	+		+	+	+	+	+
Округление									+					+	
Разрядность чисел	8	8	8	8	8	8	8	8	10	8	16	16	12	12	12

вторно нажать клавишу [=], то произойдет выполнение хранящейся в памяти МК операции над содержимым индикатора и константой. Использование константы позволяет экономить время вычисления выражений, в которых один операнд повторяется, так как отпадает необходимость его повторного ввода.

Используя режим констант, важно знать, какой из двух операндов — первый или второй — в конкретной модели МК остается в памяти после выполнения двухместной операции. В большинстве моделей МК (к ним относятся арифметические МК Б3-23, Б3-24, Б3-26, МК-44 и др.) константой становится второй операнд.

Пример 1.3.7. Рассмотрим выполнение программы:

2 [×] 5 [=] [=] [=]

на МК Б3-26. Последовательность действий приведена в таблице 1.6.

После первого нажатия клавиши [=] в индикаторе появляет-

Таблица 1.6

Ввод	X	Y
2	2	0
[×]	2	2
5	5	2
[=]	10	5
[=]	50	5
[=]	250	5

ся первое произведение (10), а предыдущее содержимое индикатора — второй операнд 5 — переходит в регистр Y и становится константой. Последующие нажатия клавиш [=] домножают содержимое регистра X на константу, что дает последовательно 50 и 250. Рассмотренный пример показывает, что троекратное нажатие клавиши [=] в данном случае привело к возведению в третью степень второго множителя, т. е. было вычислено выражение: $2\times5^3=250$. В том, что константой становится в данном случае именно второй операнд, легко убедиться, переставив операнды местами:

$$5\ [\times]\ 2\ [=]\ [=]\ [=].$$

Выполнение этой программы даст совершенно другой результат: $5\times2^3=40$.

Важно заметить, что в некоторых моделях МК арифметического типа константная автоматика срабатывает не только при нажатии клавиши [=], но и при нажатии клавиш [+], [−], [×], [÷]. В этом смысле выполнение, например, программ:

$$2\ [\times]\ 3\ [=]\ [=]\ [=] \quad \text{и}$$
$$2\ [\times]\ 3\ [\times]\ [\times]\ [\times]$$

даст одинаковый результат. Отметим два поучительных примера простейших вычислений, основанных на использовании константной автоматики.

Пример 1.3.8. Возведение в целую положительную степень. Пусть требуется вычислить a^n, где $n\in N$. Вычисление обеспечивается программой:

$$a\ \underbrace{[\times]\ [=]\ [=]\ \ldots\ [=]}_{n-1\text{ раз}}.$$

На МК, у которых режим констант вводится также и клавишами арифметических действий, программа может быть более удобной:

$$a\ \underbrace{[\times]\ [\times]\ \ldots\ [\times]}_{n\text{ раз}}.$$

Рассмотрим для примера вычисление 12^5. Программа будет иметь вид:

$$12\ [\times]\ [\times]\ [\times]\ [\times]\ [\times].$$

Ход вычислений виден из таблицы 1.7.

Таблица 1.7

Ввод	X	Y
12	12	0
×	12	12
×	144	12
×	1 728	12
×	20 736	12
×	248 832	12

Пример 1.3.9. Вычисление обратной величины числа.

Клавиатура некоторых арифметических МК имеет клавишу [1/x], с помощью которой обратная величина находится автоматически по программе: x [1/x]. Если такой клавиши нет, то первое, что приходит в голову, это выполнить программу:

$$1\ [\div]\ x\ [=].$$

Способ вычисления неудобен из-за необходимости набирать на клавиатуре два числа (1 и x). Это неудобство особенно неприятно, если x уже имеется в регистре X (например, как результат предыдущих вычислений). Более рационально задача может быть решена программой:

$$x\ [\div]\ [\div]\ [\div].$$

Неожиданность приема объясняется константной автоматикой. Приводим для примера ход вычислений при нахождении значения $^1/_5$ (таблица 1.8).

Таблица 1.8

Ввод	X	Y
5	5	0
÷	5	5
÷	1	5
÷	0,2	5

После второго нажатия клавиши [÷] на индикаторе появляется 1, что дает при последующем нажатии этой клавиши число, обратное 5.

Контрольные вопросы

1. Какие клавиши используются для ввода чисел на индикатор?
2. Каким способом очищаются операционные регистры X и Y?
3. Каково назначение клавиши [↔]?
4. Какова последовательность действий вычислителя при выполнении на МК арифметических операций сложения, вычитания, умножения и деления?
5. Какова последовательность перемещения чисел в операционных регистрах X и Y в процессе выполнения арифметической операции?
6. В чем состоит второе назначение клавишей [+], [−], [×], [÷]?
7. В чем заключается режим констант и как он может быть использован в вычислениях?

Упражнения

1. Заполнить таблицу 1.9, проследив с помощью клавиши [↔] за содержимым регистра Y.

Таблица 1.9

Ввод	X	Y
23,3 [−] 8,5 [=]		

На основании таблицы сделать вывод о том, какое из чисел помещает МК в регистр Y при нажатии клавиши [=].

2. Составить программы вычисления значений выражений вида:

а) $(u \pm v) \times t$; в) $u \times v/t$;
б) $(u \pm v)/t$; г) $(u \pm v) \times t \pm w$.

3. Составить программы вычисления значений выражений, используя клавишу [/—/]:

а) $u - v \times t$; б) $u - v/t$.

4. Составить программы вычисления значений выражений, 1) используя клавишу [↔]; 2) используя дополнительный регистр памяти:

а) $u/(v+t)$; в) $u - v \times t$;
б) $u - v/t$; г) $u - v/(t-w)$;
д) $u - (v+t)/w - g$;
е) $(u/(v \times t + w) - g)/v - r$.

5. Заполнить таблицу 1.10, проследив с помощью клавиши [↔] за содержимым регистра Y.

Таблица 1.10

Ввод	X	Y
16 [−] 3 [=] [=] [=]		

На основании таблицы сделать вывод о том, какой из операндов — первый или второй — становится в данном МК константой.

6. Составить программы вычисления значений выражений, учитывая константную автоматику:

а) a^3b; б) x/y^5; в) a^3b^4.

7. По заданной программе восстановить выражение, значение которого вычисляется:

а) a [×] [=] [+] b [÷] [↔] c [=];

б) a [÷] b [↔] [=] [=] [−] c [↔] [=].

1.4. Микрокалькуляторы инженерного типа

Рассмотрение вычислительных возможностей микрокалькулятора инженерного типа (табл. 1.15) начнем с модели «Электроника Б3-18» — первого наиболее массового отечественного образца МК, предназначенного для научно-технических расчетов. Именно на примере этой модели МК широкие круги пользователей смогли впервые реально ощутить уровень достижений в области портативной вычислительной техники. Изготовленные массовым тиражом «Электроника Б3-18» вместе со своими модификациями Б3-18А и Б3-18М и сейчас еще имеют широкое применение. Клавиатура микрокалькулятора показана на рисунке 7. Если судить по надписям на клавишах, МК Б3-18 обладает всеми возможностями рассмотренного выше арифметического МК. Однако возможности МК этого класса значительно богаче: в них предусмотрено непосредственное вычисление натуральных и десятичных логарифмов, антилогарифмов, тригонометрических и обратных тригонометрических функций, корней, степеней обратных величин действительных чисел. Характерной для многих моделей инженерных МК особенностью является наличие клавиши совмещенной функции [F], позволяющей использовать каждую клавишу для выполнения двух операций — в соответствии с ее прямым назначением, обозначенным на самой клавише, а также согласно назначению, обозначенному над клавишей. В обычном режиме каждая клавиша используется в соответствии с обозначением, сделанным на самой клавише. При желании использовать клавишу для выполнения функции, обозначенной над клавишей, сначала нажимается клавиша [F], устанавливающая режим совмещения. Снятие режима совмещения осуществляется клавишей [CF].

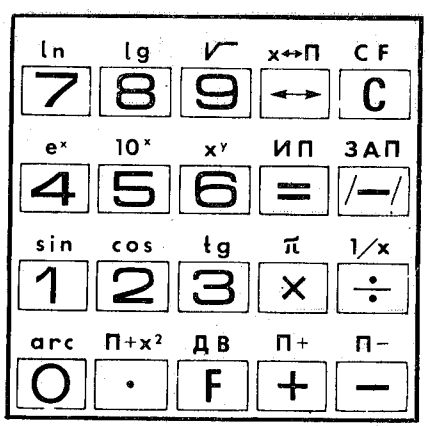


Рис. 7

При изображении вычислительных программ с помощью клавиш мы будем использовать принятое раньше их обозначение — квадрат с вписанной внутри функцией — независимо от того, изображена ли эта функция на самой клавише или над нею.

Ввод чисел в инженерных МК поизводится по тем же правилам, что и в простейших калькуляторах. Если при этом имеется клавиша [ДВ] (данные восстановлены), то ею можно воспользоваться для исправления ошибочно введенной последней цифры. Пусть вместо 3,14 ошибочно введено 3,17. Исправление делается так:

[F] [ДВ] [CF] 4.

Клавишами [F] [ДВ] устраняется ошибочно избранная цифра 7, клавиша [CF] снимает режим совмещения и позволяет продолжить ввод цифр. Разумеется, подобную процедуру имеет смысл проделывать лишь в том случае, когда ошибка возникает на позднем этапе ввода числа с большим количеством знаков.

Клавиатура инженерных МК имеет обычно специальную клавишу для ввода числа π, работающую или в прямом режиме, или в режиме совмещения. Так, в МК Б3-18 вызов на индикатор числа π обеспечивается нажатием клавиш [F] [π].

Выполнение арифметических действий. Наряду с регистрами X и Y инженерные МК имеют дополнительные регистры памяти (П), что позволяет уменьшить или совсем избежать записей промежуточных результатов в цепочках арифметических действий. В моделях МК Б3-18 имеется один дополнительный регистр памяти. Пересылка числа из регистра X (индикатора) в регистр П производится нажатием клавиш [F] [ЗАП], извлечение числа из регистра П на индикатор — командой [F] [ИП]. И в том и в другом случае стирается прежнее содержимое регистра, в который производится запись. По аналогии с клавишей [↔], осуществляющей обмен содержимым регистров X и Y, команда [F] [x↔П] производит перестановку содержимого регистров X и П. Взаимодействие регистров X, Y и П с помощью указанных команд в МК Б3-18 показано на рисунке 8. Гашение содержимого регистра П производится командой [C] [F] [ЗАП].

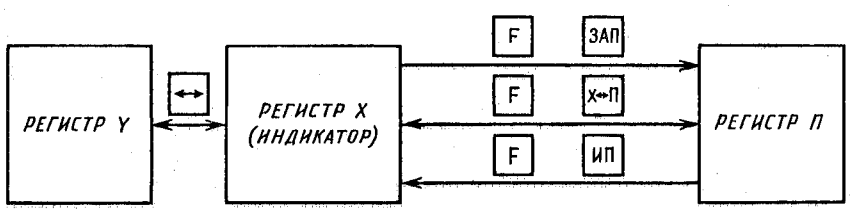


Рис. 8

Рассмотрим пример арифметического вычисления с использованием регистра П.

Пример 1.4.1. Пусть требуется найти значение выражения:

$$2{,}6\times 3{,}7+8{,}1\times 0{,}9.$$

Если регистры памяти очищены, вычисление обеспечивается программой:

2,6 [×] 3,7 [=] [F] [ЗАП] 8,1 [×] 0,9 [=] [+] [F] [ИП] [=].

Движение чисел в регистрах в ходе вычислений показано в таблице 1.11.

Таблица 1.11

Ввод	X	Y	П
2,6	2,6	0	0
[×]	2,6	2,6	0
3,7	3,7	2,6	0
[=]	9,62	3,7	0
[F] [ЗАП]	9,62	3,7	9,62
8,1	8,1	3,7	9,62
[×]	8,1	8,1	9,62
0,9	0,9	8,1	9,62
[=]	7,29	0,9	9,62
[+]	7,29	7,29	9,62
[F] [ИП]	9,62	7,29	9,62
[=]	16,91	9,62	9,62

Микрокалькулятор Б3-18 имеет клавиши [П+], [П−], [П+x²], с помощью которых регистр П можно использовать в качестве сумматора для накопления результатов. Команды

[F] [П+], [F] [П−], [F] [П+x²]

осуществляют соответственно прибавление к содержимому регистра П числа на индикаторе, вычитание из содержимого П числа на индикаторе, прибавление к содержимому П квадрата числа

на индикаторе. Использование этих клавиш позволяет вести вычисления с одновременным накоплением результатов.

Пример 1.4.2. Выражение вида:

$$x_1x_2+y_1y_2-z_1z_2$$

может быть вычислено по программе:

x_1 [×] x_2 [=] [F] [ЗАП] y_1 [×] y_2 [=]

[F] [П+] z_1 [×] z_2 [=] [F] [П−] [F] [ИП].

Пример 1.4.3. Пусть надо вычислить значение выражения:

$$a^2+b^2+c^2.$$

Задача решается программой:

a [F] [П+x²] b [F] [П+x²] c [F] [П+x²] [F] [ИП].

Вычисление значений выражений, содержащих элементарные функции. Для получения значений функций $\ln x$, $\lg x$, $\sqrt{x}$, e^x, 10^x, $\sin x$, $\cos x$, $\operatorname{tg} x$, $1/x$ на МК Б3-18 надо ввести x на индикатор, затем последовательно нажать клавишу [F] и клавишу, над которой написано название нужной функции. Например, $\lg x$ вычисляется следующим образом:

x [F] [lg]

Вычисляя значение функции x^y, МК использует представление

$$x^y=e^{y\ln x},$$

поэтому y может быть произвольным, но x обязательно должен быть положительным. Для получения x^y на МК Б3-18 выполняется программа:

x [F] [x^y] y [=].

Для вычисления значений обратных тригонометрических функций ($\arcsin x$, $\arccos x$, $\operatorname{arctg} x$) пользуются клавишей [arc]: вводят на индикатор значение аргумента x, затем нажимают клавиши [F] [arc] и одну из клавиш [sin], [cos], [tg]. Вычисления с тригонометрическими функциями могут вестись в двух режимах — для значений аргумента, выраженного в градусной или радианной мере. Для установки того или иного режима на передней панели МК имеется в этом случае специальный переключатель «град/рад».

Примеры простейших вычислений с использованием функций приведены в таблице 1.12. В тех случаях, когда положение переключателя «град/рад» безразлично, в соответствующей графе таблицы проставлен прочерк.

При вычислении сложных выражений, содержащих элементарные функции, для запоминания промежуточных результатов можно использовать регистр П. Дело, однако, осложняется тем, что при вычислении на МК значений элементарных функций (кроме $\sqrt{x}$ и $1/x$) используется регистр X. Это обстоятельство принуждает перед каждым вычислением элементарной функции предыдущий промежуточный результат пересылать в регистр П, что удлиняет процесс счета (как было отмечено выше, это не относится к вычислениям функций $\sqrt{x}$ и $1/x$).

Таблица 1.12

Выражение	«град/рад»	Программа	Ответ
sin 98°	«град»	98 [F] [sin]	0,990268
cos 0,6	«рад»	0,6 [F] [cos]	0,825336
$\sqrt{150{,}9}$	—	150,9 [F] [$\sqrt{}$]	12,284136
$2{,}6^{3{,}8}$	—	2,6 [F] [x^y] 3,8 [=]	37,74834
arcsin 0,83	«град»	0,83 [F] [arc] [sin]	56,09873
arcsin 0,83	«рад»	0,83 [F] [arc] [sin]	0,979107
cos 79°18′	«рад»	18 [÷] 60 [+] 79 [F] [cos]	0,190809

Пример 1.4.4. Надо вычислить значение:

$$\frac{8{,}3}{\sqrt{37{,}9}}.$$

Задача решается по программе:

8,3 [÷] 37,9 [F] [$\sqrt{}$] [=]

(Ответ: 1,3482129.)

Пример 1.4.5. Для вычисления аналогичного по виду выражения

$$\frac{0{,}6}{\sin 0{,}84}$$

потребуется уже иная по характеру программа:

0,84 [F] [sin] [F] [ЗАП] 0,6 [÷] [F] [ИП] [=].

(Ответ: 0,8057552.)

Заметим, что в данном случае можно составить более короткую программу, если воспользоваться клавишей [↔]:

0,84 [F] [sin] [÷] 0,6 [↔] [=].

Впрочем, во многих случаях при вычислении сложных выражений с функциями на МК с ограниченными возможностями запоминания целесообразнее выписать промежуточный результат на бумагу вместо того, чтобы терять время на обдумывание и реализацию чрезмерно хитроумной программы. Последнее оправдывается, пожалуй, лишь при неоднократном использовании таких программ для различных исходных данных.

Последующее совершенствование функциональных свойств инженерных микрокалькуляторов привело к реализации новых дополнительных возможностей: расширение числового диапазона путем удлинения разрядной сетки и введения плавающей формы представления чисел; увеличение числа дополнительных регистров памяти; исключение режима совмещения клавиш, что экономит время вычислителя, но приводит к расширению клавиатуры; введение приоритета операций, а также скобок, регулирующих порядок выполнения действий, что существенно облегчает выполнение цепочных (или, как говорят иногда, смешанных) вычислений. В той или иной мере перечисленные возможности реализованы на разных моделях современных микрокалькуляторов инженерного типа, таких, как МКШ-2, МК-41, МК-45 и др. Рассмотрим более подробно характерные особенности одного из них — настольного микрокалькулятора «Электроника МК-41».

Микрокалькулятор МК-41 имеет 14-разрядный индикатор, предусматривающий два режима представления чисел — с естественным размещением запятой и с плавающей запятой (см. п. 1.2). Левый крайний разряд, как обычно, предназначен для изображения знака «—» отрицательного числа (знак «+» не высвечивается). Для изображения значащей части числа используются следующие десять разрядов индикатора; последние три разряда в режиме с естественным размещением запятой не задействуются. Отсюда следует, что этот режим допускает работу с числами из довольно-таки внушительного диапазона: от $\pm 10^{-9}$ до $\pm(10^{10}-1)$.

Для работы с числами, выходящими из указанных границ, используется представление чисел в форме с плавающей запятой. В этом случае десять разрядов, вмещающих значащую часть числа, используются для изображения мантиссы, а следующие

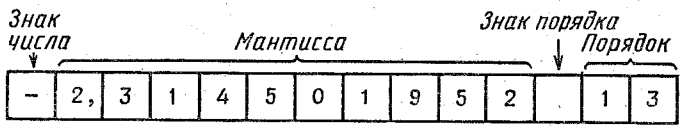


Рис. 9

три — для изображения знака и не более чем двухзначной величины порядка (на рисунке 9 показан пример изображения на индикаторе МК-41 числа $-2{,}314501952\cdot10^{13}$; знаковый разряд порядка не занят, потому что, как и при изображении знака самого числа, знак «+» не высвечивается). Очевидно, что использование плавающей формы представления чисел сильно расширяет формальный диапазон их представления; технические возможности калькулятора допускают оперирование с 10-разрядными положительными и отрицательными числами в диапазоне от $\pm10^{-99}$ до $\pm10^{99}$.

Клавиатура МК-41 работает без режима совмещения и имеет 36 клавиш (рис. 10); на передней панели размещен переключатель «град/рад». Для ввода чисел используются цифровые клавиши от [0] до [9], клавиша десятичной запятой [,], клавиша изменения знака [/—/] и клавиша ввода порядка [ВП]. При вводе числа в форме с плавающей запятой после ввода мантиссы нажимается клавиша [ВП] и цифры порядка. Если порядок отрицателен, нажимается клавиша [/—/] (или сразу после клавиши [ВП], или после набора первой или второй цифры порядка).

Пример 1.4.6. Для ввода на индикатор МК-41 числа, изображенного на рисунке 9, клавиши можно нажать в следующем порядке:

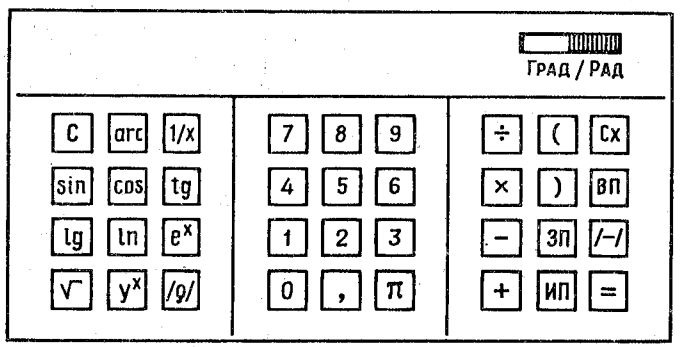


Рис. 10

[2] [,] [3] [1] [4] [5] [0] [1] [9] [5] [2] [/—/] [ВП] [1] [3].

Следует заметить, что в МК-41 при вводе отрицательной мантиссы знак «—» не может вводиться первым: клавишу [/—/] следует нажимать после любой по порядку цифры мантиссы. Для гашения содержимого индикатора в МК-41 имеется специальная клавиша [Cx]; ею можно пользоваться, в частности, при исправлении ошибки ввода путем повторного набора числа.

Как отмечено выше, разрядная сетка МК-41 вмещает 10-разрядные числа. Не всегда, однако, в вычислениях требуется сохранять большое количество цифр после запятой. Для регулирования количества десятичных знаков после запятой в МК-41 предусмотрена возможность установки любого желаемого режима отображения чисел на индикаторе. При использовании представления чисел с естественным размещением запятой это достигается путем последовательного нажатия клавиш [arc] [,] [n], где [n] — одна из цифровых клавиш, определяющая количество десятичных знаков после запятой. Заметим, что при включении калькулятора индикатор автоматически устанавливается в режим [arc] [,] [2], т. е. после запятой высвечивается два десятичных знака. Перевод в режим чисел с плавающей запятой производится командой [arc] [n], где [n] — одна из цифровых клавиш, определяющая количество десятичных знаков после запятой в мантиссе M при условии, что $1\leqslant M<10$.

Замечательной особенностью рассмотренных режимов индикации чисел в МК-41 является то, что изображение результатов вычислений с установленным количеством разрядов после запятой сопровождается округлением младшего сохраняемого разряда по правилу: если первая из отбрасываемых цифр больше или равна 5, последняя сохраняемая цифра увеличивается на единицу. Исключением является лишь режим [arc] [9], при котором округление не производится.

Пример 1.4.7. Введем на индикатор МК-41 какое-либо число, например 27,80974356. Задавая теперь разные режимы индикации, можно пронаблюдать на индикаторе соответствующие им представления заданного числа (табл. 1.13.), в том числе и процедуру автоматического округления.

Примеры простейших вычислений на МК-41 приведены в таблице 1.14 (все вычисления даны в режиме [arc] [,] [2]).

Основные двухместные арифметические операции [+], [−], [×], [÷] выполняются на МК-41 обычным образом: сначала вводится первое число, затем нажимается клавиша соответствующей операции и вводится второе число. Результат операции получается после нажатия клавиши [=]. В МК-41 имеются еще две двухместные операции: возведение числа в степень (клавиша [y^x]) и извлечение квадратного корня из суммы квадратов (клавиша [/ρ/]);

Таблица 1.13

Режим индикации	Представленные числа
[arc] [,] [2]	27,81
[arc] [,] [3]	27,810
[arc] [,] [4]	27,8097
[arc] [2]	2,78 01
[arc] [6]	2,780974 01

Таблица 1.14

Выражение	«град/рад»	Программа	Ответ
$1{,}267+0{,}38$		1,267 [+] 0,38 [=]	1,65
$34{,}7:8{,}49$		34,7 [÷] 8,49 [=]	4,09
$3{,}8^{0{,}4}$	—	3,8 [y^x] 0,4 [=]	1,71
$\sqrt{0{,}71^2+5{,}4^2}$	—	0,71 [/ρ/] 5,4 [=]	5,45
$\sin 56^\circ$	«град»	56 [sin]	0,83
tg 2,46	«рад»	2,46 [tg]	−0,83
arccos 0,83	«рад»	0,83 [arc] [cos]	0,59
lg 28,7	—	28,7 [lg]	1,46

порядок выполнения этих операций понятен из третьей и четвертой строк таблицы 1.14. Что же касается вычисления значений имеющихся на клавиатуре элементарных функций, то порядок этих действий также обычен, а отсутствие режима совмещения клавиш упрощает действия вычислителя.

Микрокалькулятор МК-41 имеет три дополнительных (адресуемых) регистра памяти для хранения исходных данных, промежуточных результатов или констант, неоднократно используемых в ходе вычислений. Дополнительные регистры памяти имеют номера 1, 2 и 3. Для обращения к этим регистрам используются клавиши [ЗП] (запись числа из регистра X в дополнительный регистр памяти) и [ИП] (вызов числа из дополнительного регистра памяти в регистр X).

Для записи в дополнительный регистр памяти числа, хранящегося на индикаторе, необходимо нажать клавишу [ЗП] и одну из цифровых клавиш [1], [2] или [3], определяющих номер адресуемого дополнительного регистра. Для вызова числа из дополнительного регистра памяти следует нажать клавишу [ИП] и одну из клавиш [1], [2] или [3]. При этом следует помнить, что при вызове числа из адресуемого регистра памяти содержимое этого регистра сохраняется, а при записи нового числа в адресуемый регистр его прежнее содержимое пропадает.

Особенностью 1-го дополнительного регистра памяти в МК-41 является то, что его содержимое может использоваться в качестве второго операнда любой двухместной операции как константа.

Пример 1.4.8. Пусть надо выполнить ряд двухместных операций, в которых вторым операндом является одно и то же число 3:

$$5+3,\ 4\times 3,\ 6:3,\ 7^3.$$

Введем вначале число 3 в регистр X и перепишем его значение в 1-й дополнительный регистр памяти:

3 [ЗП] 1.

Теперь сразу вслед за этим можно провести все требуемые действия, не вводя уже на индикатор значение второго операнда:

5 [+] [=]. (Ответ: 8.) 4 [×] [=]. (Ответ: 12.)

6 [÷] [=]. (Ответ: 2.) 7 [y^x] [=]. (Ответ: 343.)

Вместе с тем калькулятор МК-41 обладает рядом новых

Таблица 1.15

Инженерные калькуляторы		С автономным источником питания									Настольные		
Функциональные возможности		Б3-18А	Б3-18М	Б3-37	Б3-19М	Б3-32	Б3-35	Б3-36	Б3-15	Б3-38	МКШ-2	МК-41	МК-45
4 арифметических действия, изменение знака числа, действия с константой		+	+	+	+	+	+	+	+	+	+	+	+
Вычисления с применением скобок						+	+	+	+	+	+	+	+
Обмен содержимым операционных регистров		+	+	+	+		+	+					
Математические функции	x^2					+							
	x^y (y^x), $1/x$, $\sqrt{x}$	+	+	+	+	+	+	+	+	+	+	+	+
	$\sqrt{x^2+y^2}$							+				+	
	$x^{1/y}$									+			
	$\ln x$, $\lg x$, e^x	+	+	+	+	+	+	+	+	+	+	+	+
	10^x	+	+	+	+	+	+	+		+	+		
	тригонометрические (прямые и обратные)	+	+	+	+	+	+	+	+	+	+	+	+
Представление аргумента тригонометрических функций	в радианах	+	+	+	+	+	+	+	+	+	+	+	+
	в градусах	+	+	+		+	+	+		+	+	+	+
	в градах									+			
Решение систем линейных уравнений с двумя неизвестными						+					+		
Решение квадратного уравнения						+					+		
Число π		+	+	+	+	+	+	+	+	+	+	+	+
Факториал $n!$							+	+		+			+
Работа с отдельным регистром памяти	запись	+	+	+	+	+	+	+	+	+	+	+	+
	сложение	+	+	+		+	+	+		+	+		+
	вычитание	+	+	+			+	+		+			+
	умножение						+	+					+
	деление						+	+					+

характерных особенностей, которые мы рассмотрим подробнее.

Иерархия операций. Для выполнения промежуточной двухместной операции в МК-41, как и в других калькуляторах, не обязательно нажимать клавишу [=]. При выполнении смешанных вычислений клавиша [=] нажимается, как правило, в конце задачи. В промежуточных действиях вместо [=] можно нажимать клавиши следующих по формуле двухместных операций, однако исполнение операций производится с учетом установленного логикой МК старшинства операций. Самыми старшими из двухместных операций в МК-41 являются операции возведения в степень и извлечение квадратного корня из суммы квадратов двух чисел. Затем следуют операции умножения и деления, а за ними — сложения и вычитания. Если вслед за двухместной операцией нажимается клавиша операции того же или меньшего старшинства, то предыдущая двухместная операция исполняется. В противном случае МК только запоминает введенные числа и операции до тех пор, пока не будет нажата клавиша с меньшим или равным приоритетом, или клавиша [=].

Введение иерархии операций серьезно облегчает действия вычислителя. При выполнении цепочных вычислений на таких МК уже нет надобности обращаться к осмысленному запоминанию результатов промежуточных действий, а достаточно нажимать клавиши в той последовательности, которая определена самой формулой (табл. 1.15).

Пример 1.4.9. Рассмотрим порядок вычислений по двум простым и похожим с виду формулам: $2\times3+5$ и $2+3\times5$. И в том и в другом случае программа вычислений на МК-41 будет состоять из указанной в самих формулах последовательности действий:

2 [×] 3 [+] 5 [=] (Ответ: 11.)

и

2 [+] 3 [×] 5 [=]. (Ответ: 17.)

Однако с учетом старшинства операций нажатие клавиши [+] в первой программе приводит к исполнению предыдущей операции умножения, в то время как нажатие клавиши [×] во второй программе приводит лишь к запоминанию компонентов предыдущей операции сложения. Особенности вычислений по этим двум формулам хорошо просматриваются при сравнении показаний индикатора (табл. 1.16).

Новые возможности МК-41 позволяют избегать использования

3. Чем различаются действия, осуществляемые нажатием клавиш [П+] и [ЗАП]?

4. Каков порядок нажатия клавиш при вычислении значений элементарных функций?

5. Каков порядок ввода чисел в форме с плавающей запятой?

6. Как производятся цепочные действия на МК: а) с установленной иерархией арифметических операций; б) с использованием клавиш [(] и [)]?

Упражнения

1. На индикатор МК Б3-18 вместо 328,4 ошибочно введено 328,6. Составить программу исправления ошибки ввода с использованием клавиши [ДВ].

2. Составить программы вычисления значений выражений, используя дополнительные регистры памяти:

а) $ax+by$; в) $(a+x)^2+(b+y)^2$;
б) $(a+x)(b+y)$; г) $(a+x)^2/(b^2-y^2)$.

3. Как установить с помощью МК, что больше: e^π или π^e?

4. Составить программы вычисления значений выражений, используя клавишу [↔] и дополнительный регистр памяти:

а) $\frac{x}{\cos y}$; в) $a^{\sqrt{b}}$;

б) $\frac{\sqrt{x}}{\cos y}$; г) $\frac{\arccos x+e^y}{\ln(x+y^2)}$.

Составить аналогичные программы для МК с иерархией операций и скобками.

1.5. Программируемые микрокалькуляторы

Первой отечественной моделью программируемых микрокалькуляторов (ПМК) является «Электроника Б3-21». В последующих моделях ПМК этого типа (Б3-34, МК-54 — с автономными источниками питания, МК-46, МК-56 — с питанием от сети переменного тока) функциональные возможности значительно расширены, но вместе с тем логика и входной язык у всех этих моделей имеют много общего.

Главное отличие программируемых МК от арифметических и инженерных микрокалькуляторов состоит в том, что они допускают ввод и автоматическое исполнение заранее составленной последовательности команд — *программы,* включающей как арифметические (вычислительные), так и логические (проверка условий, выбор направления дальнейших вычислений и т. п.) действия.

Умелое составление программ для МК этого класса позволяет во многих случаях с помощью весьма короткой последовательности команд заставить ПМК выполнять большое количество вычислительных операций, что значительно повышает эффективность расчетов.

На программируемых калькуляторах можно вести и обычные расчеты, без составления и ввода программы. Коротко коснемся этого режима работы ПМК лишь для того, чтобы подчеркнуть особенность общей для всех ПМК вычислительной логики, основанной на использовании так называемой *обратной польской записи.* В 1921 г. польский математик и логик Ян Лукасевич, разрабатывая особый язык для формализации логических и математических выражений, предложил размещать знаки арифметических операций перед или после операндов. Первый из этих способов стали называть прямой, а второй — обратной польской записью. В обратной польской записи, например, выражение $a+b$ принимает вид: $ab+$. Применение такого представления выражения для МК обусловлено более простой технической реализацией вычислений («Электроника Б3-19»).

Клавиатура одной из современных моделей ПМК «Электроника МК-56» показана на рис. 11. Содержимое регистра X, как обычно, отображается на индикаторе. Для пересылки содержимого регистра X в рабочий регистр Y используется клавиша [В↑]. При выполнении арифметической операции на ПМК необходимо: 1) ввести в X первое число; 2) клавишей [В↑] переслать первое число в регистр Y; 3) ввести в регистр X второе число; 4) нажать

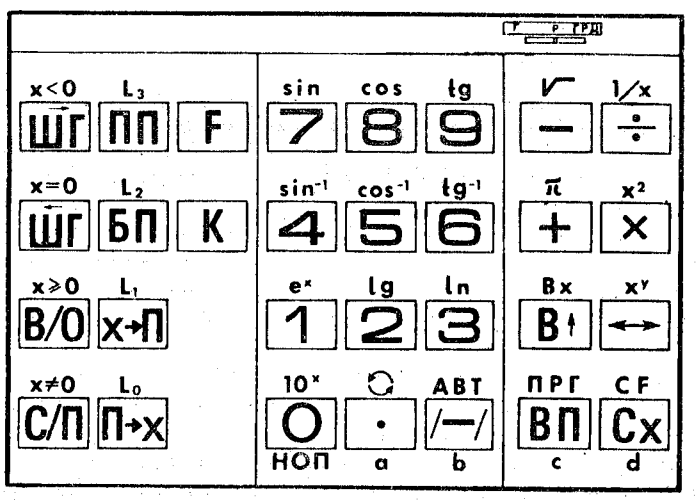


Рис. 11

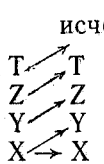
одну из клавиш [+], [−], [×], [÷]. Как видно, клавиша [=] в данном случае оказывается ненужной (ее нет и на клавиатуре). Вычисление значений одноместных элементарных функций производится обычным образом: в регистр X вводится аргумент, нажимается клавиша [F] и нужная клавиша с обозначением

Таблица 1.17

Задача	Программа	Индикатор	
sin 0,9	0,9 [F] [sin]	7,8332691	−01
arccos 0,2	0,2 [F] [cos⁻¹]	1,3694383	
$2{,}46^2$	2,46 [F] [x^2]	6,0516	
$10^{-0{,}41}$	0,41 [/−/] [F] [10^x]	3,8904512	−0,1
0,34 + 9,72	0,34 [B↑] 9,72 [+]	10,06	
12,4 × 8,5	12,4 [B↑] 8,5 [×]	105,4	

функции ($\sin^{-1}$, $\cos^{-1}$, tg^{-1} обозначают соответствующие обратные тригонометрические функции). Примеры простейших вычислений на МК-56 приведены в таблице 1.17 (аргумент в радианах). Несмотря на некоторую необычность порядка ввода данных при выполнении двухместных операций, обратная польская запись в ряде случаев оказывается даже более удобной для вычислений. Однако, чтобы понять эти особенности работы ПМК, необходимо познакомиться с организацией стековой памяти.

Стековую память МК-56 образуют четыре регистра: известные уже нам операционные регистры X и Y, а также еще два регистра Z и T. Ввод чисел всегда производится в регистр X. При нажатии клавиши [B↑] копия числа из регистра X передается в регистр Y, то, что было в Y, пересылается в Z, а прежнее содержимое Z — в регистр T. При этом содержимое регистра X сохраняется, а прежнее содержимое регистра T исчезает. Это передвижение можно изобразить так:



Указанное передвижение происходит при каждом нажатии клавиши [B↑]. Пользуясь этим, в регистры стека можно записать любые числа. Так, в результате нажатия клавиш

4 [B↑] 3 [B↑] 2 [B↑] 1

произойдет запись чисел 4, 3, 2, 1 соответственно в регистры T, Z, Y, X. Очистка всех регистров стека производится так:

[Cx] [B↑] [B↑] [B↑].

В дополнение к четырем стековым регистрам в МК-56 имеется еще один регистр, называемый *регистром предыдущего результата* (X1). Этот регистр всегда сохраняет значение числа, которое находилось в регистре X до выполнения операции. При выполнении одноместных операций ПМК оперирует с числом, находящимся в регистре X, при этом содержимое регистров Y, Z, T сохраняется, а число, находившееся до выполнения операции в регистре X, передается в регистр X1. При выполнении двухместных операций ПМК оперируют числами, находящимися в регистрах X и Y. При этом после нажатия операционной клавиши происходит обратное перемещение информации: содержимое регистра T пересылается в Z, то, что было в Z, переходит в регистр Y, в регистр X заносится результат выполненной операции, а прежнее содержимое регистра X передается в регистр X1. К сказанному нужно добавить, что если число на индикаторе является результатом предыдущих вычислений, то набор на клавиатуре нового числа автоматически передвигает информацию из регистра X в регистр Y. Таким образом, результат выполнения предыдущей операции может участвовать в качестве второго числа при выполнении последующей операции.

Пример 1.5.1. Рассмотрим с учетом указанных перемещений порядок вычислений по формуле:

$$a \times b + c \times d.$$

Эта формула, несмотря на кажущуюся простоту, является, как известно, непростым «орешком» для МК, не имеющих иерархии операций.

На ПМК вычисления по этой формуле выполняются весьма просто:

a [B↑] b [×] c [B↑] d [×] [+].

В таблице 1.18 приведено подробное описание всех перемещений при выполнении этой программы (для удобства принято, что перед вычислениями во всех регистрах стека лежат нули).

Таблица 1.18

Ввод	X	Y	Z	T
a	a	0	0	0
[B↑]	a	a	0	0
b	b	a	0	0
[×]	ab	0	0	0
c	c	ab	0	0
[B↑]	c	c	ab	0
d	d	c	ab	0
[×]	cd	ab	0	0
[+]	$ab+cd$	0	0	0

Аналогичным образом объясняется и вычисление по формуле $(a+b)(c+d)$, которое обеспечивается программой:

$$a\ [B\uparrow]\ b\ [+]\ c\ [B\uparrow]\ d\ [+]\ [\times].$$

Для хранения исходных данных и промежуточных результатов ПМК, как и калькуляторы других типов, имеют дополнительные адресуемые регистры памяти. В МК-56 таких регистров 14, они имеют номера (адреса) 0, 1, 2, ..., 9, a, b, c, d*. Запись числа из регистра X в адресуемые регистры осуществляется нажатием клавиши [x→П] и одной из клавиш, обозначающих номер адресуемого регистра. При этом число, переданное в адресуемый регистр, сохраняется в регистре X. Так, например, запись числа из регистра X в регистры R4 и Ra на МК-56 осуществляется командами:

[x→П] 4 и [x→П] a.

Для вызова числа, хранящегося в адресуемом регистре, в регистр X необходимо нажать клавишу [П→x] и клавишу, обозначающую номер адресуемого регистра.

Главное предназначение ПМК не в том, чтобы производить разовые вычисления по формуле,— для этих целей могут использоваться инженерные и арифметические калькуляторы. Особое назначение программируемых МК состоит в том, что они могут запоминать целые программы, а потом автоматически их

*В дальнейшем адресуемые регистры будем обозначать буквой R с расположенным за ним номером: R0, R1, R2 и т. д.

выполнять. Для запоминания команд программы у ПМК имеется специальная *программная память*. Программная память МК-56 состоит из 98 ячеек; первая ячейка имеет номер 00, последняя 97. Чтобы получить общее представление о программируемом использовании ПМК, рассмотрим примеры.

Пример 1.5.2. Составить программу для автоматического вычисления на МК-56 длин окружностей по заданному радиусу r.

Как известно, длина окружности вычисляется по формуле $L=2\pi r$. Программа ручных вычислений по формуле на МК-56 весьма проста:

$$2\ [F]\ [\pi]\ [\times]\ r\ [\times].$$

Предположим теперь, что значение радиуса r заранее записано в регистр X (т. е. находится на индикаторе). При этом условии действия по вычислению L сведутся к следующему:

[B↑] 2 [×] [F] [π] [×]

(перед вводом в регистр X числа 2, находившееся в нем значение радиуса r с командой [B↑] переместится в регистр Y).

Запишем теперь команды этой программы шаг за шагом в программную память ПМК. Для этого калькулятор нажатием клавиш [F] [ПРГ] переводится в режим «Программирование». В процессе пошагового ввода программы на индикаторе будет наблюдаться движение двузначных кодов команд (слева направо). При этом крайнее правое двузначное число регистрирует значение счетчика вводимых команд и принимает последовательные значения: 00, 01, 02 и т. д.*. Текущее состояние счетчика команд выражает адрес ячейки программной памяти, по которому будет записана следующая команда программы. Покомандное изложение приведенной выше программы вычисления длины окружности $L=2\pi r$ в предположении, что значение радиуса находится в регистре X, дано в таблице 1.19. Здесь же указаны адреса ячеек, хранящих команды, а также описание действий, которые производит каждая команда. Таким образом, вся программа состоит из шести команд и занимает шесть ячеек памяти с адресами: 00, 01, 02, 03, 04, 05. Когда ввод всей программы закончен, микрокалькулятор нужно вернуть в счетный режим работы, для чего нажимают клавиши [F] [АВТ] (этот режим носит название «Автоматическая работа»).

*Программу можно заносить в память с произвольного адреса. Для этого, прежде чем перейти в режим «Программирование», нужно нажать клавишу [БП], а затем ввести значение адреса первой команды.

Программа пускается нажатием клавиш [В/О] [С/П] *.

Таблица 1.19

Адрес команды	Команда	Выполняемое действие
00	[В↑]	Запись числа r в регистр Y
01	2	Ввод числа 2 в регистр X
02	[×]	Вычисление $2\times r$
03	[F] [π]	Ввод числа π в регистр X
04	[×]	Вычисление $2\times\pi\times r$
05	[С/П]	Остановка и индикация ответа в регистре X

Однако, прежде чем сделать это, необходимо в регистр индикации X записать значение радиуса (при составлении программы мы исходим из предположения, что значение радиуса находится в регистре X). Пусть, к примеру, нужно вычислить длину окружности радиуса $r=5$. Тогда после выполнения всех описанных выше действий по вводу программы нужно выполнить операции:

5 [В/О] [С/П].

На индикаторе высветится ответ: 31,415926. Если нужно получить длины окружностей для значений радиуса $r_1, r_2, \ldots, r_n$, то нужно n раз обратиться к программе, введенной в память ПМК с помощью команд:

r_i [В/О] [С/П] $(i=1, 2, \ldots, n)$.

Записанная в память ПМК программа будет храниться там до тех пор, пока на ее место не будет записана другая программа или не произойдет отключение ПМК.

На программируемых микрокалькуляторах весьма эффективно решается задача вычисления таблицы значений функции (эту часто встречающуюся практическую задачу называют еще *табулированием функции*). Задача табулирования ставится обычно следующим образом. Дается функция $f(x)$, отрезок $[a; b]$, на котором функция определена, и шаг h. Требуется вычислять значения функции в точках a, $a+h$, $a+2h$, ... до тех пор, пока очередное значение аргумента не выйдет за правую границу

*Клавиша [В/О] устанавливает нулевое значение счетчика команд, клавишей [С/П] производится пуск.

отрезка. При этом оформляется таблица вида 1.20. Поскольку в вычислениях на микрокалькуляторе вычислитель сам управляет ходом вычислительного процесса, можно считать, что для постановки задачи табулирования функции на ПМК, помимо задания аналитического выражения функции, достаточно указать лишь левую границу отрезка $x=a$ и шаг h. Фактически задача сводится к задаче табулирования на интервале $[a; \infty)$.

Таблица 1.20

x	$f(x)$
a $a+h$ $a+2h$ $a+3h$ ...	$f(a)$ $f(a+h)$ $f(a+2h)$ $f(a+3h)$ ...

Пример 1.5.3. Составить программу табулирования функции

$$f(x)=\frac{5\sin x}{x^2+1}$$

на МК-56 с начальным значением $x=1$ и шагом $h=0{,}1$.

При составлении программы будем предполагать, что значения x и h записаны соответственно в адресуемые регистры R0 и R1:

x | R0
h | R1 .

Принимая во внимание характер заданной формулы и вычислительные возможности МК-56, будем при составлении программы придерживаться следующей последовательности вычисления значения $f(x)$:

1) вычислить значение знаменателя x^2+1 и запомнить его в одном из адресуемых регистров памяти (R2);

2) вычислить значение числителя $5\sin x$;

3) вызвать в регистр X значение знаменателя из регистра R2 (значение числителя при этом перейдет в регистр Y);

4) выполнить операцию деления.

Программа вычисления одного значения $f(x)$ будет иметь вид:

[П→x] 0 [F] [x^2] 1 [+] [x→П] 2 [П→x] 0 [F] [sin] 5 [×]
[П→x] 2 [÷].

Итак, программа по существу готова. Достаточно завершить ее командой [С/П] (остановка и индикация значения $f(x)$) и можно записывать в память. Остается очевидным лишь одно неудобство: после каждого прохода этой программы (и снятия

очередного значения функции с индикатора) перед каждым новым пуском придется вручную вводить в регистр R0 новое значение аргумента x. Этого неудобства, впрочем, можно избежать, если вслед за остановкой и индикацией значения функции в программе предусмотреть автоматическое вычисление нового значения аргумента и пересылку его в регистр R0:

[П→x] 0 [П→x] 1 [+] [x→П] 0 [С/П].

Теперь остановка по команде [С/П] будет сопровождаться индикацией нового значения аргумента. Вслед за этим остается осуществить автоматический переход к началу программы. Для этой цели в системе команд МК-56 есть команда безусловного перехода [БП], вслед за которой нужно указать адрес ячейки программной памяти 00, в которой находится первая команда программы. Полностью искомая программа представлена в таблице 1.21.

Таким образом, для табулирования функции $f(x)=\frac{5\sin x}{x^2+1}$

с начальным значением аргумента $x=1$ и шагом $h=0{,}1$ с помощью программы, изложенной в таблице 1.21, необходимо:

1) ввести программу в память ПМК;

2) записать значения $x=1$ и $h=0{,}1$ в регистры R0 и R1 соответственно (командами 1 [x→П] 0 и 0,1 [x→П] 1);

3) пустить программу (команда [В/О] [С/П]).

Пункт 1) выполняется в режиме [ПРГ], пункты 2) и 3) — в режиме [АВТ]. Программа составлена так, что выходные данные индицируются на световом табло в последовательности:

$$x,\ f(x),\ x+h,\ f(x+h),\ldots .$$

При этом индикация каждого члена последовательности, начиная со второго, осуществляется в результате выполнения команды [С/П]. После каждой остановки и снятия показателя индикатора работа программы возобновляется нажатием клавиши [С/П]. Результаты работы программы табулирования для заданных в примере 1.5.3 условий частично представлены в таблице 1.22. Процесс табулирования прекращается по усмотрению самого вычислителя. Для задания нового участка или шага табулирования достаточно перед пуском программы ввести новые

Таблица 1.21

Адрес команды	Команда	Выполняемое действие
00	[П→x] 0	Вызов x в регистр X из R0
01	[F] [x^2]	Вычисление x^2
02	1	Запись числа 1 в регистр X
03	[+]	Вычисление x^2+1
04	[x→П] 2	Запись x^2+1 в регистр R2
05	[П→x] 0	Вызов x в регистр X из R0
06	[F] [sin]	Вычисление sin x
07	5	Запись числа 5 в регистр X
08	[×]	Вычисление 5 sin x
09	[П→x] 2	Вызов x^2+1 в регистр X из R2
10	[÷]	Вычисление $f(x)=5\sin x/(x^2+1)$
11	[С/П]	Остановка и индикация $f(x)$
12	[П→x] 0	Вызов x в регистр X из R0
13	[П→x] 1	Вызов h в регистр X из R1
14	[+]	Вычисление $x=x+h$
15	[x→П] 0	Запись x в регистр R0
16	[С/П]	Остановка и индикация x
17	[БП]	Безусловный переход
18	00	Адрес безусловного перехода

значения в регистры R0 и R1. Однако на этом, к сожалению, и заканчивается вариативность составленной программы. Для того чтобы перейти к табулированию другой функции, всю программу придется составлять заново. Этот недостаток програм-

мы можно исправить, если при ее организации использовать следующий подход: вычисление значения функции $f(x)$ оформить в виде самостоятельного блока — *подпрограммы,* а в основной программе к этой подпрограмме обращаться. Тогда для перехода к табулированию другой функции достаточно будет заменять только блок вычисления $f(x)$.

Таблица 1.22

x	$f(x)=\frac{5\sin x}{x^2+1}$
1	2,1036776
1,1	2,0163063
1,2	1,9099162
1,3	1,7910003
1,4	1,6646110
1,5	1,5346077
1,6	1,4038955
1,7	1,2746334
...	...

Для перехода на подпрограмму в языке МК-56 имеется специальная команда, реализуемая клавишей [ПП]. С помощью этой команды происходит переход на подпрограмму по адресу, указанному непосредственно после команды перехода, и, кроме того, запоминается адрес следующей команды основной программы. По этому адресу происходит возврат в основную программу после отработки подпрограммы. Подпрограмма всегда завершается командой возврата [В/О]. Подпрограмму вычисления $f(x)$ в общей программе табулирования естественно располагать в конце, это облегчит затем замену подпрограммы при переходе к табулированию новой функции.

В таблице 1.23 приведена программа табулирования функции с подпрограммой, реализующей вычисление значений функции $f(x)$ из примера 1.5.3. В ячейках 00—10 размещена основная программа табулирования. Подпрограмма вычисления $f(x)$ начинается с ячейки 11, с этого адреса должна располагаться новая подпрограмма вычисления значения функции при переходе к другой задаче табулирования.

При решении некоторых задач требуется автоматическое неоднократное выполнение (или, как говорят, зацикливание) отдельных участков одной и той же программы. Язык программирования ПМК предусматривает такую возможность.

Таблица 1.23

Адрес	Команда	Выполняемое действие
00	[П → x] 0	Вызов x в регистр X из R0
01	[ПП]	Переход на подпрограмму
02	11	Адрес подпрограммы
03	[С/П]	Остановка и индикация $f(x)$
04	[П → x] 0	Вызов x в регистр X из R0
05	[П → x] 1	Вызов h в регистр X из R1
06	[+]	Вычисление $x=x+h$
07	[x → П] 0	Запись x в регистр R0
08	[С/П]	Остановка и индикация x
09	[БП]	Безусловный переход к началу
10	00	Адрес безусловного перехода
11	[F] [x^2]	Вычисление x^2
12	1	Запись числа 1 в регистр X
13	[+]	Вычисление x^2+1
14	[x → П] 2	Запись x^2+1 в регистр R2
15	[П→x] 0	Вызов числа x в регистр X из R0
16	[F] [sin]	Вычисление $\sin x$
17	5	Запись числа 5 в регистр X
18	[×]	Вычисление $5\sin x$
19	[П → x] 2	Вызов x^2+1 в регистр X из R2
20	[÷]	Вычисление $f(x)=5\sin x/(x^2+1)$
21	[В/О]	Возврат к команде с адресом 03

Пример 1.5.4. Требуется найти сумму всех тех членов ряда

$$\sum_{n=1}^{\infty} \frac{1}{n^2},$$

величина которых не меньше заданного числа ε.

Выпишем несколько членов заданного ряда:

$$1+\frac{1}{4}+\frac{1}{9}+\frac{1}{16}+\frac{1}{25}+\dots .$$

Если, например, $\varepsilon=0{,}21$, то согласно условию задачи искомую сумму составят лишь два первых члена ряда: $1+\frac{1}{4}=1{,}25$.

Предположим, что значение ε находится на индикаторе (в регистре X), сумма членов ряда будет накапливаться в регистре R1 (вначале его нужно будет очистить), текущее значение номера члена n — в регистре R2, а значение очередного члена ряда в виде $1/n^2$ будет находиться в регистре R3. Исходя из этих предположений несложно составить программу, которая вычисляла бы последовательные члены заданного ряда в регистре R3, проверяла бы условие $(R3)<\varepsilon$* и, если оно не выполняется, добавляла бы значения этого члена к содержимому регистра R1. А когда окажется, что $(R3)<\varepsilon$, должна происходить остановка с индикацией содержимого регистра R1.

Как видно, чтобы реализовать эту идею, надо уметь определять истинность условия $(R3)<\varepsilon$ и в зависимости от исхода проверки либо прекращать счет, либо продолжать его. На языке команд это означает, что нужно проверить заданное условие и в зависимости от результата перейти к указанному для каждого из этих двух случаев адресу очередной команды. Программируемые МК имеют соответствующие команды. Это команды перехода по условию, которые реализуются клавишей [F] и одной из клавиш с обозначением условий: [x≥0], [x<0], [x≠0]. Каждая из этих команд проверяет содержимое регистра X на выполнение заданного условия. Если условие не выполняется, то следующей по программе будет исполнена команда, адрес которой указан непосредственно за командой условного перехода. Если условие выполняется, то следующей будет исполняться команда, записанная после адреса перехода.

Другой командой, необходимой для организации циклических программ, является команда безусловного перехода, реализуемая клавишей [БП].

*(R3) обозначает содержимое (число) из регистра R3.

В таблице 1.24 приведена программа вычисления суммы ряда по заданному числу ε, предварительно введенному в регистр X. Ход вычислений и порядок использования команд условного и безусловного перехода можно уяснить из подробных пояснений действия каждой команды, приведенных в правом столбце. Для пуска этой программы нужно ввести значение ε в регистр X и нажать клавиши:

[В/О] [С/П].

Так, при $\varepsilon=0{,}001$ результатом программы будет 1,6131908.

Мы рассмотрели далеко не все свойства программируемых МК. Эти калькуляторы предоставляют пользователю возможность решать задачи, многие из которых до недавнего времени были под силу лишь ЭВМ. Практика показывает, что для полного использования возможностей ПМК, кроме усвоения их логики и системы команд, требуется еще овладение культурой программирования.

Контрольные вопросы

1. В чем основное отличие программируемых микрокалькуляторов (ПМК) от МК арифметического и инженерного типа?
2. Каков порядок выполнения на ПМК двухместной арифметической операции?
3. Какие регистры образуют стековую память МК-56?
4. Каков порядок перемещения информации в стековых регистрах при нажатии клавиши [В↑]?
5. Какие перемещения происходят в стековых регистрах ПМК при выполнении одноместной операции?
6. Какие перемещения происходят в стековых регистрах ПМК при выполнении двухместной операции?
7. Что происходит с результатом предыдущих вычислений, хранящимся в регистре X, при вводе в X нового числа?
8. Как производится запись и вызов чисел из адресуемых регистров памяти?
9. В каком режиме производится ввод программы в программную память ПМК? В каком режиме производится пуск программы?
10. Как записываются и работают команды безусловного и условного переходов?

Таблица 1.24

Адрес команды	Команда	Выполняемое действие
00	[x → П] 0	Запись числа ε из регистра X в регистр R0
01	0	Ввод числа 0 в регистр X
02	[x → П] 1	Запись числа 0 в регистр R1 (S=0)
03	[x → П] 2	Запись числа 0 в регистр R2 (в будущем в R2 будет размещаться значение n)
04	1	Ввод числа 1 в регистр X
05	[П → x] 2	Пересылка числа 1 в регистр Y и вызов числа из регистра R2
06	[+]	Вычисление текущего значения n
07	[x → П] 2	Запись n в регистр R2
08	[F] [x^2]	Вычисление n^2
09	[F] [1/x]	Вычисление $1/n^2$
10	[x → П] 3	Запись $1/n^2$ в регистр R3
11	[П → x] 0	Пересылка $1/n^2$ в регистр Y и вызов числа ε в регистр X
12	[−]	Вычисление $1/n^2-\varepsilon$
13	[F] [x⩾0]	Проверка $1/n^2-\varepsilon<0$ (надо ли продолжать?)
14	21	Адрес перехода при $1/n^2-\varepsilon<0$ (к окончанию цикла)
15	[П → x] 1	Вызов числа S из регистра R1
16	[П → x] 3	Вызов числа $1/n^2$ из регистра R3
17	[+]	Вычисление $S+1/n^2$
18	[x → П] 1	Запись $S+1/n^2$ в регистр R1
19	[БП]	Безусловный переход
20	04	Адрес безусловного перехода
21	[П → x] 1	Вызов найденной суммы из регистра R1 в регистр X
22	[С/П]	Остановка и индикация результата

Упражнения

1. Составить программы для однократных вычислений на МК-56 значений следующих выражений:

а) $3{,}18\sqrt{24{,}76}$; в) $24{,}3^{0{,}7}+1\,/\,(\lg 0{,}261-2{,}8)$;

б) $0{,}9^3 \sin 0{,}84$; г) $9{,}71\cdot\cos(\sqrt{0{,}64}+2{,}1^{1{,}5})$.

2. Составить программу для ввода чисел a, b, c, d в регистры стека X, Y, Z, T соответственно.

3. Написать программу для ручных вычислений на ПМК общего сопротивления цепи R при параллельном включении сопротивлений R_1, R_2, R_3 и R_4 по формуле:

$$R=\frac{1}{\frac{1}{R_1}+\frac{1}{R_2}+\frac{1}{R_3}+\frac{1}{R_4}}.$$

4. Составить программу для автоматического вычисления значений площади круга $S=\pi R^2$ по заданным значениям радиуса R, вводимым в регистр индикации X.

5. Дано уравнение $3^x-4x=0$. Методом табулирования функции $f(x)=3^x-4x$ на МК-56 локализовать отрезки, содержащие корни уравнения.

Указание. Составить программу табулирования функции $f(x)$ на отрезке [0; 2] с шагом 0,1. Непрерывная функция имеет на отрезке корень, если она меняет знак на его концах. Если функция к тому же монотонна на этом отрезке, корень единственный.

6. Составить циклическую программу вычисления суммы всех членов ряда $\sum_{n=1}^{\infty}\frac{n}{e^n}$, величина которых не меньше заданного числа ε.

7. Составить циклическую программу вычисления наибольшего общего делителя (НОД) двух целых положительных чисел a и b.

Глава 2

МЕТОДЫ ОЦЕНКИ ПОГРЕШНОСТЕЙ

Рассматривая в предыдущей главе технику вычислений на микрокалькуляторах, мы не затрагивали вопросов точности вычислений. Понятно, однако, что большое количество цифр, высвечиваемых на индикаторе МК при выполнении расчетов, не делает менее актуальной задачу оценки точности результатов. Это относится и к учету точности исходных данных, и к анализу накопления вычислительных ошибок, т. е. затрагивает весь круг вопросов, традиционно относимых к методам приближенных вычислений.

2.1. Источники ошибок в вычислениях по формуле

Одной из основных причин, ограничивающих точность расчетов по формуле, является *степень точности исходных данных.* Дело в том, что используемые в вычислениях числовые значения носят в подавляющем большинстве случаев приближенный характер, ибо получаются на практике в результате измерений, взвешиваний или других способов инструментального получения числовых значений каких-либо реальных величин. В тех исключительных случаях, когда требуется особо высокая точность исходных значений, ее достижение связано обычно с немалыми затратами, в стандартных же случаях эта точность обычно не очень высока. Часто она оказывается несопоставима с теми колоссальными возможностями, которыми обладает при обработке числовых значений микрокалькулятор. Именно здесь и кроется опасность возникновения заблуждений о высокой точности современной счетной техники: например, произвели действие над числами с малым количеством цифр, а полученный ответ занял весь индикатор, 3,2:2,3=1,3913043. Грамотный пользователь не сделает из этого поспешного вывода, а подойдет к оценке точности полученного значения с учетом точности исходных данных. Другие причины, влияющие на точность вычисления,— это *технические возможности микрокалькулятора,* прежде всего конечность разрядной сетки МК. Это обстоятельство приводит к тому, что не вмещающиеся на индикаторе разряды результата (когда в ответе получается бесконечная десятичная дробь или дробь конечна, но содержит больше значащих цифр, чем разрядность индикатора) остаются в ходе вычислений неизвестными для вычислителя. В большинстве случаев МК устроен так, что эти разряды попросту отбрасываются. Допустим, что использованные в приведенном выше примере исходные значения 3,2 и 2,3 — точные числа, тогда полученное на МК их частное 1,3913043 содержит ошибку, так как его

разрядная сетка не вместила всех цифр результата. В том, что ошибка допущена, легко убедиться, проверив деление умножением:

$$1{,}3913043 \times 2{,}3 = 3{,}1999998.$$

В данном случае вычислитель никак не может судить об истинной величине допущенной погрешности, а может только сказать, что ошибка не превышает единицы самого младшего из изображенных на индикаторе разрядов результата.

Иногда, как следует из инструкций по эксплуатации, МК выдает результат с точностью, не использующей всех разрядов индикатора. Например, МК «Электроника Б3-18» при 8-разрядном индикаторе вычисляет значения элементарных функций с шестью знаками после запятой, т. е. с точностью до единицы в шестом младшем разряде. Очевидно, что этот тип погрешности обусловлен *погрешностью метода,* используемого при вычислении значений функций. К числу причин, искажающих результат вычислений, следует отнести и всевозможные *промахи,* допускаемые иногда в процессе счета: ошибочный ввод данных на индикатор, смешение клавиш операций, использование неверной программы вычислений и т. д. Сюда относятся также ошибки, возникающие из-за *сбоев,* возникающих в самом вычислительном приборе. Средством борьбы против промахов разного рода служит предварительная грубая прикидка ожидаемого результата или способ двойных вычислений, а также специально организуемые системы текущего контроля, связанные с существом решаемой задачи. Не принимая во внимание возможные случайные промахи, можно считать, что основное влияние на погрешность результата вычислений на МК по готовой формуле оказывает погрешность исходных данных. Более того, в пределах разрядной сетки МК этот фактор чаще всего оказывается единственным, так как влияние технических особенностей МК начинает сказываться лишь тогда, когда точность счета приближается к границам возможностей микрокалькулятора. Для исчерпывающего представления о погрешностях результата вычислений следует учитывать влияние всех возможных ошибок. Подобный анализ нельзя провести без использования начальных понятий о методах приближенных вычислений.

Контрольные вопросы

1. Каковы основные причины возникновения погрешностей в вычислениях по готовой формуле?
2. Какие из этих погрешностей зависят от микрокалькулятора?

2.2. Основные понятия теории приближенных вычислений

Если X — истинное значение некоторой величины, а x — ее известное приближение, то абсолютная величина ошибки приближения x определяется так:

$$e_x = |X - x|. \qquad (2.1)$$

Величина e_x, называемая *абсолютной погрешностью* приближенного значения x, в большинстве случаев остается неизвестной для вычислителя, так как для ее вычисления нужно знать точное значение X. Вместе с тем на практике обычно удается установить верхнюю границу абсолютной погрешности, т. е. такое (по возможности наименьшее) число Δx, для которого справедливо

$$|X - x| \leqslant \Delta x. \qquad (2.2)$$

Число Δx в этом случае называют *границей абсолютной погрешности* (или предельной абсолютной погрешностью) приближения x.

Пример 2.2.1. Рассмотрим число $\pi = 3{,}14159265358...$. Если же вызвать π на индикатор 8-разрядного МК, получим приближение этого числа:

$$\pi' = 3{,}1415926.$$

Абсолютная погрешность значения π':

$$e_{\pi'} = |\pi - \pi'| = 0{,}00000005358... .$$

Получили бесконечную дробь, непригодную для практических расчетов. Но

$$e_{\pi'} < 0{,}00000006 = \Delta\pi',$$

следовательно, число $\Delta\pi' = 0{,}6 \cdot 10^{-7}$ можно считать границей абсолютной погрешности приближения, используемого микрокалькулятором вместо числа π.

Неравенство (2.2) позволяет установить приближения к точному значению X по недостатку и по избытку:

$$|x - \Delta x| \leqslant X \leqslant |x + \Delta x|, \qquad (2.3)$$

которые могут рассматриваться как возможные значения соответственно нижней (НГ) и верхней (ВГ) границ приближения x:

$$\text{НГ}_x = x - \Delta x, \ \text{ВГ}_x = x + \Delta x. \qquad (2.4)$$

Качество приближенных значений измеряется с помощью относительной погрешности, которая определяется как отношение ошибки e_x к модулю значения X (когда оно неизвестно — к модулю приближения x). Границей относительной погрешности δx приближенного числа называется отношение предельной абсолютной погрешности к абсолютному значению приближения x:

$$\delta x = \frac{\Delta x}{|x|}. \qquad (2.5)$$

Относительную погрешность выражают обычно в процентах.

Пример 2.2.2. Вычислим границу относительной погрешности приближения к числу π, используемого микрокалькулятором:

$$\delta\pi' = \frac{0{,}6 \cdot 10^{-7}}{3{,}1415926} < 0{,}2 \cdot 10^{-7},$$

т. е. можно принять $\delta\pi' = 0{,}000002\%$. Это чрезвычайно высокая точность, если учесть, что для ординарных технических расчетов считается приемлемым уровень точности от 0,1 до 5%.

Цифра числа называется *верной* (в широком смысле), если абсолютная погрешность числа не превосходит единицы разряда, в котором стоит эта цифра. *Значащими цифрами* числа, записанного в виде десятичной дроби, называют все его верные цифры, начиная с первой слева, отличной от нуля. Запись приближенного числа только верными знаками принято называть *правильной записью.*

Пример 2.2.3. Приближенное число $a = 190{,}30700$ дано в правильной записи, т. е. в числе восемь значащих цифр, а его абсолютная погрешность не превышает единицы 5-го разряда после запятой, поэтому можно принять $\Delta a = 0{,}00001$.

Исключение из последующих вычислений неверных цифр производится путем *округления* приближенных чисел, т. е. замены числа его значением с меньшим количеством значащих цифр. При округлении возникает погрешность, называемая *погрешностью округления.* Пусть x — данное число, а x_1 — результат его округления. Погрешность округления определяется как модуль разности прежнего и нового значений числа:

$$\Delta_{\text{окр}} = |x - x_1|. \qquad (2.6)$$

В отдельных случаях вместо $\Delta_{\text{окр}}$ приходится использовать ее верхнюю оценку.

Пример 2.2.4. Выполним на МК действие 1 $\boxed{\div}$ 6. На индикаторе высветится 0,1666666. Произошло автоматическое округление бесконечной десятичной дроби $0{,}1(6)$ до количества разрядов, вмещающихся в регистре МК. При этом можно принять $\Delta_{\text{окр}} = 0{,}7 \cdot 10^{-7}$.

Рассмотренный пример «принудительного» округления называют *округлением методом отбрасывания.* Очевидно, что сам по себе метод отбрасывания оставляет все сохраняемые цифры округленного числа верными. Если вычисления ведутся в пределах разрядной сетки МК, т. е. с точностью меньшей, чем машинная точность микрокалькулятора, целесообразнее пользоваться способом *симметричного округления,* который приводит к меньшей величине ошибки округления, чем способ отбрасывания. Симметричное округление выполняется по следующим правилам:

1. Если первая слева из всех отбрасываемых цифр меньше 5, то последняя сохраняемая цифра остается без изменения.

2. Если первая слева из всех отбрасываемых цифр больше или равна 5, то последняя сохраняемая цифра увеличивается на единицу.

Из правил симметричного округления следует, что его погрешность не превышает половины единицы последнего сохраняемого разряда. Это обстоятельство позволяет вести счет с точностью большей, чем единица последнего сохраняемого разряда. По этой причине наряду с понятием «верная цифра», соответствующим методике округления путем отбрасывания, используется понятие «цифра, верная в строгом смысле», применяемое в вычислениях с симметричным округлением.

Цифра числа называется *верной в строгом смысле,* если абсолютная погрешность этого числа не превосходит половины единицы разряда, в котором стоит эта цифра.

Пример 2.2.5. Вычислим на МК $x=\sqrt{236}$. Получим $x=15{,}362291$. Округлим результат до десятых методом симметричного округления: $x_1=15{,}4$; $\Delta x_1=0{,}04$. Все цифры числа x_1 верны в строгом смысле. Абсолютная погрешность числа x_1, получаемого в результате округления приближенного значения x, складывается из абсолютной погрешности первоначального числа x и погрешности округления. Действительно, из неравенства

$$|X-x_1|\leqslant|X-x|+|x-x_1|\leqslant\Delta x+\Delta_{\text{окр}}$$

следует, что если в результате округления приближенного числа x получено значение x_1, то границей абсолютной погрешности числа x_1 можно считать сумму границы абсолютной погрешности числа x и погрешности округления.

Пример 2.2.6. Пусть в приближенном значении $a=16{,}395$ все цифры верны в широком смысле. Округлим a до сотых: $a_1=16{,}40$. Погрешность округления $\Delta_{\text{окр}}=0{,}005$. Для нахождения полной погрешности Δa_1 нужно сложить $\Delta_{\text{окр}}$ с погрешностью исходного значения a_1, которая в данном случае может быть найдена из условия, что все цифры в записи a верны: $\Delta a=0{,}001$. Таким образом, $\Delta a_1=\Delta a+\Delta_{\text{окр}}=0{,}001+0{,}005=0{,}006$. Отсюда следует, что в значении $a_1=16{,}40$ цифра 0 неверна в строгом смысле.

Контрольные вопросы

1. Что такое абсолютная погрешность приближенного значения величины? граница абсолютной погрешности?
2. Как с помощью границы абсолютной погрешности Δx приближенного значения x можно указать возможные значения его нижней и верхней границ?
3. Что такое относительная погрешность приближенного значения величины? граница относительной погрешности?
4. Какие цифры в записи приближенного числа называются верными в широком смысле? верными в строгом смысле?

5. Какие цифры в записи приближенного числа называются значащими?
6. Что такое округление числа? погрешность округления?
7. Как различаются погрешности симметричного округления и округления методом отбрасывания?
8. Из чего складывается полная погрешность округленного числа?

Упражнения

1. В результате измерения длины стола метром с сантиметровыми делениями установлено, что значение длины находится между делениями 63 и 64 см. Указать границы абсолютной и относительной погрешности значения длины, если принять ее равной 63,5 см.
2. У приближенных чисел

$$36{,}7;\ 2{,}489;\ 31{,}010;\ 0{,}031$$

все цифры верны в строгом смысле. Указать границы абсолютных и относительных погрешностей этих чисел.
3. У приближенных чисел

$$0{,}310;\quad 8{,}495;\quad 24{,}3790$$

все цифры верны в строгом смысле. Округлить заданные числа до сотых и определить в округленных значениях количество цифр, верных в строгом смысле.
4. Приближенное значение массы Земли равно $(5{,}98\pm0{,}001)\times\times10^{24}$ кг. Масса пули охотничьего ружья равна (16 ± 1) г. Какое измерение является более точным?

2.3. Определение количества верных цифр по относительной погрешности приближенного числа

Количество верных значащих цифр в приближенном числе и величина относительной погрешности этого числа взаимосвязаны: по величине δx, учитывая формулу (2.5), можно вычислять абсолютную погрешность

$$\Delta x=|x|\cdot\delta x, \tag{2.7}$$

величина которой, как следует из определения границ верных значащих цифр, явно влияет на их количество в приближенном числе. На практике иногда удобнее пользоваться правилом, устанавливающим взаимосвязь количества верных цифр непосредственно с величиной относительной погрешности. Рассмотрим этот вопрос применительно к приближенному числу x, записанному в нормальном виде (см. п. 1.2):

$$x=M\cdot10^p, \tag{2.8}$$

в предположении, что мантисса M удовлетворяет условию $0{,}1 \leqslant M < 1$ (число x в этом случае называют нормализованным). У нормализованного числа первый разряд мантиссы после запятой всегда отличен от нуля. Очевидно, что хранение в машине чисел именно в нормализованном виде позволяет сохранить больше значащих цифр мантиссы. Заметим также, что для нормализованного числа x вида (2.8) справедливо следующее:

$$|x| < 10^p. \tag{2.9}$$

Итак, имеется приближенное число x и его относительная погрешность δx. Нужно установить количество верных в строгом смысле значащих цифр в числе x. Для каждого известного значения δx можно подобрать такое наибольшее натуральное n, чтобы имело место:

$$\delta x \leqslant 10^{-n}. \tag{2.10}$$

Тогда:

$$\Delta x \leqslant |x| \cdot 10^{-n} < 10^p \cdot 10^{-n} = 10^{p-n} < \frac{1}{2} 10^{-(n-1)+p},$$

т. е.

$$\Delta x < \frac{1}{2} 10^{-(n-1)} \cdot 10^p. \tag{2.11}$$

Сопоставляя теперь (2.8) и (2.11) и используя определение цифры, верной в строгом смысле, можно сделать вывод, что в мантиссе приближенного числа x верны в строгом смысле, по крайней мере $n-1$ цифра после запятой (а поскольку x нормализовано, то все эти цифры значащие). Таким образом, для того чтобы по заданной величине относительной погрешности δx найти количество верных значащих цифр в числе x, достаточно подобрать наибольшее натуральное число n так, чтобы имело место неравенство $\delta x \leqslant 10^{-n}$, а потом полученное значение n уменьшить на единицу.

Пример 2.3.1. Пусть $x = 984{,}6$; $\delta x = 0{,}008$. Очевидно, что $0{,}008 \leqslant 10^{-2}$. Это означает, что число x имеет по крайней мере одну верную в строгом смысле цифру (первая слева цифра 9). Полученный результат легко подтвердить, используя определение цифры, верной в строгом смысле. На МК вычислим: $\Delta x = 984{,}6 \cdot 0{,}008 = 7{,}8768$, откуда следует, что в числе 984,6 цифра 9 действительно верна в строгом смысле.

Полученное правило иногда завышает требование к точности — при выполнении условия (2.10) в числе могут оказаться верными все n цифр. Зависит это от величины первых значащих цифр числа x.

Пример 2.3.2. Пусть $x = 136{,}4$; $\delta x = 0{,}008 \leqslant 10^{-2}$ (т. е. $n = 2$). Согласно правилу в числе 136,4 лишь одна верная в строгом смысле цифра ($n - 1 = 1$). Вычислим $\Delta x = 136{,}4 \cdot 0{,}008 < 1{,}1$. Как показывает найденная величина предельной абсолютной погрешности, в числе 136,4 верны в строгом смысле две цифры: 1 и 3.

В отдельных случаях, когда первая значащая цифра в относительной погрешности δx меньше 5, вместо условия (2.10) удается установить более жесткое условие:

$$\delta x \leqslant \frac{1}{2} \cdot 10^{-n}. \tag{2.12}$$

В этом случае число x имеет по крайней мере n верных в строгом смысле цифр. Действительно, с учетом (2.7), (2.9) и (2.12) имеем:

$$\Delta x \leqslant \frac{1}{2} |x| \cdot 10^{-n} \leqslant \frac{1}{2} 10^p \cdot 10^{-n} = \frac{1}{2} \cdot 10^{p-n},$$

т. е.

$$\Delta x \leqslant \frac{1}{2} 10^{-n} \cdot 10^p,$$

а это означает, что мантисса M нормализованного числа x (см. 2.8) имеет по меньшей мере n верных в строгом смысле цифр.

Пример 2.3.3. Пусть $x = 78{,}56$, $\delta x = 0{,}0003$. Имеем: $0{,}0003 < 0{,}0005 = \frac{1}{2} \cdot 10^{-3}$, т. е. в числе x верны в строгом смысле 3 цифры. Действительно, в данном случае $\Delta x = 78{,}56 \cdot 0{,}0003 < 0{,}03$, что подтверждает полученный результат.

Контрольные вопросы

1. В значениях x и Δx десятичная запятая одновременно перемещена на одинаковое число разрядов влево или вправо. Изменится ли при этом количество верных цифр в числе x?

2. Как устанавливается количество верных в строгом смысле цифр по величине относительной погрешности приближенного числа, если первая значащая цифра относительной погрешности меньше 5? больше или равна 5?

Упражнения

1. По заданным значениям приближенных чисел и их относительных погрешностей установить количество цифр, верных в строгом смысле:

а) $x = 2{,}364$; $\delta x = 0{,}07\%$;

б) $y = 109{,}6$; $\delta y = 0{,}04\%$;

в) $z = 24{,}307$; $\delta z = 0{,}005\%$.

Округлить значения x, y и z до верных цифр с сохранением одной запасной.

2. Со сколькими верными в строгом смысле десятичными знаками после запятой нужно взять указанные значения, чтобы относительная погрешность не превышала 0,1%:

а) $\sqrt{19{,}3}$; б) $\sin 0{,}9$; в) $\ln 24{,}6$?

2.4. Формулы для подсчета погрешностей арифметических действий

Одна из основных задач приближенных вычислений, имеющая большое практическое значение, состоит в выяснении того, как влияют на точность конечных результатов вычислений по формулам погрешности исходных данных. Рассмотрим этот вопрос последовательно для четырех основных арифметических действий и элементарных функций, имеющихся на клавиатуре микрокалькулятора.

Сложение и вычитание. Пусть $S=X+Y$ — сумма точных чисел, среди которых могут быть как положительные, так и отрицательные, а $s=x+y$ — сумма их приближений. Составим разность:

$$S-s=(X-x)+(Y-y).$$

Переходя к модулям, получим

$$|S-s|\leqslant|X-x|+|Y-y|,$$

т. е. $e_s\leqslant e_x+e_y$, или тем более $e_s\leqslant\Delta x+\Delta y$. Отсюда следует, что можно принять

$$\Delta s=\Delta x+\Delta y, \tag{2.13}$$

т. е. границей абсолютной погрешности алгебраической суммы можно считать сумму границ абсолютных погрешностей слагаемых.

Пример 2.4.1. Даны приближенные значения $x=235{,}4$ и $y=79{,}1834$, у которых все цифры являются верными в широком смысле. Найдем на МК их сумму:

$$S=235{,}4+79{,}1834=314{,}5834.$$

Для оценки точности результата вычислим сумму погрешностей слагаемых $10^{-1}+10^{-4}=0{,}1001<0{,}11=\Delta S$. Величина ошибки показывает, что в результате (полученной суммы) уже первый знак после запятой является сомнительным. Стоило ли терять время на учет в вычислениях всех знаков после запятой у второго слагаемого?

Из рассмотренного примера следует:

1) получение на МК результата с большим числом значащих цифр еще не означает, что все эти цифры верны;

2) при вычислении сумм и разностей чисел с сильно различающимися абсолютными ошибками с целью экономии времени целесообразно округлить исходные данные, оставив столько десятичных знаков, сколько их имеет слагаемое с наименьшим числом десятичных знаков. Так, в рассмотренном выше примере имело смысл перед выполнением действия сложения округлить y до сотых: 79,18. Руководствуясь этим правилом, следует иметь в виду, что при последовательном вычитании и сложении нескольких чисел выгоднее производить действия над числами в порядке возрастания их абсолютных величин*.

Относительные погрешности суммы и разности можно вычислять через абсолютные, пользуясь формулой (2.5), но можно использовать и специальные формулы.

Получим их:

$$\delta(x+y)=\frac{\Delta x+\Delta y}{x+y}=\frac{|x|\cdot\Delta x}{|x+y|\cdot|x|}+\frac{|y|}{|x+y|}\cdot\frac{\Delta y}{|y|}=$$
$$=\frac{|x|}{|x+y|}\cdot\delta x+\frac{|y|}{|x+y|}\cdot\delta y. \tag{2.14}$$

$$\delta(x-y)=\frac{\Delta x+\Delta y}{|x-y|}=\frac{|x|}{|x-y|}\cdot\frac{\Delta x}{|x|}+\frac{|y|}{|x-y|}\cdot\frac{\Delta y}{|y|}=$$
$$=\frac{|x|}{|x-y|}\cdot\delta x+\frac{|y|}{|x-y|}\cdot\delta y. \tag{2.15}$$

Формулы (2.14) и (2.15) позволяют сделать полезные для практического использования выводы.

Пусть слагаемые x и y — одного знака, а $\delta=\max(\delta x,\ \delta y)$. Тогда из формулы (2.14) следует:

$$\delta(x+y)\leqslant\frac{\delta(|x|+|y|)}{|x+y|}=\delta,\ \text{т. е. }\delta(x+y)\leqslant\delta.$$

Если приближенные слагаемые имеют одинаковый знак, то граница относительной погрешности их суммы не превышает наибольшей из границ относительных погрешностей слагаемых. Как видно из формулы (2.15), при вычитании близких чисел может произойти большая потеря точности. Действительно, когда вычитаемые числа почти одинаковы, то даже при условии, что их собственные ошибки малы, относительная ошибка разности может оказаться большой.

Пример 2.4.2. Найдем разность чисел $x=62{,}425$ и $y=62{,}409$, у которых все цифры верны в строгом смысле. Имеем:

$$x-y=62{,}425-62{,}409=0{,}016.$$

Граница абсолютной погрешности разности

$$\Delta(x-y)=0{,}0005+0{,}0005=0{,}001,$$

поэтому в числе 0,016 из двух значащих цифр верна лишь одна. Сравним границы относительных погрешностей результата и исходных данных:

$$\delta x=\frac{0{,}0005}{62{,}425}=0{,}000008;$$
$$\delta y=\frac{0{,}0005}{62{,}409}=0{,}000008;$$
$$\delta(x-y)=\frac{0{,}001}{0{,}016}=0{,}0625.$$

*Обоснование см., например, в [6].

Таким образом, граница относительной погрешности разности оказалась почти в 8 тысяч раз больше границы относительной погрешности исходных данных. Это означает, что в приближенных вычислениях нужно исключать вычитание близких по величине значений (например, путем преобразования вычисляемых выражений).

Умножение и деление. Пусть $p=x\cdot y$ — произведение двух приближенных чисел, а $q=x/y$ — их частное. Знаки чисел x и y не влияют на величину ошибки, поэтому для простоты примем $x, y>0$. Логарифмируя произведение и частное двух приближенных чисел, получим:

$$\ln p=\ln x+\ln y,\ \ln q=\ln x-\ln y.$$

Принимая во внимание известную в дифференциальном исчислении приближенную формулу

$$df(x)\approx\Delta f(x),$$

получим:

$$\Delta\ln z\approx d\ln z=\frac{\Delta z}{z}.$$

Применяя формулу (2.13), получим:

$$\Delta\ln p=\Delta\ln q=\Delta\ln x+\Delta\ln y,$$

т. е.

$$\frac{\Delta p}{p}=\frac{\Delta q}{q}=\frac{\Delta x}{x}+\frac{\Delta y}{y},\qquad(2.16)$$

откуда следует:

$$\delta(x\cdot y)=\delta(x/y)=\delta x+\delta y,\qquad(2.17)$$

т. е. границей относительной погрешности произведения (частного) можно считать сумму границ относительных погрешностей сомножителей (делимого и делителя).

Из формул (2.16) легко получаются формулы для вычисления границ абсолютных погрешностей произведения и частного:

$$\Delta(xy)=x\cdot\Delta y+y\cdot\Delta x,\qquad(2.18)$$

$$\Delta(x/y)=\frac{x\cdot\Delta y+y\cdot\Delta x}{y^2}.\qquad(2.19)$$

Пример 2.4.3. Числа 43,1 и 5,72 заданы верными цифрами. Найдем на МК их частное: $q=7{,}534965$. Для определения числа верных знаков результата вычислим:

$$\Delta q=\frac{43{,}1\cdot 0{,}01+5{,}72\cdot 0{,}1}{5{,}72^2}=\frac{1{,}003}{32{,}7184}=0{,}0306555.$$

Частное q имеет два верных знака. Округляя с одной запасной цифрой, получим $q=7{,}53$.

Как следует из формул (2.17), относительная погрешность

произведения и частного не может быть меньше, чем относительная погрешность наименее точного из компонентов действий. Величина относительной погрешности приближенного числа определяет количество его верных значащих цифр (п. 2.3). По аналогии с тем как при нахождении алгебраической суммы не имеет смысла сохранять в более точных слагаемых излишнее количество десятичных знаков, при умножении и делении приближенных чисел нет необходимости сохранять в более точных данных излишнее количество значащих цифр.

Для удобства все формулы для вычисления погрешностей арифметических действий сведены в общую таблицу (см. Приложение II). Знак * в таблице обозначает одну из операций: $+, -, \times, /$.

Контрольные вопросы

1. Как объясняется нецелесообразность сохранения излишних десятичных знаков в более точных слагаемых при сложении нескольких чисел?
2. Каким способом можно быстро оценить относительную погрешность суммы нескольких слагаемых одного знака, если известны относительные погрешности каждого слагаемого?
3. По какой причине в вычислениях следует избегать вычитания близких по величине чисел?
4. Как объясняется нецелесообразность сохранения излишних значащих цифр в более точных данных при умножении или делении нескольких чисел?

Упражнения

1. Произвести указанные действия и определить абсолютные и относительные погрешности результатов (исходные числа заданы верными в строгом смысле слова цифрами):

а) $24{,}37-9{,}18$; г) $1{,}504-1{,}502$; ж) $12{,}64\cdot 0{,}3$;
б) $18{,}437+24{,}9$; д) $234{,}5\cdot 194{,}3$; з) $72{,}3:0{,}34$;
в) $24{,}1-0{,}037$; е) $0{,}65\cdot 1984$; и) $8124{,}6:2{,}9$.

2.5. Формулы для подсчета погрешностей значений элементарных функций

Рассмотрим формулы для подсчета погрешностей значений элементарных функций, имеющихся на клавиатуре МК.

Пусть функция $f(x)$ дифференцируема в некоторой окрестности приближенного значения аргумента x, а e_x — абсолютная ошибка значения аргумента. Тогда абсолютная ошибка значения функции $e_f=|f(x+\Delta x)-f(x)|$. Поскольку на практике ошибка e_x обычно мала по сравнению со значением x, воспользуемся

приближенным равенством $e_f \approx |df| = |f'(x)| \cdot e_x$. Заменим e_x на Δx:

$$e_f \leqslant |f'(x)| \cdot \Delta x.$$

Это означает, что можно принять:

$$\Delta f = |f'(x)| \cdot \Delta x. \qquad (2.20)$$

Равенство (2.20) позволяет получить целую серию формул для оценки границ абсолютных погрешностей значений элементарных функций. Пусть, например, $f(x) = \sqrt{x}$. Тогда $\Delta(\sqrt{x}) = |(\sqrt{x})'| \cdot \Delta x = \frac{\Delta x}{2\sqrt{x}}$. Или аналогично:

$$\Delta(\cos x) = |(\cos x)'| \cdot \Delta x = |-\sin x| \cdot \Delta x = |\sin x| \cdot \Delta x$$

и т. д. Для вывода формулы погрешности функции x^y воспользуемся представлением $x^y = e^{y \ln x}$, а затем формулой погрешности экспоненты и произведения:

$$\Delta(x^y) = \Delta(e^{y \ln x}) = e^{y \ln x} \cdot \Delta(y \ln x) = x^y \left(\frac{y \cdot \Delta x}{x} + \ln x \cdot \Delta y \right).$$

Формула (2.20) позволяет сделать важное для практики вычислений наблюдение. Если значение модуля производной функции $f(x)$ в точке x меньше единицы, то $\Delta f < \Delta x$, т. е. абсолютная ошибка значения функции оказывается меньше абсолютной ошибки значения аргумента. Если же $|f'(x)| > 1$, то значение функции будет иметь ошибку, бóльшую ошибки аргумента. Этот факт имеет простое геометрическое толкование. При $|f'(x)| < 1$ функция меняется медленно, т. е. предел изменения значений функции меньше предела изменений аргумента (рис. 12, *а*). При $|f'(x)| > 1$ малым отклонениям аргумента будут соответствовать большие колебания значений функции (рис. 12, *б*). Второй случай (при достаточно больших значениях $|f'(x)|$) приводит к резкой потере точности и потому должен особо учитываться в практике вычислений (пример 2.5.2 на с. 66).

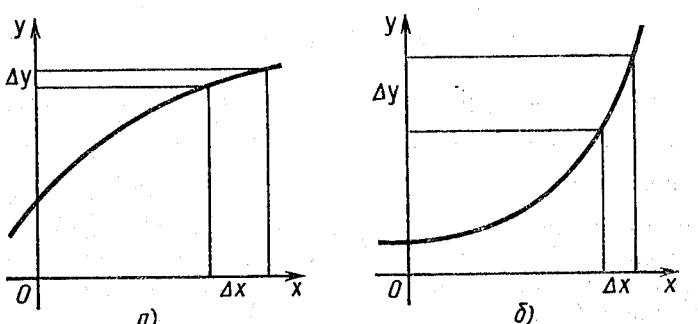


Рис. 12

Таблица 2.1

$f(x)$	$\Delta f(x)$	Программа
$\sqrt{x}$	$\frac{\Delta x}{2\sqrt{x}}$	Δx [÷] 2 [÷] x [√] [=]
$\frac{1}{x}$	$\frac{\Delta x}{x^2}$	Δx [÷] x [y^x] 2 [=]
$\sin x$	$\lvert\cos x\rvert \cdot \Delta x$	x [cos] [×] Δx [=] {[/−/]}
$\cos x$	$\lvert\sin x\rvert \cdot \Delta x$	x [sin] [×] Δx [=] {[/−/]}
$\operatorname{tg} x$	$\frac{\Delta x}{\cos^2 x}$	Δx [÷] x [cos] [y^x] 2 [=]
$\ln x$	$\frac{\Delta x}{x}$	Δx [÷] x [=]
$\lg x$	$\frac{\Delta x}{x \ln 10}$	Δx [÷] x [÷] 10 [ln] [=]
e^x	$e^x \cdot \Delta x$	x [e^x] [×] x [=]
10^x	$10^x \cdot \ln 10 \cdot \Delta x$	10 [y^x] x [×] 10 [ln] [×] Δx [=]
$\arcsin x$ $\arccos x$	$\frac{\Delta x}{\sqrt{1-x^2}}$	1 [−] x [y^x] 2 [=] [√] [1/x] [×] Δx [=]
$\operatorname{arctg} x$	$\frac{\Delta x}{1+x^2}$	1 [+] x [y^x] 2 [=] [1/x] [×] Δx [=]
x^y	$x^y\left(\lvert y\rvert \frac{\Delta x}{x} + \lvert\ln x\rvert \cdot \Delta y\right)$	x [y^x] y [×] [(] $\lvert y\rvert$ [×] Δx [÷] x [+] x [ln] {[/−/]} [×] Δy [)]

Формулы для вычисления границ абсолютных погрешностей значений некоторых функций одной переменной приведены в таблице 2.1. Там же показаны программы для вычисления значений этих погрешностей на микрокалькуляторе «Электроника МК-41» (клавиши изменения знака [/—/], заключенные в фигурные скобки, нажимаются лишь в случае необходимости — когда полученные на индикаторе предыдущие значения имеют знак «минус»). Во всех случаях предполагается, что значения x принадлежат области допустимых значений аргумента. Получаемые на МК значения границ погрешностей округляются для последующего использования до одной-двух значащих цифр (по избытку!).

Пример 2.5.1. После набора на МК-41 клавиш [arc] [,] [8] получим $\sin 0{,}8 = 0{,}717356091$. Если 0,8 — точное значение, то в соответствии с погрешностью МК найденный результат имеет точность $\pm 10^{-9}$. Если же 0,8 — приближенное значение, у которого цифра 8 верна, например, в строгом смысле, то граница абсолютной погрешности значения аргумента $\Delta x = 0{,}05$, а погрешность полученного значения синуса в соответствии с формулой оценки границы абсолютной погрешности будет:

$$\Delta(\sin x) = \cos x \cdot \Delta x \leqslant 0{,}7 \cdot 0{,}05 = 0{,}035.$$

Отсюда следует, что во втором случае полученное на МК значение $\sin 0{,}8 = 0{,}717356091$ имеет лишь одну верную значащую цифру. Округляя результат с одной запасной цифрой, получим 0,72.

В тех случаях, когда производная функции вблизи приближенного значения аргумента имеет большие по модулю значения, произойдет большая потеря точности.

Пример 2.5.2. Пусть $x = 1{,}5$, причем $\Delta x = 0{,}05$, т. е. все цифры в числе x верны в строгом смысле. Нужно вычислить $\operatorname{tg} x$. С помощью МК-41 получаем $\operatorname{tg} 1{,}5 = 14{,}10141994$. Для определения верных цифр в результате оценим его абсолютную погрешность:

$$\Delta(\operatorname{tg} x) = \frac{\Delta x}{\cos^2 x} \approx \frac{0{,}05}{0{,}005} = 10,$$

в полученном значении $\operatorname{tg} 1{,}5$ ни одну цифру нельзя считать верной.

Прокомментируем полученный результат. Заданная точность исходного значения аргумента определяет пределы его возможных значений: $1{,}45 \leqslant x \leqslant 1{,}55$. Найдем на МК значения $\operatorname{tg} x$ для граничных значений x:

$$\operatorname{tg} 1{,}45 = 8{,}238092753;$$
$$\operatorname{tg} 1{,}55 = 48{,}07848247.$$

Как видно, диапазон изменения $\operatorname{tg} x$ составляет около 40 единиц, что подтверждает отсутствие смысла в полученном выше

результате вычислений. В подобных случаях надо использовать все имеющиеся возможности, чтобы увеличить число верных знаков в исходном данном (например, использовать более точный измерительный прибор, если исходное данное получается в результате измерений).

Пусть в условиях рассмотренного выше примера 2.5.2 $x = 1{,}4923$, $\Delta x = 0{,}0005$. Тогда: $\operatorname{tg} x = \operatorname{tg} 1{,}4923 = 12{,}71327341$, $\Delta(\operatorname{tg} x) < \frac{0{,}0005}{0{,}006} < 0{,}084$, т. е. в полученном результате в строгом смысле верны 2 цифры. Округляя его с одной запасной цифрой, получим 12,7.

Из формулы (2.20) легко получается оценка границы относительной погрешности функции через границу относительной погрешности значения аргумента:

$$\delta f = \frac{\Delta f}{|f(x)|} = \frac{|f'(x)| \cdot \Delta x}{|f(x)|} = \frac{|f'(x)|}{|f(x)|} \cdot |x| \cdot \frac{\Delta x}{|x|} = \frac{|f'(x)|}{|f(x)|} \cdot |x| \cdot \delta x,$$

т. е.

$$\delta f = \frac{|f'(x)|}{|f(x)|} \cdot |x| \cdot \delta x \qquad (2.21)$$

или

$$\delta f = \frac{|f'(x)|}{|f(x)|} \cdot \Delta x. \qquad (2.22)$$

Так, пользуясь формулами (2.21) и (2.22), можно, например, получить:

$$\delta(\sqrt{x}) = \frac{|x|}{2\sqrt{x} \cdot \sqrt{x}} \cdot \delta x = \frac{1}{2}\delta x = \frac{\Delta x}{2 \cdot |x|};$$

$$\delta(\sin x) = \frac{|\cos x|}{|\sin x|} \cdot |x| \cdot \delta x = |x \cdot \operatorname{ctg} x| \cdot \delta x = |\operatorname{ctg} x| \cdot \Delta x;$$

$$\delta(e^x) = \frac{e^x}{e^x} \cdot |x| \cdot \delta x = |x| \cdot \delta x = \Delta x;$$

$$\delta(\ln x) = \frac{x \cdot \delta x}{x\,|\ln x|} = \frac{1}{|\ln x|} \cdot \delta x;$$

$$\delta(x^y) = \delta(e^{y \ln x}) = |y \cdot \ln x| \cdot \delta(y \ln x) =$$

$$= |y \ln x| \cdot [\delta y + \delta(\ln x)] =$$

$$= |\ln x| \cdot \Delta y + y \cdot \Delta(\ln x) = |y \ln x| \cdot \delta y + |y| \cdot \delta x \text{ и т. д.}$$

В зависимости от способа задания исходных данных (с абсолютными или относительными погрешностями) вычислитель может использовать для подсчета границ относительных погрешностей тот или иной вариант формулы. Некоторые из этих формул вместе с соответствующими вычислительными программами для МК-41 приведены в таблице 2.2.

Пример 2.5.3. Значение аргумента $x = 26{,}3$ имеет относительную ошибку около 0,2%. Оценить величину относительной ошибки $\ln 26{,}3$.

Таблица 2.2

$f(x)$	$\delta f(x)$	Программа
$\sqrt{x}$	$\frac{1}{2}\cdot\delta x$	δx [÷] 2 [=]
$\frac{1}{x}$	$\frac{\Delta x}{\lvert x\rvert}$	Δx [÷] x [=] {[/−/]}
$\sin x$	$\lvert\operatorname{ctg} x\rvert\cdot\Delta x$	x [tg] [1/x] [×] Δx [=] {[/−/]}
$\cos x$	$\lvert\operatorname{tg} x\rvert\cdot\Delta x$	x [tg] [×] Δx [=] {[/−/]}
$\operatorname{tg} x$	$\frac{2}{\lvert\sin 2x\rvert}\cdot\Delta x$	2 [×] Δx [÷] (2 [×] x) [sin] [=] {[/−/]}
$\ln x$	$\frac{\delta x}{\lvert\ln x\rvert}$	δx [÷] x [ln] [=] {[/−/]}
$\lg x$	$\frac{\delta x}{\lvert\lg x\rvert\cdot\ln 10}$	δx [÷] x [lg] {[/−/]} [÷] 10 [ln] [=]
e^x	$\lvert x\rvert\cdot\delta x$	$\lvert x\rvert$ [×] δx [=]
10^x	$\ln 10\cdot\Delta x$	10 [ln] [×] Δx [=]
$\arcsin x$	$\frac{\Delta x}{\lvert\arcsin x\rvert\cdot\sqrt{1-x^2}}$	Δx [÷] x [arc] [sin] [÷] (1 [−] x [×] x) [y^x] 2 [1/x] [=] {[/−/]}
$\arccos x$	$\frac{\Delta x}{\lvert\arccos x\rvert\cdot\sqrt{1-x^2}}$	Δx [÷] x [arc] [cos] [÷] (1 [−] x [×] x) [y^x] 2 [1/x] [=] {[/−/]}
$\operatorname{arctg} x$	$\frac{\Delta x}{\lvert\operatorname{arctg} x\rvert\cdot(1+x^2)}$	Δx [÷] x [arc] [tg] [÷] (1 [+] x [×] x) [=] {[/−/]}
x^y	$\lvert y\ln x\rvert\cdot\delta y+\lvert y\rvert\cdot\delta x$	y [×] x [ln] {[/−/]} [×] δy [+] $\lvert y\rvert$ [×] δx [=]

Используя соответствующую формулу и программу из таблицы 2.2 с помощью МК, получим $\delta(\ln 26{,}3)=\frac{\delta(26{,}3)}{26{,}3}=\frac{0{,}002}{26{,}3}=$ $=0{,}0006117\approx 0{,}06\%$. Используя величину найденной относительной погрешности, можно оценить количество верных в строгом смысле значащих цифр в искомом значении $\ln 26{,}3$. Поскольку имеет место $0{,}0006117<10^{-5}$ (формула 2.10, с. 58), то можно сделать вывод, что в числе $\ln 26{,}3=3{,}269568$ по крайней мере четыре первых слева цифры верны в строгом смысле.

Контрольные вопросы

1. В какой зависимости находится абсолютная погрешность значения функции одной переменной от абсолютной погрешности значения аргумента?
2. Как объясняется резкая потеря точности значения $f(x)$ для приближенных значений аргумента x на участках с большими по модулю значениями $f'(x)$?

Упражнения

1. Исходные числовые значения аргумента заданы цифрами, верными в строгом смысле. Определить, пользуясь МК, количество верных в строгом смысле цифр в следующих значениях элементарных функций:

а) $\cos 0{,}47$; г) $\arcsin 0{,}82$;

б) $\frac{1}{0{,}024}$; д) $e^{-3{,}1}$;

в) $\ln 18{,}4$; е) $2{,}6^{1{,}21}$.

2. Значение $x=2{,}701$ имеет относительную ошибку $0{,}01\%$. Оценить количество верных в строгом смысле значащих цифр в значениях:

а) $\sqrt{x}$; б) $\sin x$; в) x^x.

2.6. Практика вычислений

Рассмотрим практические приемы вычислений, которые используют изложенные выше понятия и формулы. В зависимости от принятой методики пользователь может вести операционный учет движения ошибок или сделать оценку точности после проведения нескольких вычислительных действий.

До появления микрокалькуляторов (МК) пооперационный подход к оценке точности вычислений рассматривался еще и как способ рационализации вычислений, позволяющий на каждом шагу избавляться от сомнительных цифр.

Однако то, ради чего преимущественно использовался пооперационный подход (исключение сомнительных цифр из последу-

ющих вычислений), при появлении МК с дополнительным регистром памяти утратило свой изначальный смысл. При умелом использовании дополнительных регистров памяти и других возможностей, предоставляемых современными калькуляторами, нет необходимости выписывать и заново вводить промежуточные результаты даже при вычислениях по достаточно громоздким формулам. И все же неизбежны ситуации (или потому, что так поставлена вычислительная задача, или это связано с желанием самого вычислителя), когда пооперационный учет движения ошибок необходим. Рассматривая в дальнейшем приемы вычислений, мы будем учитывать как пооперационную, так и итоговую методику оценки точности.

Вычисления по правилам подсчета цифр. При вычислении этим методом явного учета погрешностей не ведется, правила подсчета цифр показывают лишь, какое количество значащих цифр или десятичных знаков в результате можно считать надежными. Сами эти правила основываются на формулах для оценки погрешностей арифметических действий и функций (п. 2.4 и 2.5). Приведем эти правила здесь в систематизированном виде.

1. При сложении и вычитании приближенных чисел младший из сохраняемых десятичных разрядов результата должен являться наибольшим среди десятичных разрядов, выражаемых последними верными значащими цифрами исходных данных*. При этом следует избегать вычитания близких по величине чисел, а также при пооперационном применении правила для сложения и вычитания нескольких чисел подряд стараться производить действия над числами в порядке возрастания их абсолютных величин.

2. При умножении и делении приближенных чисел нужно выбрать число с наименьшим количеством значащих цифр и округлить остальные числа так, чтобы в них было лишь на одну значащую цифру больше, чем в наименее точном числе. В результате следует считать верными столько значащих цифр, сколько их в числе с наименьшим количеством значащих цифр.

3. При определении количества верных цифр в значении элементарных функций от приближенных значений аргумента следует прикинуть значение модуля производной функции. Если это значение не превосходит единицы или близко к ней, то в значении функции можно считать верными столько знаков

*Обычно точность исходных данных такова, что все они имеют десятичные знаки после запятой, т. е. являются десятичными дробями. В этом случае правило формулируется более доступно: при сложении и вычитании приближенных чисел в результате следует считать верными столько десятичных знаков после запятой, сколько их в приближенном данном с наименьшим числом знаков после запятой. Количество десятичных знаков после запятой перед выполнением действия целесообразно уравнивать, округляя до одной запасной исходные данные с бо́льшим количеством десятичных знаков.

после запятой, сколько их имеет значение аргумента. Если же модуль производной функции в окрестности приближенного значения аргумента превосходит единицу, то количество верных десятичных знаков в значении функции меньше, чем в значении аргумента на величину k, где k — наименьший натуральный показатель степени, при котором имеет место неравенство:

$$|f'(x)| < 10^k*.$$

4. При записи промежуточных результатов следует сохранять на одну цифру больше, чем рекомендуют правила 1—3. В окончательном результате эта запасная цифра округляется.

Правила подсчета цифр носят оценочный характер и не являются методом строгого учета точности вычислений. Обычно их применяют тогда, когда быстро и без особых затрат нужно получить результат, не особенно беспокоясь о его достоверности. Между тем практическая надежность этих правил достаточно высока из-за значительной вероятности взаимопогашения ошибок, не учитываемой при строгом подсчете границ погрешностей.

При пооперационной регистрации результатов вычислений используется обычная расчетная таблица — так называемая *расписка формулы*.

Пример 2.6.1. Вычислить значение величины

$$A = \frac{e^a + \sqrt{b}}{\ln(a + b^2)} \qquad (2.23)$$

по правилам подсчета цифр для приближенных значений $a = 2{,}156$ и $b = 0{,}927$, у которых все цифры верные.

Вычисления приведены в таблице 2.3. Прокомментируем ход вычислений. На МК получаем $e^{2{,}156} = 8{,}63652$. Это же дает нам и оценку величины производной в этой же точке: $2^{2{,}156} < 10^1$, т. е. в полученном значении следует сохранить на один десятичный знак меньше, чем в значении аргумента. Округляя с одной запасной, получаем 8,637 (запасная цифра выделена) и заносим результаты в таблицу. Далее: $\sqrt{0{,}927} = 0{,}9628083$, причем модуль производной $(1/(2\sqrt{a}))$ меньше единицы, поэтому сохраняем после запятой три знака и один запасной: 0,9628. При вычислении суммы в числителе находим $8{,}637 + 0{,}9628 = 9{,}5998$ и согласно этому правилу 1 округляем результат до тысячных: 9,600. При вычислении b^2 пользуемся правилом 2, при нахождении суммы $a + b^2$ — правилом 1.

При определении количества верных цифр в значении $\ln 3{,}0153$ снова применяем правило 3 (учитываем, что производная $\ln x$ при $x > 1$ имеет значение, меньшее единицы). Округляя окончательный результат без запасной цифры, получим $A = 8{,}70$ (три верные значащие цифры).

*Действительно, при этом условии с учетом формулы $\Delta f = |f'(x)| \cdot \Delta x$ можно вывести, что увеличение k на единицу означает увеличение Δf примерно в 10 раз, что в данном случае уменьшает в значении $f(x)$ количество верных десятичных знаков по сравнению со значением x на единицу.

Таблица 2.3

a	b	e^a	$\sqrt{b}$	$e^a+\sqrt{b}$	b^2	$a+b^2$	$\ln(a+b^2)$	A
2,156	0,927	8,637	0,9628	9,600	0,8593	3,0153	1,1037	8,698

Легко видеть, что с помощью инженерного микрокалькулятора вычисления по формуле (2.23) можно произвести и без выписывания промежуточных результатов. Программа для МК Б3-18 может иметь вид:

a [F] [e^x] [F] [ЗАП] b [F] [$\sqrt{}$] [F] [П+] b [×] [+] a [=]

[F] [ln] [÷] [F] [ИП] [↔] [=].

Программа для МК-41:

[(] a [e^x] [+] b [$\sqrt{b}$] [)] [÷] [(] a [+] b [×] b [)] [ln] [=].

Выполнив последнюю программу при заданных выше значениях a и b, используя клавиши [arc] [,] [9], получим на индикаторе МК-41 число 8,697338929. Как выделить в полученном числе верные цифры? Сделать это можно и без подробного поэтапного анализа, который приведен в примере 2.6.1. Действительно, так как выражение A представляет собою дробь, то последнее действие при его вычислении — деление, а следовательно, результат будет содержать верных значащих цифр не более чем в наименее точном из операндов — числителе или знаменателе. Так как корень квадратный дает верных цифр столько же, сколько и его аргумент (три), а экспонента теряет не более одного верного знака после запятой (что вместе с ненулевой целой частью также дает не менее трех значащих цифр), то в числителе и знаменателе число верных значащих цифр будет равно трем. Следовательно, значение A должно быть округлено до трех верных знаков: $A=8{,}70$.

Там, где возможен подобный анализ, при использовании МК в вычислениях по правилам подсчета цифр можно избежать пооперационного учета верных знаков.

Контрольные вопросы

1. Как формулируются правила подсчета цифр?
2. В каких случаях рекомендуется применять правила подсчета цифр?
3. Какие два способа применения правил подсчета цифр возможны в вычислениях на микрокалькуляторе?

4. Какова последовательность действий на каждом промежуточном этапе расчетной таблицы в вычислениях по правилам подсчета цифр с пооперационным учетом ошибок? на заключительном этапе?

Упражнения

Вычислить на МК значения заданных выражений по правилам подсчета цифр двумя способами: 1) с пооперационным анализом результатов; 2) с итоговой оценкой окончательного результата (у числовых данных все цифры — верные):

а) $\dfrac{\sin 0{,}93+27{,}9}{\sqrt{12{,}34}}$;

б) $\dfrac{\ln(0{,}84+4{,}37^2)}{\sqrt[4]{624{,}9}}$;

в) $\dfrac{\ln(5{,}6+\sqrt{2{,}3^2+4{,}9^2})}{\sin^2 0{,}37+\cos^2 1{,}02}$;

г) $\dfrac{\sqrt[3]{8{,}044}}{1+3{,}04^2}+0{,}82^{1{,}37}$.

Метод строгого учета границ погрешностей. Этот метод предусматривает строгий подсчет границ погрешностей по правилам вычисления погрешностей, рассмотренных в п.п. 2.4 и 2.5.

При пооперационном строгом учете ошибок промежуточные результаты, так же как и их погрешности, заносят в специальную расчетную таблицу, состоящую из двух параллельно заполняемых частей — для результатов и их погрешностей. В таблице 2.4 приведены вычисления со строгим учетом границ абсолютных погрешностей по той же формуле, что и в примере 2.6.1 (с. 71), и в предположении, что исходные данные a и b имеют границы абсолютных погрешностей $\Delta a=\Delta b=0{,}005$ (т. е. у a и b все цифры верны в строгом смысле). Промежуточные результаты

Таблица 2.4

a	b	e^a	$\sqrt{b}$	$e^a+\sqrt{b}$	b^2	$a+b^2$	$\ln(a+b^2)$	A
2,156	0,927	8,637	0,9628	9,60	0,860	3,016	1,104	8,70
Δa	Δb	$\Delta(e^a)$	$\Delta(\sqrt{b})$	$\Delta(e^a+\sqrt{b})$	$\Delta(b^2)$	$\Delta(a+b^2)$	$\Delta\ln(a+b^2)$	ΔA
0,0005	0,0005	0,0049	0,00027	0,0054	0,0016	0,0021	0,00076	0,016

вносят в таблицу после округления до одной запасной (с учетом вычисленной параллельно величины погрешности); значения погрешностей для удобства округляются по избытку до двух значащих цифр. Проследим ход вычислений на одном этапе.

С помощью МК имеем $e^{2{,}156}=8{,}63652$. Подсчитаем границу абсолютной погрешности (табл. 2.1):

$$\Delta(e^{2{,}156})=e^{2{,}156}\cdot 0{,}0005=0{,}0043182\approx 0{,}0044.$$

Судя по величине погрешности, в полученном значении экспоненты в строгом смысле верны два знака после запятой. Округляем это значение с одной запасной цифрой $e^{2,156} \approx 8,637$ (запасная цифра выделена) и вносим его в таблицу. Вслед за этим вычисляем полную погрешность полученного результата (погрешность действия плюс погрешность округления: $0,0044 + 0,00048 \approx 0,0049$), которая также вносится в таблицу. Все последующие действия выполняются аналогично с применением соответствующих формул для границ абсолютных погрешностей. Округляя окончательный результат до последней верной цифры, а также округляя погрешность до соответствующих разрядов результата, окончательно получаем:

$$A = 8,7 \pm 0,1.$$

Рассмотрим теперь, как можно получить итоговую оценку границы погрешности результата вычислений по формуле без пооперационного учета движения ошибок.

Пример 2.6.2. Значения $x = 14,7$ и $y = 3,21$ даны цифрами, верными в строгом смысле. Вычислить значение выражения:

$$B = \frac{\sqrt{x}}{y \cdot \ln x}.$$

С помощью МК-41 по непрерывной программе:

14,7 [√] [÷] 3,21 [÷] 14,7 [ln] [=]

получаем $B = 0,444374379$. Используя формулы относительных погрешностей частного и произведения, запишем:

$$\delta B = \delta(\sqrt{x}) + \delta y + \delta(\ln x), \text{ т. е. } \delta B = \frac{1}{2}\delta x + \delta y + \frac{\delta x}{|\ln x|}.$$

Пользуясь МК, получим $\delta B = 0,062$, что дает:

$$\Delta B = B \cdot \delta B = 0,028.$$

Это означает, что в результате лишь одна цифра после запятой верна в строгом смысле:

$$B = 0,44 \pm 0,03.$$

Контрольные вопросы

1. Как оформляются вычисления со строгим учетом границ погрешностей при пооперационном учете ошибок?
2. Какова последовательность действий на каждом промежуточном этапе расчетной таблицы в вычислениях по методу строгого учета границ погрешностей с пооперационным учетом ошибок? на заключительном этапе?
3. Как вычисляются границы погрешностей результата при использовании методики итоговой оценки ошибки вычислений?

Упражнения

У значений $a = 1,3794$ и $b = 29,37$ все цифры верны в строгом смысле. Вычислить с помощью МК значения заданных выражений со строгим учетом границ погрешностей двумя способами: 1) с пооперационным учетом погрешностей; 2) с итоговой оценкой точности результата:

а) $\frac{\sqrt{\operatorname{tg} a}}{\ln(a+b^2)}$; в) $\frac{\sqrt[3]{b}}{1+a^2} + b^{\sin a}$;

б) $\frac{\cos a + b}{\sqrt[4]{a^2+b^2}}$; г) $\frac{\ln(\sin a + b)}{a^{\sqrt{b}} + b^{\sqrt{a}}}$.

Метод границ. Если нужно иметь гарантированные границы возможных значений вычисляемой величины, используют специальный метод вычислений — метод границ.

Пусть $f(x, y)$ — функция, непрерывная и монотонная в некоторой области допустимых значений аргументов x и y. Нужно вычислить ее значение $f(a, b)$, где a и b — приближенные значения аргументов, причем известно, что

$$НГ_a < a < ВГ_a,\ НГ_b < b < ВГ_b.$$

Здесь НГ, ВГ — обозначения соответственно нижней и верхней границ значений аргументов. Итак, необходимо найти строгие границы значений $f(a, b)$ при известных границах значений a и b.

Допустим, что функция $f(x, y)$ возрастает по каждому из аргументов x и y. Тогда:

$$f(НГ_a, НГ_b) < f(a, b) < f(ВГ_a, ВГ_b).$$

Пусть теперь $f(x, y)$ возрастает по аргументу x и убывает по аргументу y. Тогда будет выполняться неравенство:

$$f(НГ_a, ВГ_b) < f(a, b) < f(ВГ_a, НГ_b).$$

Применим метод границ к основным арифметическим действиям. Пусть, например, $f(x, y) = x + y$. Тогда очевидно, что

$$НГ_a + НГ_b < a + b < ВГ_a + ВГ_b.$$

Точно так же для функции $f(x, y) = x - y$, возрастающей по x и убывающей по y, имеем:

$$НГ_a - ВГ_b < a - b < ВГ_a - НГ_b.$$

Аналогично для умножения и деления:

$$НГ_a \cdot НГ_b < ab < ВГ_a \cdot ВГ_b,$$
$$НГ_a / ВГ_b < a/b < ВГ_a / НГ_b.$$

Рассмотрим функцию $\frac{1}{\ln(x-y)}$. Замечаем, что при увеличении x она убывает, а с увеличением y возрастает (разумеется,

при соблюдении условий существования). Следовательно, имеет место:

$$\frac{1}{\ln(\text{ВГ}_a-\text{НГ}_b)}<\frac{1}{\ln(a-b)}<\frac{1}{\ln(\text{НГ}_a-\text{ВГ}_b)}.$$

Вычисляя по методу границ с пооперационной регистрацией промежуточных результатов, удобно использовать обычную вычислительную таблицу, состоящую из двух строк — для вычисления НГ и ВГ результата (по этой причине метод границ называют еще *методом двойных вычислений*). При выполнении промежуточных вычислений и округлении результатов используются все рекомендации правил подсчета цифр с одним важным дополнением: округление нижних границ ведется по недостатку, а верхних — по избытку. Окончательные результаты округляются по этому же правилу до последней верной цифры.

В таблице 2.5 приведены вычисления по формуле

$$A=\frac{e^a+\sqrt{b}}{\ln(a+b^2)}$$

методом границ. Нижняя и верхняя границы значений a и b определены из условия, что в исходных данных $a=2{,}156$ и $b=0{,}927$ все цифры верны в строгом смысле ($\triangle a=\triangle b=0{,}0005$), т. е. $2{,}1555<a<2{,}1565$; $0{,}9265<b<0{,}9275$.

Таблица 2.5

	a	b	e^a	$\sqrt{b}$	$e^a+\sqrt{b}$	b^2	$a+b^2$	$\ln(a+b^2)$	A
НГ	2,1555	0,9265	8,63220	0,96255	9,59475	0,85840	3,01434	1,10338	8,6894
ВГ	2,1565	0,9275	8,64084	0,96307	9,60391	0,86026	3,01676	1,10419	8,7041

Способ границ связан со способом строгого учета границ погрешностей следующим образом. Пусть X — точное значение некоторой величины; x — его приближение с известными границами НГ_x и ВГ_x. Примем x равным значению $\frac{\text{НГ}_x+\text{ВГ}_x}{2}$, тогда абсолютная погрешность e_x этого приближения будет заведомо не больше полуразности $\Delta x=\frac{\text{ВГ}_x-\text{НГ}_x}{2}$. Так, по результатам вычислений в таблице 2.5 получаем:

$$A=\frac{8{,}6894+8{,}7041}{2}=8{,}69675,\quad \Delta A=\frac{8{,}7041-8{,}6894}{2}=0{,}00735,$$

что дает $A=8{,}69675\pm0{,}00735$.

Вычисления по методу границ можно вести и без пооперационного фиксирования промежуточных результатов.

Пример 2.6.3. Нужно найти границы значения выражения:

$$Z=\frac{\sqrt{x}}{\ln(x-y^2)},$$

если $4{,}845<x<4{,}855$; $1{,}215<y<1{,}225$. Имеем:

$$\frac{\sqrt{\text{НГ}_x}}{\ln(\text{ВГ}_x-(\text{НГ}_y)^2)}<Z<\frac{\sqrt{\text{ВГ}_x}}{\ln(\text{НГ}_x-(\text{ВГ}_y)^2)}.$$

Используя дважды непрерывную программу вычисления Z на МК-41:

x [$\sqrt{}$] [÷] [(] x [−] y [×] y [)] [ln] [=],

получим:

$$1{,}807895009<Z<1{,}825100030.$$

Если нет необходимости иметь результат с таким большим количеством значащих цифр, его можно округлить (нижнюю границу — по убыванию, верхнюю — по возрастанию). Так, округляя границы Z до сотых, будем иметь:

$$1{,}80<Z<1{,}83, \text{ т. е.}$$
$$Z=1{,}815\pm0{,}015.$$

Контрольные вопросы

1. В каких случаях в приближенных вычислениях применяют метод границ?
2. Как оформляются расчеты по методу границ при пооперационном учете результатов вычислений?
3. Какова последовательность действий на каждом промежуточном этапе расчетной таблицы в вычислениях по методу границ с пооперационным учетом результатов вычислений? на заключительном этапе?
4. Как обнаруживается связь метода границ с методом строгого учета границ погрешностей?

Упражнения

Значения a и b заключены в границах $12{,}3<a<12{,}5$; $64{,}2<b<64{,}4$. Вычислить с помощью МК значения заданных выражений по методу границ двумя способами: 1) с пооперационной регистрацией промежуточных результатов; 2) без выписывания промежуточных результатов:

а) $\frac{\sqrt{a^2-b}}{\ln a}$; в) $\frac{\ln(a^2+b)}{a^2+b^2}$;

б) $\frac{a+b}{\sqrt[4]{a^2+b^2}}$; г) $\frac{\sqrt[3]{b}}{1+a^2}+a^{\ln b}$.

Глава 3
АЛГОРИТМЫ

3.1. Принцип алгоритмизации в использовании вычислительной техники

Используя микрокалькулятор для вычислений по формуле, иногда полезно заранее составить и записать на бумаге *программу*, т. е. перечень действий, которые должны быть проделаны пользователем в процессе счета. Такую программу особенно целесообразно иметь тогда, когда предстоит произвести несколько вычислений по одной и той же формуле. В этом случае сначала составляется и тщательно выверяется программа вычислений, а потом пользователь может использовать ее механически, уже не вникая в последовательность производимых действий, что облегчает и ускоряет процесс счета.

Более совершенные вычислительные машины способны запоминать программы вычислений, и тогда человеку остается только составить программу и поместить ее в память машины, а все остальное — собственно решение задачи — машина сделает автоматически, т. е. без участия человека. Такие вычислительные машины называют *программно-управляемыми*. К их числу относятся электронно-вычислительные машины (ЭВМ), появившиеся в середине нашего столетия. Это — высокопроизводительные средства обработки информации, способные решать большой круг самых разнообразных задач. Для составления программ на ЭВМ разработаны и используются специальные языки, к числу которых относятся широко распространенные языки программирования Алгол, Фортран, Кобол, ПЛ/1, Паскаль, Бейсик и др. Есть программно-управляемые модели и среди микрокалькуляторов, к ним относятся отечественные МК Б3-21, Б3-34, Б3-54, МК-56. Программируемый микрокалькулятор, несмотря на его ограниченные возможности по сравнению с универсальной крупной ЭВМ, обладает всеми основными свойствами программно-управляемой вычислительной машины.

Общим для всей программно-управляемой вычислительной техники является то, что решение задач на ней осуществляется посредством составления программы. Основой программы для вычислительной машины является *алгоритм* решения данной задачи, т. е. точное предписание о последовательности действий, которые должны быть произведены для получения результата. Алгоритм является более общим понятием, чем программа; в этом смысле программа для вычислительной машины — это запись алгоритма решения некоторой задачи в виде, пригодном для данной вычислительной машины. Отсюда следует, что основная

сущность процесса решения задач с помощью программно-управляемой техники — это разработка алгоритмов. Когда алгоритм решения задачи ясен, он без особого труда может быть представлен на любом языке программирования с учетом особенностей конкретных вычислительных машин. Говоря иными словами, главное в решении задач на программно-управляемых вычислительных машинах — это *алгоритмизация*, т. е. составление алгоритмических предписаний. Роль алгоритмизации в жизни современного общества определяется не только техническими аспектами ее использования. Алгоритмический подход неотделим от повседневной жизни людей, от их обычной работы. В подавляющем большинстве случаев результат деятельности человека прямо зависит от того, насколько четко он чувствует алгоритмическую сущность своих действий: что делать в каждый момент, в какой последовательности, каким должен быть итог действий и т. п. Все это определяет особый аспект культуры мышления и поведения, характеризующийся умением составлять и использовать различные алгоритмы.

Контрольные вопросы

1. Что такое программно-управляемая вычислительная машина?
2. В чем состоит значение алгоритмизации для процесса решения задачи на программно-управляемой вычислительной машине?

3.2. Алгоритмы и способы их представления

Как уже отмечалось выше, понятие алгоритма относится к числу фундаментальных математических понятий. Вместе с тем для ознакомления с методами алгоритмизации в связи с записью программы для вычислительных машин нет необходимости обращаться к строгому определению этого понятия.

Итак, алгоритм — это точное и полное предписание о последовательности выполнения конечного числа действий, необходимых для решения любой задачи данного типа. Слово «алгоритм» возникло от algorithmi — латинской формы написания имени великого математика аль-Хорезми, который в IX в. описал правила выполнения четырех основных арифметических действий над числами в десятичной системе счисления. Правила эти и сегодня служат простейшими примерами математических алгоритмов. Укажем основные свойства алгоритмических предписаний.

1. Описываемый алгоритмом вычислительный процесс должен быть разбит на последовательность отдельных действий. Возникающее при этом описание представляет собой последовательность четко разделенных друг от друга указаний, образующих прерывную (или, как говорят, дискретную) структуру

алгоритмического процесса — только выполнив требования одного указания, можно перейти к следующему.

Это свойство алгоритмических предписаний называют свойством *дискретности*.

2. Алгоритм не должен содержать указаний, смысл которых может восприниматься неоднозначно. Кроме того, после выполнения очередного указания алгоритма не должно возникать никаких разночтений относительно того, какое указание будет выполняться следующим. Говоря иными словами, при исполнении алгоритма никогда не должна возникать потребность в принятии каких-либо решений, не предусмотренных составителем алгоритма.

Это свойство алгоритмических описаний называется свойством *определенности* или *детерминированности*.

3. Алгоритм составляется не для решения одной конкретной задачи, а для целого класса задач данного типа. В простейшем случае эта вариативность алгоритма обеспечивается возможностью использовать различные допустимые исходные данные. Так, например, программа вычисления по формуле $y=(\ln\sqrt{x})^2$ на микрокалькуляторе типа Б3-18

$$x\ \boxed{\mathrm{F}}\ \boxed{\sqrt{}}\ \boxed{\mathrm{F}}\ \boxed{\ln}\ \boxed{\times}\ \boxed{=}$$

может применяться не для одного значения x, а для всех $x>0$. Указанное свойство алгоритмов называют свойством *массовости*.

4. Обязательное требование к алгоритмам — *результативность*. Смысл этого требования состоит в том, что при точном исполнении всех указаний алгоритма процесс должен прекратиться за конечное число шагов и при этом должен быть получен какой-либо ответ на вопрос задачи.

Алгоритмы описываются разными способами, в современной практике используется довольно много алгоритмических языков различного уровня. В каждом отдельном случае выбор языка зависит от ряда обстоятельств, и прежде всего от того, какого рода алгоритмы необходимо с его помощью описать, а также для кого предназначается описание — для машины или для человека. Ниже приводится обзор некоторых употребительных способов описания алгоритмов, рассчитанных не на машину, а на человека. Уже эти простые (и не очень жестко регламентированные) способы задания алгоритмов позволяют обнаружить некоторые общие подходы к алгоритмизации — подходы, широко используемые и при составлении программ для электронно-вычислительных машин (ЭВМ).

Формулы. Математические формулы вместе с правилами их написания представляют собой своеобразный алгоритмический язык, с успехом используемый для описания вычислительных алгоритмов некоторого специального вида. Так, например, формула

$$S=\frac{\pi D^2}{2}+\pi DH \qquad (3.1)$$

задает алгоритм вычисления площади поверхности цилиндрического тела с диаметром D и высотой H. Строго говоря, формула определяет последовательность действий не столь однозначно, как того требует понятие алгоритма (в частности, свойство определенности). Вместе с тем известно, что выбор любого порядка действий при соблюдении установленных в математике правил не отразится на результате. В обычном случае порядок действий по формуле регулируется исполнителем-человеком. В алгоритмических языках, используемых в программировании для ЭВМ, обычно предусматривается возможность задания вычислений непосредственно формулами в обычном математическом смысле; это означает, что ЭВМ в данном случае «обучена» определять правильный порядок действий по виду формулы.

Алгоритм, изображенный обычной формулой, относится к некоторому, весьма узкому классу алгоритмов, называемых *линейными*. Так называются алгоритмы, последовательность операций в которых определена самой структурой алгоритма и не зависит от конкретных значений входных данных. Таким является, например, приведенный выше формульный алгоритм вычисления площади поверхности цилиндра. Одних лишь линейных алгоритмов оказывается недостаточно для описания уже самых простейших вычислительных процессов. Так, например, алгоритм вычисления значения функции

$$y=\begin{cases}\ln x, & x>1\\ 1-x, & x\leqslant 1\end{cases} \qquad (3.2)$$

(рис. 13) уже не является линейным, так как в нем заложена операция выбора одной из формул в зависимости от заданного значения x.

Табличный способ. Запись вычислительного алгоритма в форме таблицы широко используется при организации вычислений по формуле с пооперационной регистрацией промежуточных результатов (см. главу 2). В этом случае расписка формулы на последовательность элементарных действий, обеспечиваемых имеющимся в наличии вычислительным средством, есть не что иное, как определение последовательности шагов (указаний) вычислительного алгоритма. Так, табличный алгоритм вычисления по формуле (3.1) может иметь вид таблицы 3.1, где приведены вычисления площади поверхности

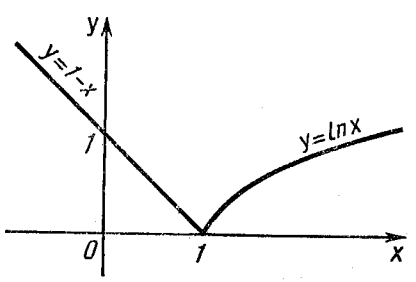


Рис. 13

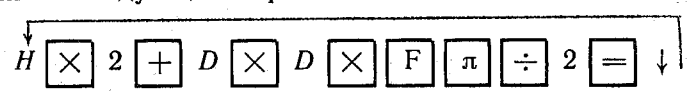
цилиндра по правилу подсчета цифр для трех пар исходных значений D и H, выраженных в сантиметрах.

Таблица 3.1

D	H	πD	πD^2	$\pi D^2/2$	πDH	S
12,6 19,3 6,8	8,9 14,2 5,4	39,58 60,63 21,4	498,7 1170 146	249,4 585,0 73,0	352 860,9 116	600 1446 190

Табличная форма записи алгоритма особенно удобна тогда, когда требуется вычислить не одно, а несколько значений одного и того же выражения для различных значений входных величин.

Программы для МК. Одновременно с привлечением для вычислений микрокалькуляторов возник (хотя и довольно-таки условный) язык для записи вычислительных алгоритмов (программ) для МК. Алгоритм для вычислений на микрокалькуляторе непрограммируемого типа — это, по сути, зафиксированный на бумаге перечень клавиш, нажатие которых в заданной последовательности приводит к решению данной вычислительной задачи. Причем, как это делалось в главах 1 и 2, ввод чисел при записи таких программ не расписывается поклавишно. В отдельных случаях есть смысл использовать дополнительные указатели, например стрелки перехода. Так, программа циклических вычислений площади поверхности цилиндра на МК типа Б3-18 по формуле (3.1) для различных значений D и H может быть изображена следующим образом:



Здесь короткой вертикальной стрелкой обозначен момент занесения результата вычислений в таблицу, а длинной — переход к началу вычисления для очередных значений исходных данных. Итоговая таблица в данном случае, помимо граф исходных данных, содержит лишь графу для записи окончательного результата (табл. 3.2).

Таблица 3.2

D	H	S
12,6 19,3 6,8	8,9 14,2 5,4	600 1446 190

Словесная запись алгоритмов. Форму словесной записи имеют многие «бытовые» алгоритмы, часто используемые в повседневной жизни: как приготовить кофе (порядок действий обычно описан на коробке), как позвонить по телефону-автомату (инструкция в телефонном справочнике) и т. п. Достоинство словесных представлений алгоритмов в том, что таким способом при желании могут быть описаны любые алгоритмы, в том числе и вычислительные. Для достижения большей четкости в словесной записи алгоритма отдельные указания (шаги) алгоритма удобно нумеровать. Для задания вычислительных действий будем использовать знак операции присваивания $:=$, употребляемый в официальных алгоритмических языках программирования. Смысл этого знака состоит в следующем: после выполнения указания $x:=A$ (читается: «x присвоить A») переменная x теряет свое прежнее значение и приобретает (присваивает) значение, полученное после выполнения всех действий, предусмотренных формулой A. Так, например, после выполнения указания $y:=x^2+3$ при $x=2$ переменная y будет иметь значение 7. При описании *нелинейных* (или *разветвляющихся*) алгоритмов используется предложение вида «если P, идти к N», где P — проверяемое условие, а N — номер одного из указаний в записи алгоритма. При составлении словесных описаний алгоритмов можно придерживаться следующего простого правила, существенно укорачивающего записи: если в процессе выполнения очередного указания явно не сообщается, к какому указанию после выполнения данного нужно перейти, то это означает, что переходить нужно к указанию, следующему за данным в порядке возрастания их номеров. Так, например, если $x=3$ и выполняется указание (с номером 4)

4. если $x<0$, идти к 7,

то произойдет переход на указание с номером 5. Для безусловного перехода к указанию с заданным номером используется указание «идти к N». Используя этот несложный аппарат, можно составлять словесные описания алгоритмов, близкие по своей сути к программам на официальных алгоритмических языках. Так, для примера, запись алгоритма вычисления по формулам:

$$y=\begin{cases}\ln x, & x>1\\ 1-x, & x\leqslant 1\end{cases}$$

может иметь вид:

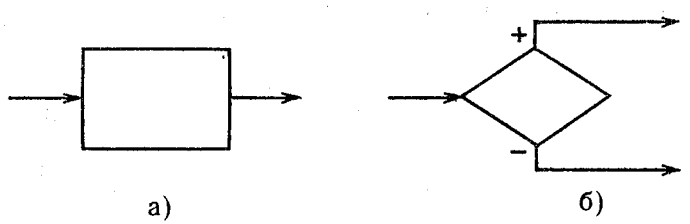
1. чтение x
2. если $x>1$, идти к 5
3. $y:=1-x$
4. идти к 6
5. $y:=\ln x$
6. запись y
7. конец

В приведенном описании использованы специальные указания — «чтение» и «запись», фиксирующие соответственно задание значений исходных данных и выдачу результатов вы-

числений. Указание «конец» означает прекращение исполнения алгоритма. Все эти указания имеют первостепенное значение в алгоритмах, адресуемых исполнителям-автоматам.

Схемы алгоритмов. Это графический способ записи, имеющий большое применение в практике составления программ для ЭВМ. Схема алгоритма представляет собой систему определенным образом связанных между собой блоков, изображенных в виде плоских геометрических фигур. Два основных типа указаний — арифметические и логические — изображаются различными фигурами (рис. 14). Арифметические указания (а) не приводят к ветвлениям и имеют всегда один выход. Логические же указания (б) специально используются для организации ветвлений и всегда имеют два выхода. Если проверяемое условие выполняется, то происходит выход по стрелке «+», если не выполняется — по стрелке «—». Логический элемент (элемент принятия решения) соответствует в обычной словесной записи алгоритма указанию «если». На рисунке 14, *а* изображен элемент общей обработки (арифметический элемент), на 14, *б* — элемент принятия решения (логический элемент).

На рисунке 15 приведена схема алгоритма вычисления значения функции по формулам (2.3). Схема состоит из одного логического элемента (один вход, два выхода) и четырех арифметических элементов (у каждого один вход и один выход). Содержание предписываемых действий, так же как и проверяемые условия, записаны внутри соответствующих блоков.



а) б)

Рис. 14

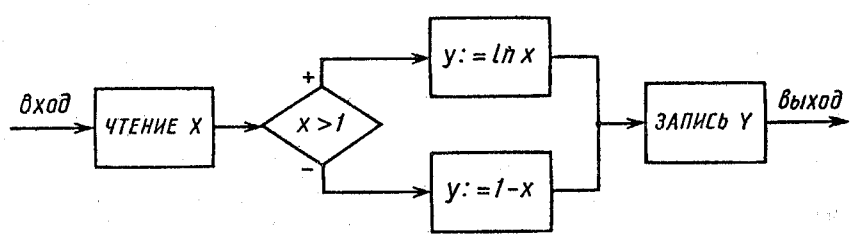


Рис. 15

Контрольные вопросы

1. Что такое алгоритм?
2. Каким основным свойствам должны удовлетворять алгоритмы?
3. В какой степени различные неформальные способы представления алгоритмов удовлетворяют свойствам этого понятия?

Упражнения

1. По заданным формулам составить вычислительные алгоритмы в виде таблицы и программы для МК: а) $S = \frac{\pi D^2}{4} + \frac{\pi DL}{2}$ (S — площадь боковой поверхности конуса; D — диаметр основания; L — образующая); б) $f = \frac{1}{2\pi\sqrt{LC}}$ (f — частота собственных колебаний в контуре, L — индуктивность катушки, C — емкость конденсатора).
2. Составить словесные описания и схемы разветвляющихся алгоритмов: а) нахождения абсолютной величины числа; б) поиска большего из двух чисел.

3.3. Конструирование алгоритмов

Наглядность и общедоступность, какими обладают схемы алгоритмов, делают этот способ записи удобным средством предварительной разработки алгоритмов для их последующего программирования на «машинных» языках. Разработка алгоритмов решения задач на ЭВМ — дело, требующее специального навыка. Алгоритм при этом должен удовлетворять следующим естественным требованиям:

— быть понятным, т. е. легко воспринимаемым; это особенно важно в тех случаях, когда приходится читать «чужие» алгоритмы;

— алгоритм должен быть легко проверяемым;

— алгоритм должен допускать возможность его модификации без существенной перестройки всей структуры.

Для достижения указанных свойств при разработке алгоритмов придерживаются особой методики, называемой *структурным подходом*. При структурном подходе к конструированию алгоритмов алгоритмы как бы «собираются» из трех основных (базовых) структур: РАЗВИЛКА, ЦИКЛ, СЛЕДОВАНИЕ, каждая из которых имеет один вход и один выход.

Базовая структура РАЗВИЛКА (см. рисунок 16) состоит из логического элемента с проверкой некоторого условия P и функциональных блоков S_1, S_2, которые в простейшем случае являются арифметическими элементами. РАЗВИЛКА может

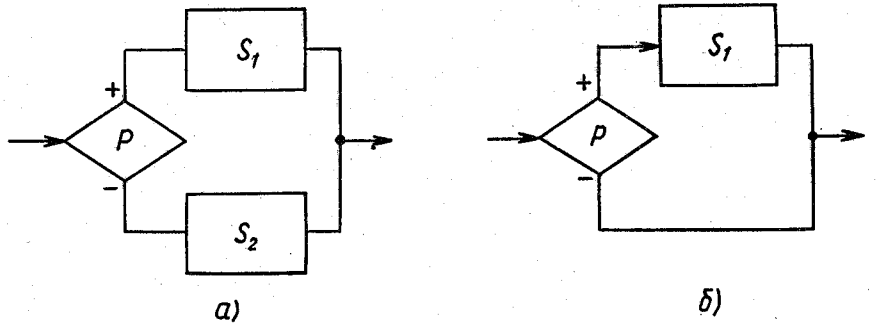


Рис. 16

быть двух видов: *полная условная конструкция* (рис. 16, *а*) и *неполная условная конструкция* (рис. 16, *б*).

Базовая структура ЦИКЛ также может быть двух видов (см. рисунок 17). В состав цикла входит логический элемент с проверкой условия P и функциональный блок S, называемый *телом цикла*. В простейшем случае S является обычным арифметическим элементом. Как следует из структуры ЦИКЛ, тело S может при определенных условиях выполняться неоднократно. В первом случае блок размещен после проверки условия P, так что может оказаться, что тело S не выполнится ни разу; этот вариант структуры ЦИКЛ называют ЦИКЛ-ПОКА (рис. 17, *а*). Здесь P является *условием продолжения цикла* (всякий раз, когда P истинно, тело S выполняется). Во втором случае блок S расположен до проверки логического условия P, так что в этом варианте цикла тело S всегда будет выполнено по крайней мере один раз; этот вариант структуры ЦИКЛ называют ЦИКЛ-ДО (рис. 17, *б*). Здесь P является *условием выхода из цикла* (как только P становится истинным, выполнение цикла завершается). Тот или иной вариант структуры ЦИКЛ используется при разработке алгоритмов также в зависимости от особенностей конкретной задачи.

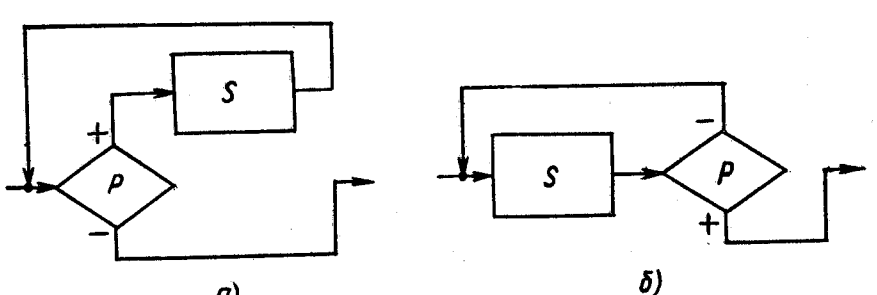


Рис. 17

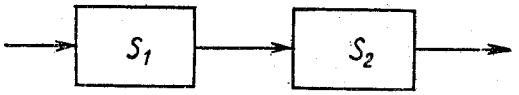


Рис. 18

Базовая структура СЛЕДОВАНИЕ (см. рисунок 18) состоит из двух функциональных блоков S_1, S_2, каждый из которых в простейшем случае может быть арифметическим элементом. Структура СЛЕДОВАНИЕ означает, что два функциональных блока могут быть размещены друг за другом.

Вслед за этим принимается соглашение: при составлении алгоритмов разрешается использовать только три указанные базовые структуры. При этом разрешается также каждый из функциональных блоков S, S_1 и S_2 заменять любой из базовых структур. Понятно, что такое соглашение позволяет строить как угодно сложные по содержанию алгоритмы, развивая их не только «вширь», но и «вглубь». На рисунке 19 изображен структурный алгоритм, который представляет собой следование функционального блока S_1 и структуры ЦИКЛ-ПОКА. При этом тело цикла представляет собой полную условную конструкцию (обведено штриховой линией).

Конструируемые таким путем алгоритмы имеют четкую и ясную структуру, легко поддаются проверке, ибо состоят из ограниченного числа различных блоков, устроенных аналогично. Рассмотрим примеры составления структурных схем алгоритмов.

Пример 3.3.1. Алгоритм поиска большего из трех чисел (рис. 20).

Алгоритм представляет собой следование двух развилок (на рисунке они выделены пунктиром). В первой (полная условная

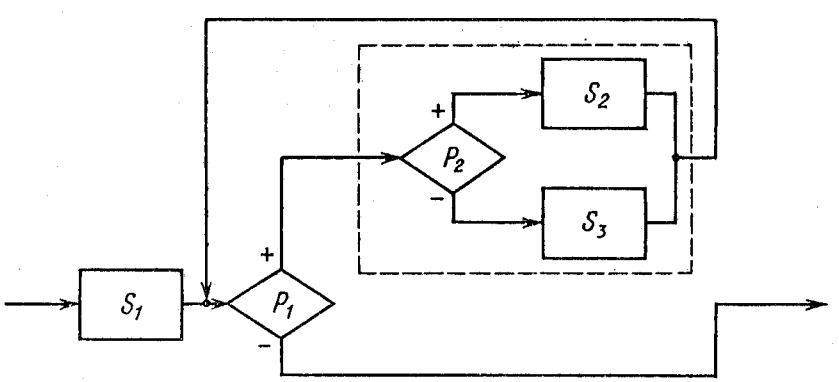


Рис. 19

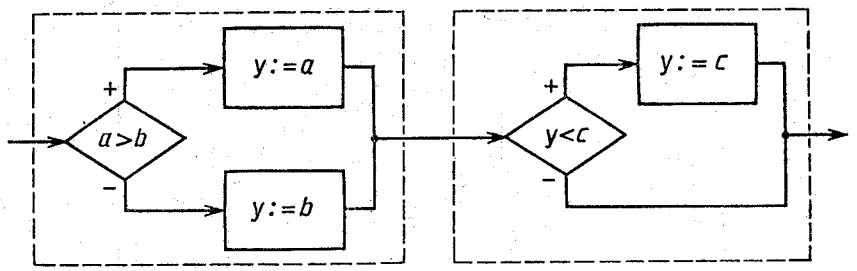


Рис. 20

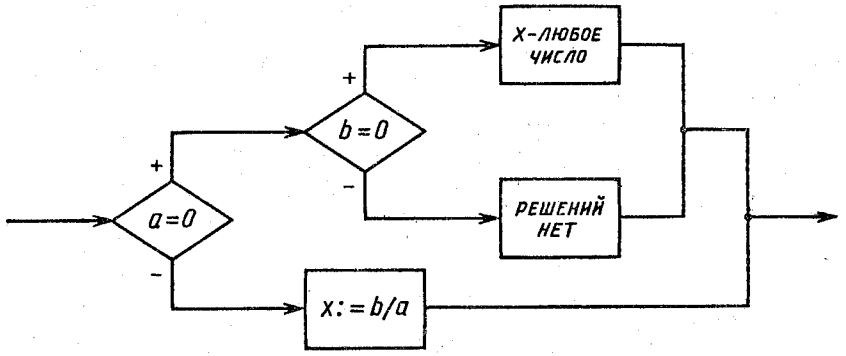


Рис. 21

конструкция) отыскивается большее из двух чисел a и b и большее из них становится значением переменной y. Во второй развилке (неполная условная конструкция) значение y сравнивается с третьим числом, и если $y<c$, то y заменяется значением c, в противном случае y остается без изменений. Таким образом, в процессе выполнения алгоритма переменная y будет иметь своим значением большее из значений a, b и c. При программировании этого алгоритма для ЭВМ должно быть предусмотрено чтение (ввод) исходных данных a, b и c, а также запись (вывод) результата. В схеме алгоритма эти действия для краткости опущены.

Пример 3.3.2. Алгоритм решения уравнения $ax=b$ в общем случае может быть описан схемой, изображенной на рисунке 21. В зависимости от значений a и b возможны три исхода: решений бесконечное множество ($a=0$ и $b=0$), решений нет ($a=0$, но $b\neq 0$), решение единственное ($a\neq 0$). Алгоритм представляет собой полную условную конструкцию, одним из функциональных блоков в которой также является полная условная конструкция.

Пример 3.3.3. В урне хранится некоторое количество черных и белых шаров. Требуется разложить эти шары по двум корзинам (черного и белого цвета): белые шары — в белую

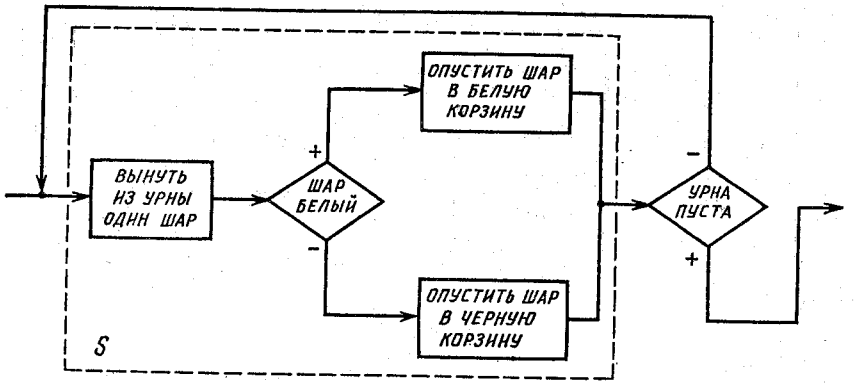


Рис. 22

корзину, черные — в черную. Составим схему алгоритма выполнения этой работы.

Очевидно, что для рассортировки шаров по корзинам с каждым шаром должны быть проделаны следующие действия: вынуть шар из урны, определить его цвет и опустить в соответствующую корзину. Если после этого в урне еще остаются шары, действия повторить снова и т. д. Схема алгоритма (рис. 22) представляет собой структуру ЦИКЛ-ДО, причем тело цикла S (расположенное в данном случае до проверки логического условия) представляет собой сложную конструкцию, состоящую из следования простого действия и полной условной конструкции. Понятно, что выполнить этот алгоритм было бы затруднительно, если бы в самом начале оказалось, что урна пуста. Если такая ситуация не исключается, то правильным решением задачи будет

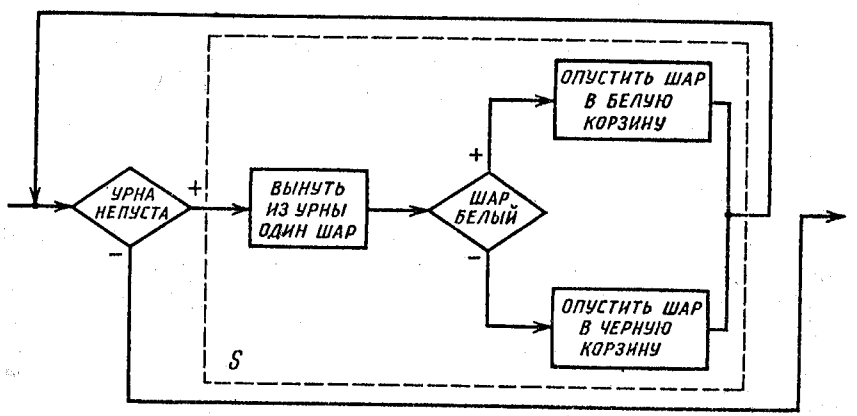


Рис. 23

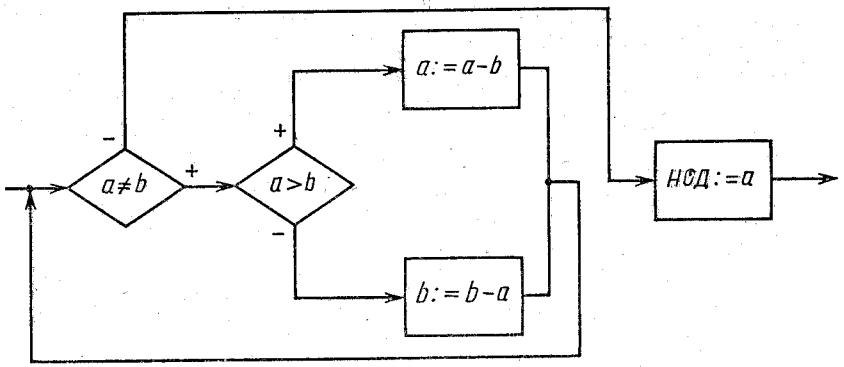


Рис. 24

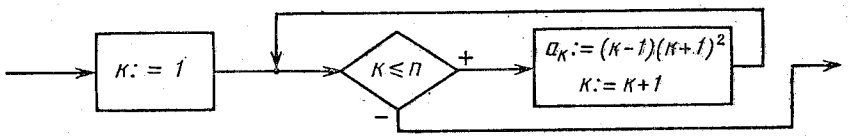


Рис. 25

схема, изображенная на рисунке 23. Здесь тело цикла S размещено после проверки условия (структура ЦИКЛ-ПОКА).

Пример 3.3.4. Алгоритм Евклида нахождения наибольшего общего делителя (НОД) двух целых положительных чисел (методом вычитания) изображен на рисунке 24.

Если числа a и b равны, любое из них можно принять за искомый результат (НОД $:=a$). Если же числа не равны, большее из них заменяется разностью между ним и другим числом, после чего снова производится проверка условия $a=b$. Схема алгоритма представляет собой следование, состоящее из ЦИКЛ-ПОКА, у которого телом является полная условная конструкция, и простого арифметического блока.

Пример 3.3.5. Вычисление (генерирование) членов последовательности, заданной формулой:

$$a_k=\frac{k-1}{(k+1)^2}\ (k=1,\ 2,\ \ldots,\ n),$$

осуществляется циклическим алгоритмом, изображенным на рисунке 25. Тело цикла содержит два арифметических оператора, изображенных для краткости в одном функциональном блоке. Алгоритм представляет собой следование функционального блока и структуры ЦИКЛ-ПОКА.

Пример 3.3.6. Дана конечная последовательность a_1, a_2,... ..., a_n (в программировании такие последовательности чисел

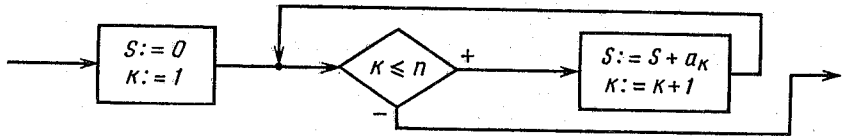


Рис. 26

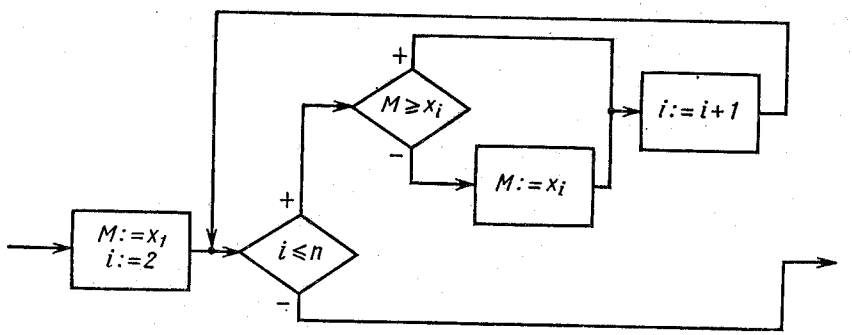


Рис. 27

называют *массивами*). Составить схему алгоритма вычисления суммы:

$$S=a_1+a_2+\ldots+a_n.$$

Если принять $S=0$, то нахождение искомой суммы обеспечивается n-кратным повторением указания

$$S:=S+a_k$$

при изменении k от 1 до n. Схема алгоритма изображена на рисунке 26.

Пример 3.3.7. Поиск большего элемента в массиве x_1, x_2, ..., x_n.

Возьмем «рабочую» переменную M и присвоим ей значение элемента x_1. После этого будем проверять истинность неравенства $M\geqslant x_i$ для $i=2,\ 3,\ \ldots,\ n$. В результате каждой проверки значение M либо оставим без изменения (если неравенство $M\geqslant x_i$ истинно), либо заменим на x_i (если неравенство ложно, т. е. $M<x_i$). Очевидно, что по окончании этого процесса M будет иметь значение наибольшего по величине элемента заданного массива. Схема алгоритма изображена на рисунке 27.

Структурный подход к составлению алгоритмов основывается на том, что каждый алгоритм может быть представлен в структурной форме. Однако не всегда в ходе поиска удается сразу получить алгоритм в структурном виде. Иногда приходится предпринять специальные приемы, преобразующие неструктурные алгоритмы в структурные. Простейший прием — *размножение блоков*.

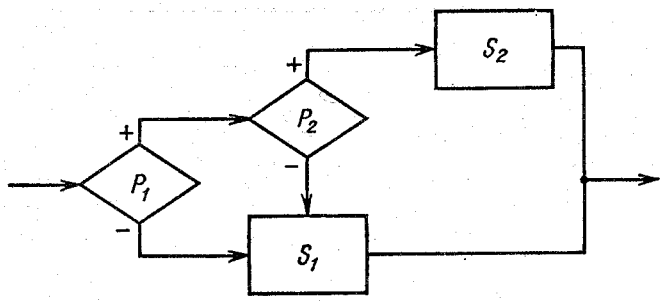


Рис. 28

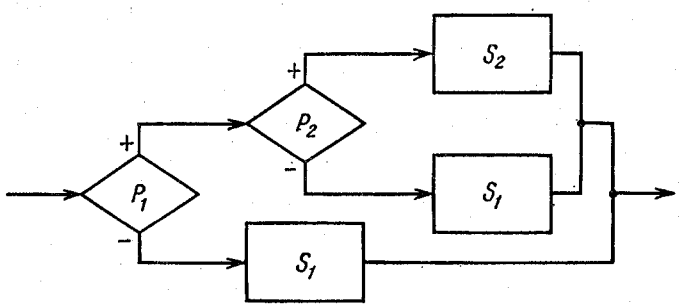


Рис. 29

На рисунке 28 изображена схема неструктурного алгоритма. Этот алгоритм легко преобразовать к структурному виду, если продублировать блок S_1. Алгоритм, изображенный на рисунке 29, эквивалентен данному в том смысле, что во всех возможных случаях он предписывает выполнять те же действия.

Иногда для получения структурных алгоритмов приходится применять более изысканные методы. Рассмотрим пример.

Пример 3.3.8. Дан массив $x_1, x_2, \ldots, x_n$. Требуется определить, имеется ли в этом массиве хотя бы одна пара взаимно обратных соседних чисел.

Судя по постановке задачи, надо циклически проверять выполнение равенства $x_i\, x_{i+1} = 1$ при $i = 1, 2, \ldots, n-1$. Как только равенство окажется истинным, процесс прекращается и выдается сигнал о том, что искомая пара элементов имеется — сигнал «да». Если же процесс доходит до конца, но равенство не выполняется, должен быть выдан сигнал «нет». Подходя к организации этого циклического алгоритма обычным образом (т. е. начиная цикл с проверки условия продолжения $i \leqslant n-1$), получим схему, изображенную на рисунке 30. Этот алгоритм действует верно, но он не является структурным. Причина неструктурности в данном случае состоит в том, что имеется два условия окончания цикла: 1) исчерпание всех элементов; 2) выполнение равенства

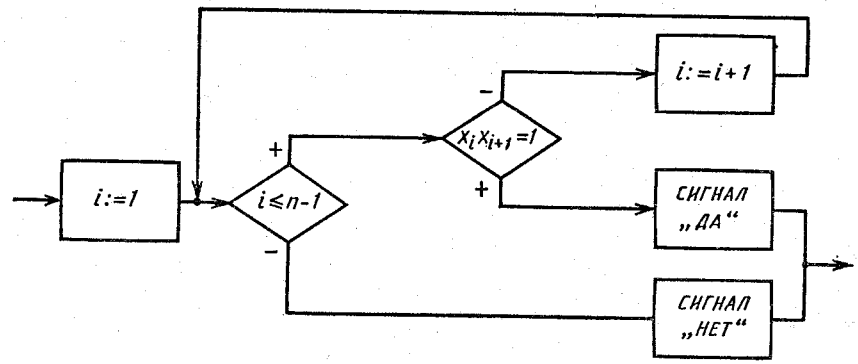


Рис. 30

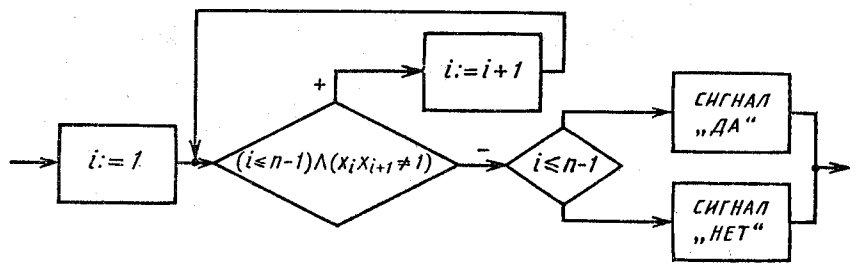


Рис. 31

$x_i\, x_{i+1} = 1$. В зависимости от того, каким из этих двух условий закончился цикл, необходимо предпринимать различные действия. Один из способов преодоления подобного затруднения — объединение обоих условий продолжения цикла в одно:

$$(i \leqslant n-1) \wedge (x_i\, x_{i+1} \neq 1),$$

а для принятия окончательного решения по окончании цикла необходимо проверить, какое именно условие привело к выходу из цикла (рис. 31). Рассмотренный подход дает структурный алгоритм, однако программирование проверки объединенного в конъюнкцию общего условия приведет к новым затруднениям, если в алгоритмическом языке отсутствуют логические операции. Другой подход — введение дополнительной переменной, выполняющей роль признака, который и используется для управления переходами в соответствии со смыслом задачи (рис. 32).

Контрольные вопросы

1. Каким требованиям должна удовлетворять разработка алгоритмов на ЭВМ?
2. В чем суть основных структур алгоритмов: РАЗВИЛКА, ЦИКЛ, СЛЕДОВАНИЕ?

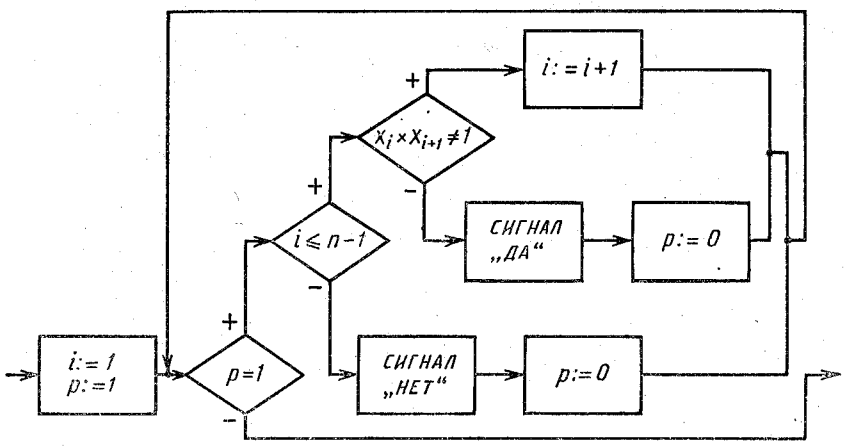


Рис. 32

3. В чем заключается структурный подход к разработке алгоритмов? Как обеспечивается соблюдение основных требований к разработке алгоритмов с использованием этого подхода?

4. Каковы основные приемы преобразования неструктурных алгоритмов в структурные?

Упражнения

Составить схемы алгоритмов решения следующих задач:

1. Вычислить корни уравнения $ax^2+bx+c=0\ (a\neq 0)$.

2. Вычислить одно значение функции:

а) $y=\begin{cases} 1-z^2, & z\leqslant 0 \\ z^2-1, & z>0, \end{cases}$ где

$$z=\begin{cases} \sin x, & x\geqslant 1 \\ e^x, & x<1, \end{cases}$$

б) $$y=\begin{cases} x^2+1, & x<0 \\ 2x+1, & 0\leqslant x\leqslant 1 \\ 4x-1, & x>1. \end{cases}$$

3. Решить неравенства: а) $ax<3$; б) $ax>b$.

4. Вычислить 50 членов последовательности с общим членом:

$$a_k=\frac{2k}{k^2+1}\ (k=1,\ 2,\ \ldots,\ 50).$$

5. Вычислить произведение элементов массива $x_1,\ x_2,\ \ldots,\ x_{100}$.

6. Определить, имеется ли в массиве $a_1,\ a_2,\ \ldots,\ a_n$ хотя бы одна пара противоположных соседних чисел.

Глава 4
ЭЛЕМЕНТЫ ПРОГРАММИРОВАНИЯ НА ЯЗЫКЕ БЕЙСИК

4.1. Основные устройства ЭВМ

Процесс автоматической обработки информации (решение задачи с помощью ЭВМ представляет собой именно такой процесс) включает следующие обязательные операции: ввод (чтение) исходной информации, запоминание и хранение информации, обработка информации (собственно решение задачи), управление процессом обработки, вывод результатов. Для выполнения отмеченных операций ЭВМ имеет в своем составе как минимум следующие основные устройства:

а) запоминающее устройство (память) (ЗУ);
б) арифметико-логическое устройство (процессор) (АЛУ);
в) устройство управления (УУ);
г) устройства ввода-вывода (УВВ).

В различных ЭВМ эти основные устройства реализуются в специфических для разных классов ЭВМ технических решениях. В весьма общем и упрощенном виде схема связей между устройствами ЭВМ изображена на рисунке 33.

Память машины служит для хранения всей необходимой информации — исходных, промежуточных и окончательных числовых значений, а также программы, т. е. алгоритма, представленного в специальном виде.

Память ЭВМ обычно состоит из двух частей: *оперативной* и *внешней*. Это разделение связано со скоростью выборки информации и не имеет принципиального значения при изучении начал программирования.

Процессор обеспечивает выполнение операций над исходными данными. Он получает исходные данные из памяти и по ним

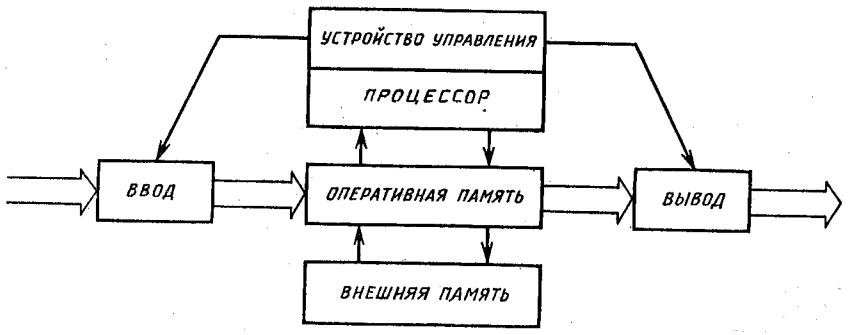


Рис. 33

вырабатывает промежуточные и окончательные результаты, которые снова отправляются в память, и выполняет только те операции, которые определены для данной машины ее конструктивными возможностями. В некоторых микроЭВМ в понятие «процессор» наряду с собственно устройством переработки информации включаются также и элементы других устройств машины: память, ввод-вывод, устройство управления. Характерная особенность процессоров — высокое быстродействие. Быстродействие ЭВМ высокого класса определяется миллионами операций в секунду.

Устройство управления координирует работу всех устройств машины: вызывает из памяти сведения о выполнении очередной операции, расшифровывает эти сведения, вызывает из памяти участвующие в операции числа и отправляет их в процессор, а затем пересылает в память полученный результат.

Устройства ввода-вывода осуществляют связь человека с машиной. Вся исходная информация перед решением задачи записывается в памяти с помощью устройства ввода. В одних случаях для этой цели используются промежуточные носители информации, на которые информация заносится заранее: перфоленты, перфокарты, магнитные ленты, магнитные диски и т. п. Некоторые ЭВМ снабжаются пультовой пишущей машинкой, похожей на обычную, с помощью которой исходные данные и программа (на особом языке) могут быть введены в память непосредственно через клавиатуру с параллельной печатью на бумаге. Эта же машинка может отпечатать и результаты счета. Для печати буквенно-цифровой информации ЭВМ снабжаются также специальными быстродействующими построчными печатающими устройствами.

Для диалогового режима работы многие ЭВМ снабжаются устройствами визуального отображения информации — *дисплеями*, которые имеют экран с изображением алфавитно-цифровой и графической информации и клавиатуру для ввода данных.

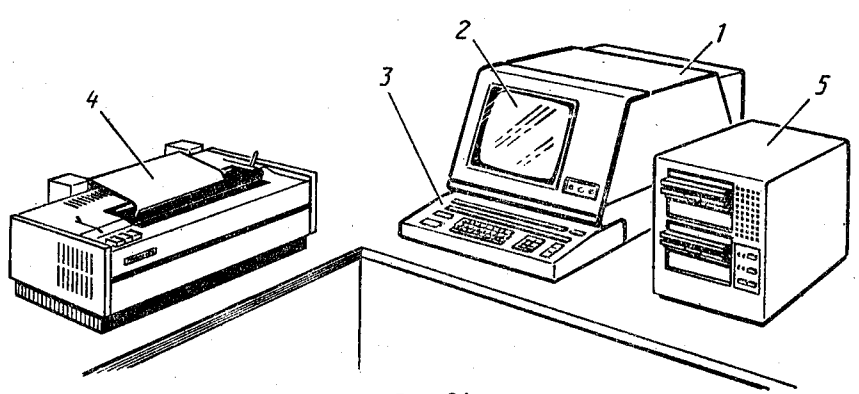


Рис. 34

Комплексы устройств ввода-вывода, которыми снабжаются современные ЭВМ, позволяют хранить, обрабатывать и выдавать информацию в самой разнообразной форме: в форме графиков, в виде таблиц и прочих бумажных документов, в виде изображения на экране, в виде перфокарт, перфолент, магнитных лент и магнитных дисков и т. п., пригодных для нового ввода в машину.

В зависимости от вычислительных возможностей (объем памяти, быстродействие), габаритов, комплекса периферийных устройств ЭВМ принято подразделять на большие, мини-ЭВМ и микроЭВМ. В настоящее время получает широкое распространение в различных сферах хозяйственной, научной и учебной деятельности (в том числе в высших и средних учебных заведениях) новое поколение малогабаритной вычислительной техники — микроЭВМ, к числу которых относятся отечественные профессиональные и персональные микроЭВМ серий «Электроника», «Искра», ЕС и др. Обладая малыми габаритами (микроЭВМ обычно размещается на столе). Эти машины благодаря возможности использования развитого языка программирования и широкого перечня периферийных устройств позволяют эффективно решать разнообразные плановые, экономические и научно-технические задачи.

На рисунке 34 изображен минимальный комплект устройств микроЭВМ «Искра-226»: процессор (*1*), включающий конструктивно символьно-графический дисплей (*2*), и клавишное устройство (*3*), печатающее устройство (*4*), а также блок внешней памяти — накопитель на гибких магнитных дисках (*5*). Размер экрана дисплея — 31 см по диагонали, формат 80×24 символа или 560×256 точек.

Контрольные вопросы

1. Какие обязательные составные части включает в себя процесс автоматической обработки информации?
2. Какие основные устройства входят в состав программно-управляемой вычислительной машины?
3. Каково назначение основных устройств ЭВМ: памяти, процессора, устройства управления, устройств ввода-вывода?
4. Какой минимальный набор устройств включает микроЭВМ «Искра-226»?

4.2. Алфавит и простейшие конструкции языка Бейсик

Алгоритмический язык Бейсик предназначен для решения большого круга математических и инженерных задач в режиме диалога человек-ЭВМ. Программы на языке Бейсик могут быть реализованы на микроЭВМ типа «Электроника-60», «Электроника ДЗ-28», «Искра-226» и др., получающих в настоящее время широкое распространение в учебных заведениях. Рассмотрим первоначальные сведения о языке Бейсик, необходимые для составления простейших программ.

1. Алфавит. Символы, воспринимаемые ЭВМ на языке Бейсик, могут быть подразделены на следующие группы:

а) прописные латинские буквы A, B, C, ..., Z;

б) прописные русские буквы А, Б, В, ..., Я (кроме Ё и Ъ);

в) цифры Ø, 1, 2, 3, 4, 5, 6, 7, 8, 9*;

г) знаки арифметических операций

↑ (возведение в степень)**;
* (умножение); / (деление);
+ (сложение); — (вычитание);

д) знаки отношений

= (равно); > (больше);
< (меньше); >= (больше или равно);
<= (меньше или равно); <> (не равно);

е) специальные символы . , : ; " ' () [] ! ? %***;

ж) служебные символы «пр.» (пробел), «в. к.» (возврат каретки);

з) буква π, имеющая значение числа 3,14159265359.

Клавишное устройство ЭВМ «Искра-226», в которой ввод символов сопровождается их отображением на экране дисплея, отождествляет возврат каретки с переходом на новую строку; для этих целей клавиатура «Искра-226» имеет клавишу CR/LF (рис. 35).

2. Числа. В языке Бейсик используются *целые* и *действительные* числа. Знак «+» перед числом не ставится, а целая часть при записи десятичных дробей отделяется точкой. Нулевую часть дробного числа можно опускать. Примеры записи чисел:

целые	действительные
Ø	.Ø1
143	12.
—362	—24.356

Действительные числа могут представляться в экспоненциальной форме. Так, например, число 0,005342 может быть представлено в любой из следующих форм:

.5342E—2
.ØØØØ5342E2
534.2E—5

Здесь буква E имеет смысл «возвести 10 в степень». Показатель степени (порядок) может не иметь знака, если он положителен; отрицательный порядок обязательно должен иметь знак «—».

3. Переменные. Числовая переменная в Бейсике обозначается любой буквой латинского алфавита или любой буквой, за которой следует одна цифра. Например

*В языке Бейсик для ЭВМ «Искра-226» используются десятичная и шестнадцатеричная системы исчисления.

**В некоторых версиях Бейсика операция возведения в степень обозначается символом ^.

***Алфавит языка Бейсик может включать и другие специальные символы, которые здесь не приводятся из-за трудностей набора.

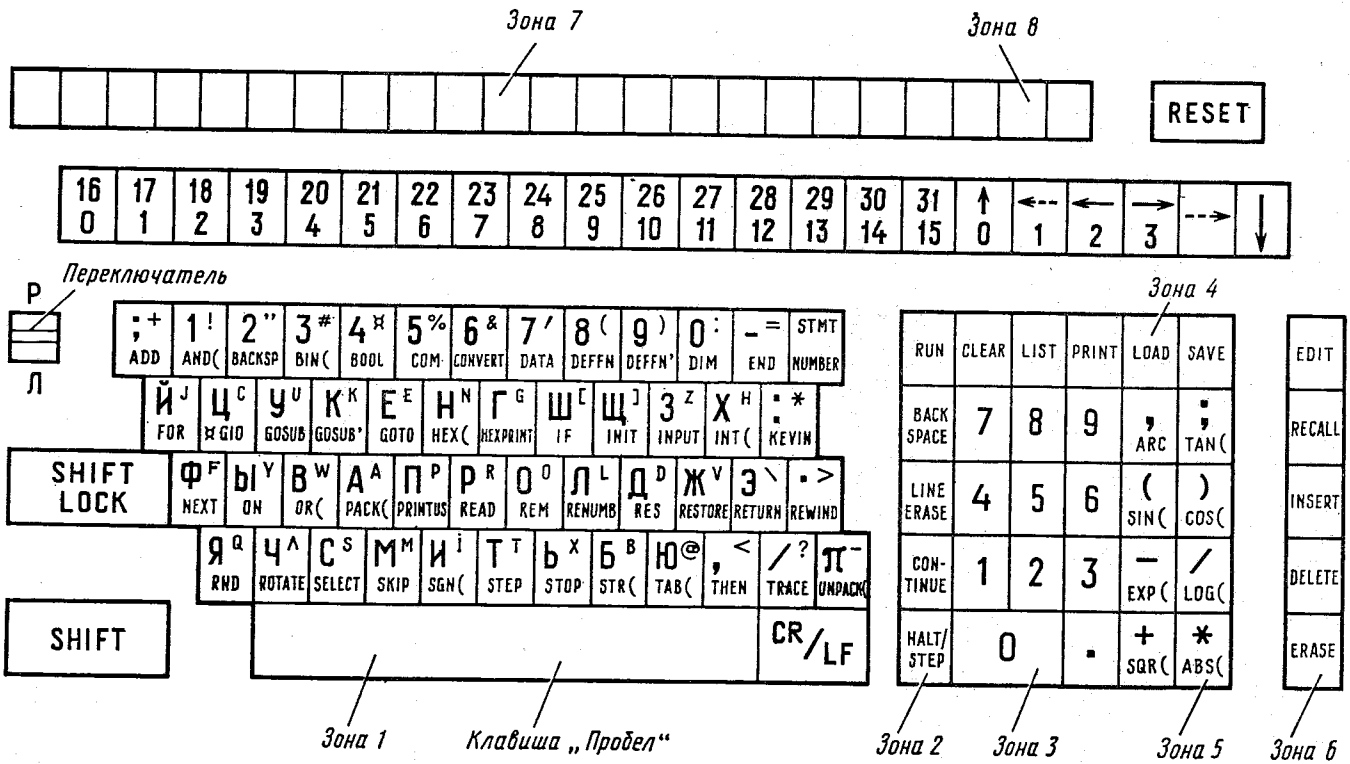

Зона 7
Зона 8
RESET
16 0
17 1
18 2
19 3
20 4
21 5
22 6
23 7
24 8
25 9
26 10
27 11
28 12
29 13
30 14
31 15
↑ 0
← 1
← 2
→ 3
Переключатель
Р
Л
; + ADD
1 ! AND(
2 " BACKSP
3 # BIN(
4 ¤ BOOL
5 % COM
6 & CONVERT
7 ' DATA
8 (DEFFN
9) DEFFN'
0 : DIM
- = END
STMT NUMBER
Й J FOR
Ц C ¤GIO
У U GOSUB
К K GOSUB'
Е E GOTO
Н N HEX(
Г G HEXPRINT
Ш [IF
Щ] INIT
З Z INPUT
Х H INT(
: * KEVIN
SHIFT LOCK
Ф F NEXT
Ы Y ON
В W OR(
А A PACK(
П P PRINTUS
Р R READ
О O REM
Л L RENUMB
Д D RES
Ж V RESTORE
Э \ RETURN
. > REWIND
Я Q RND
Ч ^ ROTATE
С S SELECT
М M SKIP
И İ SGN(
Т T STEP
Ь X STOP
Б B STR(
Ю @ TAB(
, < THEN
/ ? TRACE
π ~ UNPACK(
SHIFT
CR/LF
Зона 1
Клавиша „Пробел“
Зона 4
RUN
CLEAR
LIST
PRINT
LOAD
SAVE
BACK SPACE
7
8
9
, ARC
; TAN(
LINE ERASE
4
5
6
(SIN(
) COS(
CON-TINUE
1
2
3
— EXP(
/ LOG(
HALT/ STEP
0
.
+ SQR(
* ABS(
EDIT
RECALL
INSERT
DELETE
ERASE
Зона 2
Зона 3
Зона 5
Зона 6

Рис. 35

A I X5 B2

Совокупность букв и цифр, обозначающих переменную, называют ее *именем* (или *идентификатором*). В ЭВМ «Искра-226» переменные, как и числа, могут быть *целыми* и *действительными**. Значениями целых переменных всегда являются целые числа, значениями действительных — действительные. Признаком целой переменной в ее обозначении является присутствие символа %, располагаемого вслед за именем переменной. Примеры простых переменных целого типа:

A% I% X2%

Мы рассмотрели примеры безиндексных, или *простых,* переменных. Кроме простых переменных, могут использоваться *переменные с индексами,* служащие для обозначения элементов массивов. Переменная с индексом обозначается именем массива, за которым в круглых скобках указываются индексы — числовые или буквенные. Именем массива может быть любая простая переменная. Примеры:

A(2), B%(1, 3), X5(2*I, J).

В языке Бейсик допускается использование только одномерных и двухмерных массивов. В общем случае индексом в переменной с индексами может быть произвольное арифметическое выражение.

4. Арифметические выражения составляются из чисел и переменных с помощью знаков арифметических действий и круглых скобок.

Например:

Обычная запись:	Запись на Бейсике:
$5x$	5*X
ax^2+bx+c	A * X↑2 + B * X + C
$a_i\,(x^2+14{,}3y_j)$	A (I) * (X↑2 + 14.3 * Y (J))

Внутри скобок и в бесскобочных записях порядок действий совпадает с общепринятым: сначала выполняются все возведения в степень, потом умножения и деления и, наконец, сложения и вычитания. Операции одного приоритета выполняются слева направо.

Наряду с числами и переменными в состав арифметических выражений могут входить *стандартные функции* Бейсика. Каждая из функций обозначается соответствующим именем, за которым следует аргумент в круглых скобках. Ниже приводится перечень стандартных функций Бейсика ЭВМ «Искра-226».

*В этой версии Бейсика имеется третий тип констант и переменных — *символьный,* наличие которого позволяет обрабатывать не только числовую, но и символьную информацию; в настоящем пособии символьный тип не рассматривается.

Название функции	Обозначение
Синус X	SIN (X)
Косинус X	COS (X)
Тангенс X	TAN (X)
Арксинус X	ARCSIN (X)
Арккосинус X	ARCCOS (X)
Арктангенс X	ARCTAN (X)
Случайное число между 0 и 1	RND (X)
Абсолютное значение X	ABS (X)
Целая часть X	INT (X)
Сигнум (знак) X	SGN (X)
Натуральный логарифм X	LOG (X)
Экспонента X (e^X)	EXP (X)
Квадратный корень из X	SQR (X)

Аргументом X может быть число, переменная, арифметическое выражение (в том числе содержащее функции). Тригонометрические функции в обычных условиях вычисляются для аргументов, выраженных в радианах*. Аргумент функции RND (X) не используется и может быть любым числом. Функция SGN (X) вычисляется по правилу:

SGN (X) = 1, если X = 0
SGN (X) = 0, если X = 0
SGN (X) = −1, если X < 0

Примеры записи арифметических выражений с функциями.

Обычная запись:	Запись на Бейсике:
$3\sin\frac{x}{2}$	3*SIN (X/2)
$\frac{-b-\sqrt{b^2-4ac}}{2a}$	(−B − SQR (B↑2 − 4*A*C))/(2*A)
$\operatorname{arctg}(3x-2e^{\lvert x-1{,}8\rvert})$	ARCTAN (3*X − 2*EXP (ABS (X − 1.8)))

5. Условия. Наряду с арифметическими выражениями, значениями которых являются десятичные числа, в языке Бейсик используются также логические выражения, принимающие только два значения — «истинно» и «ложно». В частном случае логиче-

*В версии Бейсика ЭВМ «Искра-226» при желании может быть установлен любой из трех режимов вычислений тригонометрических и обратных тригонометрических функций: градусы, радианы или грады (1 град = π/200). Для этой цели используется специальный оператор-описатель SELECT (SELECT D — градусы, SELECT R — радианы, SELECT G — грады). При отсутствии указаний в форме оператора SELECT автоматически устанавливается режим R (радианы).

ское выражение представляет собой два арифметических выражения, соединенных между собой одной из операций отношения: $=$, $<$, $<=$, $>$, $>=$, $<\ >$. Такой вид логического выражения называют *простым условием*. Примеры простых условий:

Обычная запись:	Запись на Бейсике:
$x \geqslant 0$	X>=0
$ax^2 = bx + c$	A*X↑2=B*X+C
$\ln x < 2 \sin x$	LOG (X)<2*SIN (X)

Контрольные вопросы

1. Какие основные группы символов составляют алфавит языка Бейсик?
2. Какие типы чисел используются в языке Бейсик и как они записываются?
3. Как обозначаются простые (целые и действительные) переменные? переменные с индексами?
4. Как определяется порядок действий в арифметических выражениях?
5. Как составляются простые условия в языке Бейсик?

Упражнения

1. Записать на Бейсике следующие числа:

а) 0,0094; в) 10^{-3};
б) $-13{,}037$; г) $-5{,}86 \cdot 10^4$.

2. Выделить из заданных чисел целые и действительные:

а) -0 г) .425
б) 0.0 д) -265
в) .425 е) $-26.3E5$

3. Записать на языке Бейсик следующие арифметические выражения:

а) $x^{1{,}37}$; в) $(0{,}1275 + 2 \sin \sqrt[4]{\alpha})^3 + 2^{\cos \beta}$;

б) $a_0 + a_1 x + a_2 x^2$; г) $\dfrac{\sqrt{\sin^2 (x-y) - 1}}{1 + |x^2 - y^2|}$.

4. Записать на языке обычных математических обозначений:

а) (S−A*X)/B в) ABS (X−Y)/(1+X)↑(A↑2)
б) A/B/C*D*E г) (4*A (I, J) ↑(−2))/LOG(ABS(X))

5. Даны текущие значения переменных: X=2, Y=3, A=5. Определить истинность условий:

а) X<=2 в) A+X↑2=Y↑2
б) X*Y>X↑Y г) INT (Y/X)=Y−X

4.3. Организация программы

Программа на языке Бейсик представляет собой последовательность *строк*. Каждая строка программы снабжается десятичным номером, которые могут принимать значения от 0 до 9999. Выполнение программы осуществляется в порядке возрастания номеров строк*, причем не требуется, чтобы строки программы нумеровались непременно последовательными натуральными номерами. Для того чтобы облегчить в процессе составления программы вставление новых строк между уже имеющимися, строки программы обычно нумеруют с каким-либо шагом, например через 10 номеров: 10, 20, 30 и т. д.

Основными компонентами программы являются *операторы*, из которых и состоят строки программы. Одна строка программы может содержать один или несколько операторов, которые отделяются друг от друга двоеточием. Каждый оператор состоит из двух частей: *служебного слова* (имени) и информационной части оператора, называемой *телом*. Наиболее употребительный оператор, обеспечивающий в программах вычисления,— это *оператор присваивания*, который в языке Бейсик имеет вид:

LET <переменная> = <арифм. выражение>**.

Действие оператора LET заключается в присваивании переменным, стоящим слева, значений арифметических выражений (в частности, арифметическое выражение может быть числом или отдельно взятой переменной), стоящих справа от знака равенства.

Примеры:

LETI% = 1 — целой переменной I% присваивается значение 1;
LETA = 23.7 — действительной переменной A присваивается значение 23,7;
LETX = A * SIN (T) — действительной переменной X присваивается значение выражения $A \sin T$ при текущих значениях A и T.

С помощью оператора LET можно присвоить одно и то же значение не только одной переменной, но и нескольким переменным сразу; в этом случае переменные перечисляются в левой части оператора LET через запятую. В ЭВМ «Искра-226» служебное слово LET в записи оператора присваивания может опускаться.

В программах для ЭВМ обычно предусматривается ввод значений исходных данных *(аргументов)* и вывод *результатов*. Для этого используется оператор INPUT, имеющий вид:

INPUT <список переменных>.

Переменные в <списке переменных> разделяются запятыми. Встретив в программе оператор INPUT, ЭВМ делает паузу, выво-

*Если не предусмотрено программное изменение последовательности выполнения строк.

**Перевод английских служебных слов дан в конце пособия (Приложение III).

дит знак вопроса и ждет, когда с клавиатуры устройства ввода будет введено столько значений, сколько переменных в списке оператора INPUT. После ввода последнего из них нажимается клавиша перехода на новую строку, и ЭВМ продолжит выполнение программы.

Вывод (печать) результатов осуществляется в языке Бейсик оператором PRINT, имеющим вид:

PRINT <список>,

где <список> может содержать переменные, арифметические выражения (в частности, числовые константы), строки текста или то и другое. Если список оператора PRINT состоит из переменной, то при выполнении оператора произойдет вывод значения этой переменной. Если список содержит выражение, например:

PRINT X+ARCTAN (T),

то оператор PRINT произведет все предусмотренные выражением операции и выдаст полученный результат (число). Наличие запятой между элементами списка оператора PRINT показывает, что информация выводится в *зонном формате* (длина зоны — 16 символов; при выводе информации на экран дисплея ЭВМ «Искра-226» следует иметь в виду, что одна строка вмещает 80 символов). Элемент списка оператора PRINT, перед которым имеется запятая, выводится в начало следующей зоны. Если между элементами строки оператора PRINT стоит точка с запятой, то информация выводится в *уплотненном формате*. Важной особенностью оператора PRINT является то, что этот оператор может использоваться для вывода сообщения, комментария или любой строки символов. Для этого текст, который должен выводиться, заключается в кавычки. Пусть, например, текущее значение переменной P в программе равно 46. Тогда в результате выполнения оператора

PRINT "ПРОПУЩЕНО ЧАСОВ"; P

будет выведен текст:

ПРОПУЩЕНО ЧАСОВ 46

В текст программы можно включать строки с необходимыми пояснениями, комментариями, что облегчает чтение программы. Для этой цели используется оператор REM, записываемый следующим образом:

REM <цепочка символов>,

где <цепочка символов> может содержать текст (без кавычек), составленный из любых символов, кроме двоеточия. Оператор REM не исполняется машиной и может размещаться в любом месте программы. Пример:

14Ø REM БЛОК СИГМА (X)

Составим теперь несложную вычислительную программу.

Пример 4.3.1. Пусть D — диаметр основания цилиндра, а H — его высота. Площадь поверхности цилиндра S и объем V вычисляются по формулам:

$$S=\frac{\pi D^2}{2}+\pi DH, \quad V=\frac{\pi D^2}{4}\cdot H.$$

Программа вычислений значений S и V по задаваемым значениям D и H может иметь вид:

```
1Ø REM ЦИЛИНДР
2Ø INPUT D, H
3Ø LET S=π * D ↑2/2+π * D * H
4Ø LET V=π * D ↑2 * H/4
5Ø PRINT "S="; S, "V="; V
6Ø END
```

При вводе значений $D=0.6$, $H=1$ машина выведет следующие результаты:

S=2.45Ø4422698 V=.282743338823

Программа состоит из 6 строк. Первая строка содержит неисполняемый оператор REM с комментариями, выполняющими роль заголовка программы. Во второй строке расположен оператор ввода INPUT, с помощью которого определяются начальные значения переменных D и H. В строках с номерами 30 и 40 расположены операторы присваивания, вычисляющие искомые значения переменных S и V (символ π, имеющийся на клавиатуре ЭВМ «Искра-226», на экране дисплея и в печати изображается в форме #PI). И наконец, в строке с номером 50 расположен оператор вывода PRINT, который составлен таким образом, что машина выведет не просто сами значения S и V, а снабдит эти значения принятыми для подобных случаев обозначениями: $S=$, $V=$. В последней строке программы поставлен оператор END, который обозначает конец программы. Использование этого оператора в конце текста программы не обязательно, так как выполнение программы автоматически заканчивается после выполнения всех ее строк. Однако при наличии оператора END по окончании счета ЭВМ дает сообщение об объеме свободной памяти.

Приведенная выше программа ЦИЛИНДР составлена так, что в результате ее выполнения вычисляется лишь одна пара значений S и V. Если бы понадобилось вычислить значения S и V для другой пары значений переменных D и H, программу пришлось бы пускать заново. Однако программу нетрудно видоизменить так, чтобы ее без повторного пуска можно было бы использовать для новых вычислений. Достаточно добиться, чтобы после выполнения оператора вывода (строка 50) снова выполнялся оператор ввода

(строка 20). Это достигается заменой оператора END (строка 60) оператором безусловного перехода:

GOTO <номер строки>.

Действие этого оператора заключается в переходе к строке с указанным номером. Преобразованная программа будет иметь вид:

```
1Ø REM ЦИЛИНДРЫ
2Ø INPUT D, H
3Ø LET S=π * D ↑2/2+π * D * H
4Ø LET V=π * D ↑2 * H/4
5Ø PRINT "S="; S, "V="; V
6Ø GOTO 2Ø
```

Программу ЦИЛИНДРЫ можно использовать для неоднократных вычислений, так как каждый раз после вывода очередной пары значений S и V машина будет переходить к строке 20 и запрашивать новые значения исходных данных D и H. Ниже приведен протокол выполнения программы для двух пар значений D и H:

```
? .6,1
  S=2.45Ø442698          V=.282743339923
? 1.4, 2.3
  S=13.19468914Ø8        V=3.54Ø574Ø2Ø595
```

Вычисления в данном случае можно организовать и так, чтобы все исходные данные (если их не слишком много) были сразу размещены в гексте программы и автоматически выбирались машиной по мере необходимости. Для этой цели имеется пара специальных операторов DATA и READ, которые всегда используются совместно и имеют вид:

DATA <список констант>
READ <список переменных>

Строка с оператором DATA вводит в программу список констант, которые перечисляются после слова DATA через запятую, например:

DATA 2.4, 3.2, 10.3, 2.71E — 3

Сами по себе операторы DATA никакого действия не вызывают и могут располагаться в любом месте программы. Чтение размещенных в операторе DATA констант осуществляется оператором READ, содержащим перечень переменных, расположенных вслед за словом READ через запятую, например:

READ A, B, C, D

При выполнении программы операторы DATA игнорируются до тех пор, пока не встретится оператор READ. Затем отыскивается первый по порядку оператор DATA и перечисленные в нем константы в порядке их следования присваиваются переменным в операторе READ. Так, если в программе будут использованы указанные выше примеры операторов DATA и READ, то в результате выполнения оператора READ произойдут следующие присваивания: $A=2$, $B=4$, $C=3.2$, $D=10.3$ (значение 2.71E — 3 в этом случае остается неиспользованным). Операторов DATA и READ в программе может быть несколько. Однако количество констант, вводимых в программу операторами DATA, должно быть не меньше, чем количество переменных, содержащихся в операторах READ. Если оператор DATA содержит больше значений, чем имеется переменных в списке оператора READ, то следующий оператор READ начинает присвоение с первого не использованного при предыдущем чтении значения в операторе DATA*. Если же окажется, что оператор READ содержит больше переменных, чем имеющихся в операторе DATA значений, то разыскивается следующий оператор DATA, а в случае его отсутствия выдается сообщение об ошибке (см. Приложение).

Ниже приводится вариант программы ЦИЛИНДРЫ, в котором использованы операторы DATA и READ:

```
1Ø REM ЦИЛИНДРЫ
2Ø READ D, H
3Ø S=π * D ↑2/2+π * D * H
4Ø V=π * D ↑2 * H/4
5Ø PRINT "S="; S, "V="; V
6Ø GOTO 2Ø
7Ø DATA .6, 1, 1.4, 2.3, .9, 11.2
```

В ходе выполнения программы переменные D и H сначала получат значения соответственно 0,6 и 1, затем 1,4 и 2,3 и т. д. Каждый раз при повторном прохождении программы оператор READ будет обращаться к первой неиспользованной паре данных в операторе DATA. Печать результатов для заданных оператором DATA исходных значений будет иметь вид:

```
S=2.45Ø4422698      V=.282743339923
S=13.19469814Ø8     V=3.54Ø574Ø2Ø595
S=32.9395987289     V=7.125132138343
```

После исчерпания всех чисел в операторе DATA машина выдаст сообщение ERR 27 и остановится.

Пример 4.3.2. В треугольнике даны длины сторон A и B и величина угла между ними X. Нужно вычислить длину стороны C, площадь треугольника S и радиус описанной окружности R. Пусть для определенности $A=8.7$, $B=17.3$, $X=68°$. Программа имеет вид:

*В языке Бейсик имеется оператор RESTORE, с помощью которого взаимосвязь между операторами READ и DATA в программах может устанавливаться более сложным способом (см. [14]).

```
1Ø REM ТРЕУГОЛЬНИК
2Ø SELECT D
3Ø READ A, B, X
4Ø C=SQR (A*A+B*B-2*A*B*COS (X))
5Ø S=A*B*SIN (X)/2
6Ø R=C/(2*SIN (X))
7Ø PRINT "C="; C; "S="; S; "R="; R
8Ø DATA 8.7, 17.3, 68
```

В соответствии с формой задания значения X в строке 20 размещен оператор SELECT, устанавливающий в качестве единицы измерения аргументов тригонометрических функций градусы (режим D).

В результате выполнения программы для заданных значений A=8.7, B=17.3, X=68° ЭВМ выведет значения:

C=16.193Ø82574Ø5 S=69.7752Ø9754 R=8.7324Ø1Ø73578

Понятно, что если исходные данные A, B и X являются числами приближенными, то в отпечатанных значениях C, S и R большинство десятичных знаков являются сомнительными. В Бейсике имеется функция округления ROUND (X, n), где X — арифметическое выражение, задающее округляемое значение, а n — арифметическое выражение, задающее уровень округления. В случае, когда значение n не является целым, его дробная часть автоматически отбрасывается. Округление с помощью функции ROUND выполняется по правилам:

а) при $n>0$ происходит округление до n-й цифры после десятичной точки.

Примеры: ROUND (5.326,2) =5.33
ROUND (2.17,1.5) =2.2

б) при $n=0$ — округление до ближнего целого.

Примеры: ROUND (−2.6, 0) = −3
ROUND (6.5,0) =6

в) при $n<0$ — округление до $(|n|+1)$-й цифры влево от десятичной точки.

Примеры: ROUND (2386, −1) =2390
ROUND (−84732, −3) = −85000

Так, если в приведенной выше программе строку 70 переписать в виде:

```
7Ø PRINT "C="; ROUND (C, 2), "S="; ROUND (S, 2),
       "R="; ROUND (R, 2)
```

то программа отпечатает округленные результаты:

S=16.19 S=69.78 R=8.73

Используя функцию ROUND (X, n) и какую-либо из методик учета погрешностей (см. главу 2), можно программным путем находить значение n для каждой вычисляемой величины X и производить округление до верной цифры. Однако такой подход сильно усложнит программу. В небольших по объему вычислениях при фиксированных значениях входных величин уровни округления вычисляемых значений могут быть определены «вручную» и заранее заложены в операторах PRINT.

Контрольные вопросы

1. Какова общая структура программы на языке Бейсик?
2. Каким оператором Бейсика обеспечивается ввод исходных данных с клавиатуры в ходе исполнения программы?
3. Как работает оператор вывода PRINT?
4. С помощью какого оператора в программу на языке Бейсик включаются пояснения, комментарии?
5. Как с помощью оператора GOTO организуется неоднократная работа одной и той же программы?
6. Каким образом с помощью операторов DATA и READ организуется запись и считывание исходных данных в тексте программы?

Упражнения

1. Составить программу вычисления площади поверхности S и объема V конуса, заданного диаметром основания D и длиной образующей L:

а) с использованием оператора INPUT;

б) с использованием операторов DATA и READ (для трех пар произвольно выбранных значений D и L).

2. Радиус окружности, вписанной в равносторонний треугольник, равен R. Составить программу вычисления стороны, высоты и площади треугольника.

3. Сосуд имеет форму перевернутого конуса, осевое сечение которого — равносторонний треугольник. В сосуд брошен железный шар радиуса R и налита вода так, что поверхность воды касается погруженного в сосуд шара. Составить программу вычисления уровня воды после того, как шар будет вынут.

В программе использовать функцию округления ROUND, причем уровни округления выходных данных заложить вместе с входными значениями радиуса R в операторе DATA.

4.4 Программирование ветвлений

Для программирования алгоритмов, содержащих ветвления, в языке Бейсик имеется *оператор условного перехода* (или оператор IF), имеющий вид:

IF <условие> THEN <номер строки>

Действует этот оператор так: если <условие> при текущих значениях входящих в него переменных истинно, то происходит переход к строке, номер которой указан после THEN; в противном случае происходит переход к оператору программы, расположенному вслед за оператором IF.

Пример записи фрагмента программы с оператором условного перехода:

```
2Ø IF X<Ø THEN 5Ø
3Ø PRINT Y
```

Пусть в момент выполнения оператора IF текущее значение переменной $X=-3$. Тогда условие $X<Ø$ будет истинно и произойдет переход к строке с номером 5Ø. Если же, например, $X=5$, то условие $X<Ø$ будет ложно и произойдет переход к строке с номером 3Ø. Прежде чем перейти к составлению программ, содержащих ветвления, рассмотрим общие приемы программирования средствами языка Бейсик базовой структуры РАЗВИЛКА (см. п. 3.3).

Полная условная конструкция. Программирование этой базовой структуры (см. рис. 16, *а*) рассмотрим подробно. Обозначим условный номер строки программы буквой ω. Составим строку программы с оператором IF, проверяющим выполнение условия P:

ω IF P THEN ...

Многоточие здесь означает, что нам пока неизвестен номер строки, к которой нужно переходить при выполнении условия P. В то же время на основании правил выполнения оператора IF нам известно, что в случае невыполнения условия P машина перейдет к оператору, следующему за оператором IF (т. е. к строке с номером ω+10, если в строке ω оператор IF будет единственным). А согласно схеме, изображенной на рис. 16, *а*, в этом случае (когда P ложно) должен выполняться блок S_2: вот эту логическую ветвь мы и станем продолжать:

```
ω IF P THEN ...
ω+10 [S2]
```

(для простоты предполагается, что содержимое блока S_2 умещается в одной строке). Согласно схеме базовой структуры РАЗВИЛКА после выполнения блока S_2 должен происходить переход к выходу; поставим оператор GOTO в той же строке ω+10:

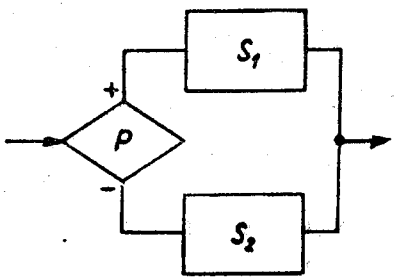
```
ω IF P THEN ...
ω+10 [S2] : GOTO ...
```

Многоточие после GOTO означает, что номер строки выхода из базовой структуры РАЗВИЛКА также пока неизвестен. Для

продолжения программы при условии, когда P истинно, может быть использована строка ω+20, что позволяет исключить многоточие в строке ω:

```
ω IF P THEN ω+20
ω+10 [S2] : GOTO ...
ω+20 [S1]
```

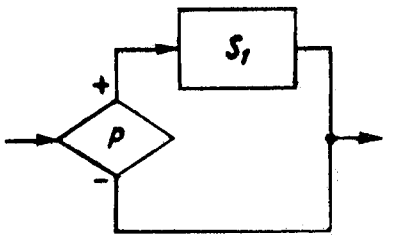
Теперь уже совершенно ясно, что выходом из базовой структуры РАЗВИЛКА будет строка ω+30, этим номером и следует дополнить оператор GOTO в строке ω+10. Ниже приведен фрагмент программы, реализующей полную условную конструкцию, слева — схема, справа — условное представление соответствующего программного фрагмента.




```
ω IF P THEN ω+20
ω+10 [S2] : GOTO ω+30
ω+20 [S1]
ω+30 выход
```

Разумеется, в конкретных случаях может оказаться, что для программного представления каждого из блоков S_1 и S_2 понадобится не по одной, а по нескольку строк программы. Это приведет к увеличению количества строк в программной реализации базовой структуры РАЗВИЛКА. Общая же логика построения программы останется неизменной.

Неполная условная конструкция. На схеме (см. рис. 16, *б*, с. 86) блок S должен выполняться в том случае, когда условие P истинно. Если P ложно, то должен происходить переход к выходу. В качестве условия оператора IF в данном случае целесообразно использовать не P, а его отрицание $\bar{P}$:




```
ω IF P̄ THEN ω+20
ω+10 [S1]
ω+20 выход
```

(читателю предоставляется возможность самостоятельно убедиться в том, что использование в операторе IF самого условия P привело бы к более громоздкой программе).

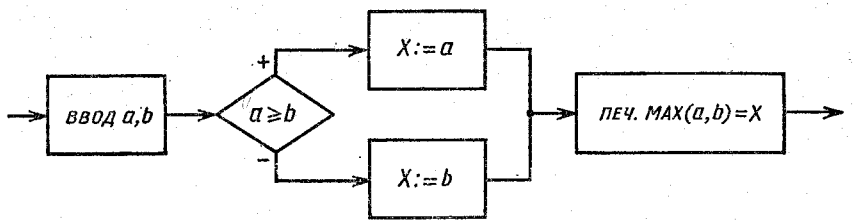


Рис. 36

Перейдем теперь к программированию алгоритмов, содержащих ветвления.

Пример 4.4.1. Составить программу поиска большего из двух чисел.

Схема алгоритма изображена на рисунке 36. Программа с использованием оператора ввода INPUT может иметь вид:

```
1Ø REM MAX (A, B)
2Ø INPUT A, B
3Ø IF A>B THEN 5Ø
4Ø X=B:GOTO 6Ø
5Ø X=A
6Ø PRINT "MAX ("; A; ", "; B; ") ="; X
7Ø GOTO 2Ø
```

Тело оператора PRINT организовано так, что результат будет отпечатан следующим образом: MAX (A, B) =X, где A, B — исходные числа, а X — найденный результат. Если задать программе значения $A=5$, $B=3$, ответ будет выдан в виде:

MAX (5, 3) =5

Пример 4.4.2. Составить программу решения уравнения $ax=b$ для произвольных значений числовых параметров a и b.

Схема алгоритма изображена на рисунке 21 (с. 88). Составим программу с использованием операторов DATA и READ для трех пар значений a и b, иллюстрирующих работу алгоритма в каждом из трех логически возможных случаев:

```
1Ø REM УРАВНЕНИЕ AX=B
2Ø READ A, B
3Ø IF A=Ø THEN 7Ø
4Ø LET X=B/A
5Ø PRINT "РЕШЕНИЕ X="; X
6Ø GOTO 11Ø
7Ø IF B=Ø THEN 1ØØ
8Ø PRINT "РЕШЕНИЙ НЕТ"
9Ø GOTO 11Ø
1ØØ PRINT "РЕШЕНИЙ БЕСК. МН."
11Ø GOTO 2Ø
12Ø DATA Ø, Ø, Ø, 4, 2, 3
13Ø END
```


Пример 4.4.3. Составить программу решения неравенства $ax>b$ (a и b — произвольные действительные числа).

Рассмотрев всевозможные сочетания значений параметров a и b, приходим к схеме алгоритма, изображенной на рисунке 37.

Программа, реализующая этот алгоритм, может иметь вид:

```
1Ø REM AX>B
2Ø INPUT A, B
3Ø PRINT "НЕРАВЕНСТВО"; A; " * X>"; B
4Ø IF A=Ø THEN 9Ø
5Ø C=B/A: PRINT "РЕШЕНИЕ"
6Ø IF A>Ø THEN 8Ø
7Ø PRINT "X<"; C:GOTO 12Ø
8Ø PRINT "X>"; C:GOTO 12Ø
9Ø IF B<Ø THEN 11Ø
1ØØ PRINT "РЕШЕНИЙ НЕТ":GOTO 12Ø
11Ø PRINT "X — ЛЮБОЕ ЧИСЛО"
12Ø GOTO 2Ø
```

Для иллюстрации работы программы на ввод необходимо подать пары значений a и b, соответствующие каждому из четырех

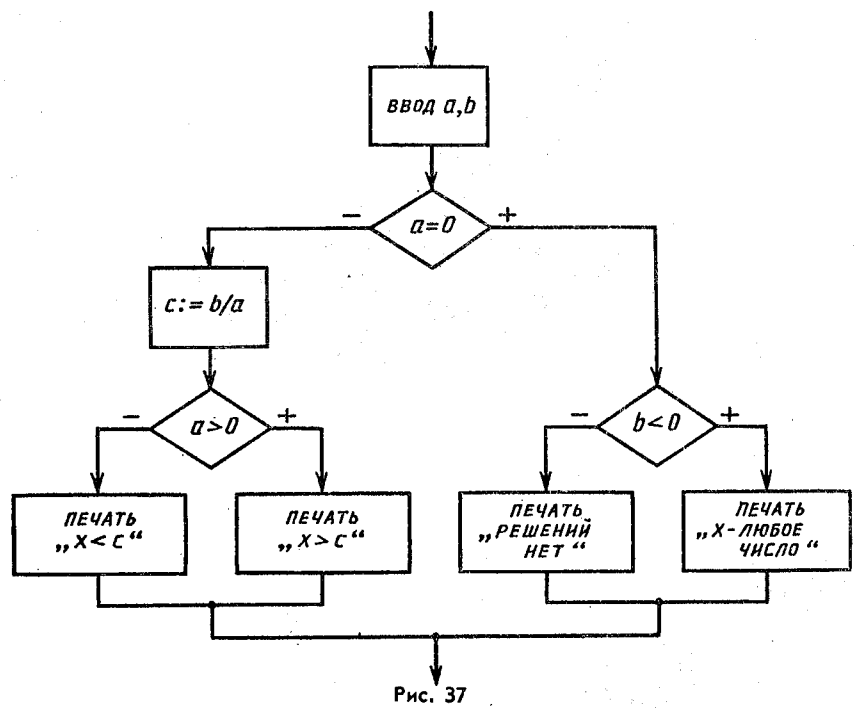


Рис. 37

возможных исходов, например: 0 и -5 (x — любое число), 0 и 3 (решений нет), 2 и 6 ($x>3$), -2 и 6 ($x<-3$). Выдача результата в программе организована таким образом, что каждый раз сначала выводится само решаемое неравенство. Например, после ввода значений 0 и -5 программа выдаст текст:

НЕРАВЕНСТВО Ø * X > −5
X — ЛЮБОЕ ЧИСЛО

Контрольные вопросы

1. Какова структура оператора условного перехода в языке Бейсик? Как действует этот оператор?
2. Как программируется на языке Бейсик базовая структура РАЗВИЛКА в виде полной условной конструкции?
3. Как программируется на языке Бейсик базовая структура РАЗВИЛКА в виде неполной условной конструкции?

Упражнения

1. Составить программы вычисления значений функции:

а) $f(z)=\begin{cases} z^2, & \text{если } z\geqslant 1,\\ 1-z, & \text{если } z<1; \end{cases}$

б) $\varphi(x)=\begin{cases} 0 & \text{для целого } x,\\ \operatorname{ctg}\pi\cdot x & \text{для нецелого } x. \end{cases}$

2. Составить программу поиска большего из трех чисел (см. схему алгоритма на рисунке 20, с. 88).

3. Составить программы решения квадратного уравнения $ax^2+bx+c=0\,(a\neq 0)$ в области действительных и в области комплексных чисел.

4. Составить программу решения неравенства $ax<3$.

5. Составить программы вычисления значений функций:

а) $f(x)=\begin{cases} \dfrac{\sin x}{1-x^3}, & \text{если } x<-1,\\ \arccos x^2, & \text{если } -1\leqslant x\leqslant 1,\\ \ln(x+0{,}8), & \text{если } x>1; \end{cases}$

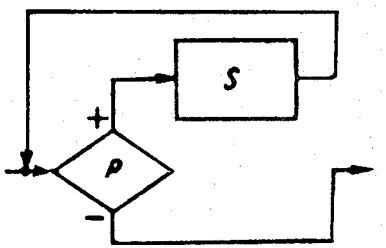
б) $\varphi(u)=\begin{cases} 2u+1, & \text{если } u\leqslant 0,\\ \cos u, & \text{если } u>0, \end{cases}$ где

$u=\begin{cases} \ln(-x), & \text{если } x<0,\\ \sqrt{x}, & \text{если } x\geqslant 0. \end{cases}$

4.5. Программирование циклов

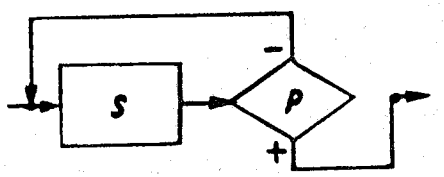
Для составления циклических программ используется базовая структура ЦИКЛ (см. п. 3.3). Программные аналоги этой структуры легко составляются с использованием оператора IF. Ниже приведены схемы программ на языке Бейсик для базовых структур ЦИКЛ-ПОКА и ЦИКЛ-ДО (для простоты предполагается, что программное описание тела S умещается в одной строке).

Базовая структура ЦИКЛ-ПОКА:



ω IF $\bar{P}$ THEN ω+20
ω+10 S :GOTO ω
ω+20 выход

Базовая структура ЦИКЛ-ДО:



ω S
ω+10 IF $\bar{P}$ THEN ω
ω+20 выход

Замена в записи оператора IF условия P его отрицанием, как и при программировании базовой структуры РАЗВИЛКА, позволяет упростить программу. Рассмотрим соответствующие примеры.

Пример 4.5.1. Составить программу нахождения наибольшего общего делителя (НОД) двух целых положительных чисел.

Схема алгоритма (метод вычитания) изображена на рисунке 24 (с. 90). Замечаем, что алгоритм представляет собой структуру ЦИКЛ-ПОКА. Программа имеет вид:

```
1Ø REM НОД (A, B)
2Ø INPUT A, B
3Ø X=A:Y=B
4Ø IF X=Y THEN 8Ø
5Ø IF X>Y THEN 7Ø
6Ø Y=Y-X:GOTO 4Ø
7Ø X=X-Y:GOTO 4Ø
8Ø PRINT "НОД ("; A; ", "; B; ")="; X
9Ø GOTO 2Ø
```

Печать результата имеет форму НОД (A, B) = X, где A и B — исходные числовые значения, а X — найденный числовой результат. Так, например, если программе задать значения $A=12$ и $B=18$, то ответ будет выдан в виде:

НОД (12, 18) = 6.

Чтобы сохранить для печати исходные значения данных A и B, в начале программы (строка 3Ø) эти значения передаются рабочим переменным X и Y.

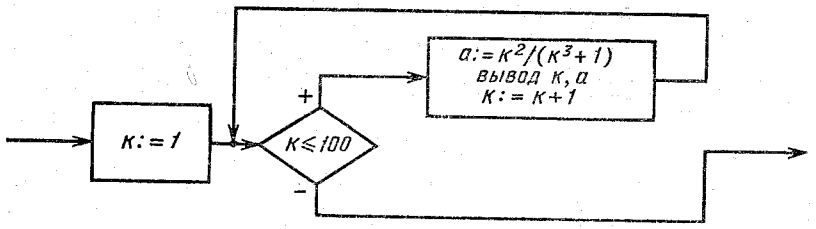


Рис. 38

Пример 4.5.2. Составить программу вычисления (генерирования) 100 членов последовательности, заданной формулой:

$$a_k=\frac{k^2}{k^3+1}\ (k=1,\ 2,\ \ldots,\ 100).$$

Схема алгоритма, использующего структуру ЦИКЛ-ПОКА, изображена на рисунке 38. Соответствующая программа имеет вид:

```
1Ø REM A=K ↑2/(K ↑3+1)
2Ø REM ЦИКЛ-ПОКА
3Ø K=1
4Ø IF K>1ØØ THEN 9Ø
5Ø A=K ↑2/(K ↑3+1)
6Ø PRINT K; A
7Ø K=K+1
8Ø GOTO 4Ø
9Ø END
```

Каждый член последовательности выводится на печать вместе со своим номером (строка 6Ø). Печать станет еще нагляднее и естественней, если оператор PRINT организовать так:

```
6Ø PRINT "A ("; K; ")="; A
```

Не следует думать, что логика построения циклической программы каждый раз жестко определяется самим программируемым алгоритмом. Очень часто один и тот же алгоритм можно запрограммировать как с помощью базовой структуры ЦИКЛ-ПОКА, так и с помощью базовой структуры ЦИКЛ-ДО. Речь может идти лишь об использовании адекватного и наиболее целесообразного для данного алгоритма способа программирования. Впрочем, часто выбор того или иного способа программной реализации связывается только со вкусами или привычками автора программы. Рассмотренная выше программа вычисления 100 членов последовательности (пример 4.5.2) может быть составлена по схеме ЦИКЛ-ДО. Действительно, схему этого алгоритма можно составить и так, как показано на рисунке 39. Соответствующая программа будет иметь вид:


```
1Ø REM A=K ↑2/(K ↑3+1)
2Ø REM ЦИКЛ-ДО
3Ø K=1
4Ø A=K ↑2/(K ↑3+1)
5Ø PRINT "A ("; K; ")="; A
6Ø K=K+1
7Ø IF K<=1ØØ THEN 4Ø
8Ø END
```

С точки зрения задачи, для решения которой составлены две последние программы, они равносильны, так как дают одинаковый результат. Однако надо понимать, что такие программы, составленные на основе структур ЦИКЛ-ПОКА и ЦИКЛ-ДО, все-таки различны. Это обусловлено различием самих базовых структур. Как известно, блок S в структуре ЦИКЛ-ДО всегда выполняется по меньшей мере один раз, независимо от истинности условия P, в то время как в структуре ЦИКЛ-ПОКА при ложности P блок S не выполнится ни разу. Если допустить, что в задачах, подобных рассмотренной выше, верхняя граница параметра цикл K заранее не известна и определяется, например, в процессе предыдущего счета, то может оказаться, что вторая программа (ЦИКЛ-ДО) в этой ситуации даст нелепый результат. Действительно, если граница значения параметра K окажется неположительной, то последовательность становится неопределенной. Вторая программа в этом случае все-таки выведет результат в виде значения первого члена (при $k=1$). Возможность подобных ситуаций требует особого внимания при составлении циклических программ, суммирующих последовательности.

Пример 4.5.3. Составить программу вычисления суммы n членов последовательности

$$a_k=\frac{k}{k^2+1}.$$

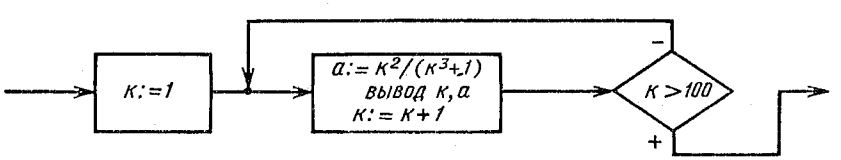


Рис. 39

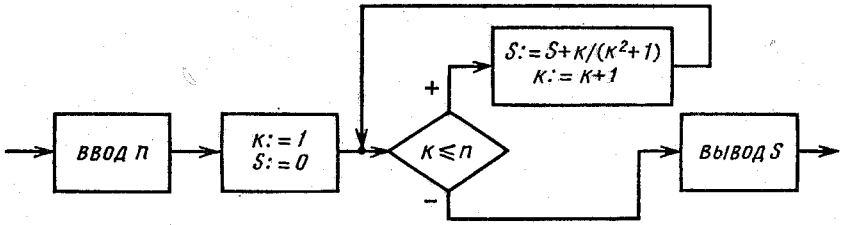


Рис. 40

Программу составим так, чтобы число n определялось путем ввода.

В основу алгоритма положим базовую структуру ЦИКЛ-ПОКА (рис. 40). Соответствующая программа будет иметь вид:

```
1Ø REM СУММА
2Ø INPUT N
3Ø K=1:S=Ø
4Ø IF K>N THEN 7Ø
5Ø S=S+K/(K↑2+1)
6Ø K=K+1:GOTO 4Ø
7Ø PRINT "СУММА S="; S
8Ø END
```

Иной принцип организации имеют циклические программы суммирования числовых рядов не по заданному количеству членов, а в зависимости от значения самих членов.

Пример 4.5.4. Составить программу вычисления суммы всех членов ряда

$$\sum_{k=1}^{\infty} \frac{k}{x^k},$$

не меньших по абсолютной величине заданного числа ε.

Значение первого члена ряда равно $1/x$. Значение каждого последующего члена также может быть вычислено по формуле $a=k/x^k$ (при $k=2, 3, \ldots$). Однако добавлять к искомой сумме очередной член ряда следует лишь после проверки условия $|a|\geqslant\varepsilon$. Схема алгоритма, основанного на базовой структуре ЦИКЛ-ПОКА, изображена на рисунке 41. Пользуясь этой схемой, легко составить программу на языке Бейсик:

```
1Ø REM СУММА РЯДА
2Ø INPUT E, X
3Ø K=1:S=Ø:A=1/X
4Ø IF ABS (A)<E THEN 7Ø
5Ø S=S+A:K=K+1
6Ø A=K/X↑K:GOTO 4Ø
7Ø PRINT"S="; S
8Ø END
```

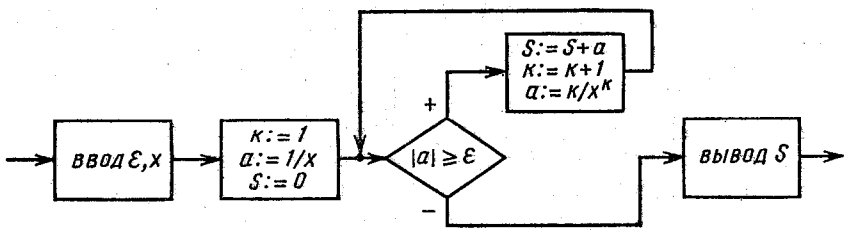



Рис. 41

При составлении программы решения данной задачи использовать базовую структуру ЦИКЛ-ДО оказалось бы ошибочным, так как при определенном сочетании задаваемых программе значений ε и x значение результата S может оказаться равным 0, в то время как программа ЦИКЛ-ДО и в этом случае выдаст в качестве результата значение первого члена $1/x$.

Рассмотрим особо случай, когда условием P в базовой структуре ЦИКЛ-ДО (см. рис. 17, *б*) является неравенство вида $\alpha>\beta$, где параметр цикла α имеет своим начальным значением α_0, а в каждом обороте цикла преобразуется по закону арифметической прогрессии $\alpha:=\alpha+h$ (h — заданный шаг). Соответствующий циклический алгоритм описывается схемой, изображенной на рисунке 42. Для программирования циклических алгоритмов такого вида в языке Бейсик имеются два специальных оператора: *оператор начала цикла* (или оператор FOR) и *оператор конца цикла* (оператор NEXT). Операторы FOR и NEXT всегда используются совместно.

Оператор начала цикла (его называют еще *заголовком цикла*) имеет вид:

FOR <прост. перем.>=<ар. выраж.> TO <ар. выраж.>
STEP<ар. выраж.>

Вслед за оператором начала цикла располагают строки программы, которые должны выполняться циклически (эти строки образуют *тело цикла*). За последней строкой тела цикла размещается строка со специальным оператором конца цикла, который имеет вид:

NEXT <простая переменная>

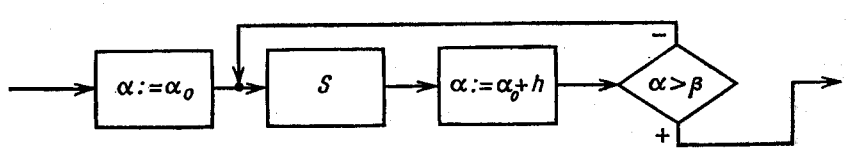


Рис. 42

Переменная, указываемая после NEXT, должна быть той же (простой) переменной, что и после FOR (эту переменную называют *параметром цикла*). Выражения, стоящие слева и справа от TO, задают соответственно начальное и конечное значения параметру цикла. Выражение, стоящее после слова STEP, задает шаг (число), на величину которого изменяется параметр цикла после каждого повторения тела цикла. Шаг может быть как положительным, так и отрицательным.

Условная программная запись алгоритма, изображенного на рисунке 41 с помощью операторов FOR и NEXT, может иметь вид:

ω FOR $\alpha=\alpha_0$ TO β STEP h
$\omega+10$ [S]
$\omega+20$ NEXT α

Выполнение этой программы происходит следующим образом. Параметру цикла α присваивается его начальное значение α_0 и один раз выполняется тело цикла S (т. е. группа операторов, размещенных между заголовком цикла и соответствующим оператором NEXT). Вслед за этим оператор NEXT увеличивает значение параметра цикла α на величину шага h и проверяет, не превосходит ли новое значение параметра цикла значения арифметического выражения β, расположенного за словом TO в заголовке цикла. Если этого не происходит, то осуществляется повторное выполнение тела цикла, в противном случае — выход из цикла, т. е. переход к оператору, следующему за NEXT. Если шаг цикла равен $+1$, указание STEP 1 в операторе начала цикла может опускаться. Операторы FOR и NEXT могут быть записаны в строках с несколькими операторами при условии, что FOR является первым оператором в своей строке, а NEXT — последним оператором в своей строке*.

Легко видеть, что все алгоритмы, программируемые с помощью операторов FOR-NEXT, могут быть запрограммированы и другими средствами языка Бейсик. Например, программная реализация алгоритма средствами FOR-NEXT, приведенная выше, может быть заменена равносильной записью с использованием оператора IF:

ω $\alpha=\alpha_0$
$\omega+10$ [S] : $\alpha=\alpha+h$
$\omega+20$ IF $\alpha<=\beta$ THEN $\omega+10$
$\omega+30$ выход

*Последнее замечание не относится к случаю вложенных циклов (см. дальше), когда в одной строке может размещаться несколько операторов NEXT.

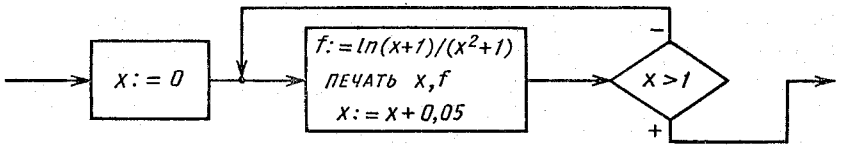


Рис. 43

Приведенная выше запись может рассматриваться как определение операторов цикла FOR-NEXT. Тем не менее использование операторов цикла (там, где это возможно) позволяет более лаконично описывать циклические алгоритмы по схеме ЦИКЛ-ДО. Так, например, программа генерирования членов последовательности с общим членом $a_k=\frac{k^2}{k^3+1}$ (см. схему на рисунке 39) может быть составлена и таким образом:

```
1Ø REM A=K ↑2/(K ↑3+1)
2Ø REM FOR-NEXT
3Ø FOR K=1 TO 1ØØ
4Ø A=K ↑2/(K ↑3+1)
5Ø PRINT "A ("; K; ") ="; A
6Ø NEXT K
```

Здесь тело цикла составляют строки 4Ø и 5Ø, параметром цикла является переменная K (выражение STEP1 в заголовке цикла опущено).

Близкими по характеру к алгоритмам генерирования членов числовой последовательности являются алгоритмы табулирования функций.

Пример 4.5.5. Составить программу табулирования функции

$$f(x)=\frac{\ln(x+1)}{x^2+1}$$

на отрезке [0; 1] с шагом 0,05.

Схема алгоритма табулирования (ЦИКЛ-ДО) изображена на рисунке 43. Программа на языке Бейсик:

```
1Ø REM ТАБ
2Ø PRINT "X", "F"
3Ø FOR X=Ø TO 1 STEP.Ø5
4Ø F=LOG (X+1) / (X ↑2+1)
5Ø PRINT X, F
6Ø NEXT X
```

Программа будет выводить две колонки числовых значений: слева — значения аргумента X, справа — значения функции F. Сравнивая эту программу с программой, составленной в примере 1.5.3 (п. 1.5, с. 43), можно сделать вывод о том, насколько

эффективнее решение программируемых задач на микроЭВМ, обладающей языком высокого уровня, по сравнению даже с достаточно хорошим программируемым микрокалькулятором.

Операторы цикла FOR-NEXT могут эффективно использоваться при составлении циклических программ, работающих с массивами. В языке Бейсик разрешено использовать только одномерные и двухмерные массивы. При этом они должны быть обязательно описаны в программе. Описание массива осуществляется с помощью специального оператора-описателя DIM. По этому описанию ЭВМ резервирует в памяти необходимое количество мест для элементов массива. Оператор DIM размещается в программе раньше, чем начинается работа с соответствующим массивом. Описание ведется так: вслед за словом DIM помещается имя массива, а за ним (в круглых скобках) верхние границы индексов — для двухмерных массивов через запятую. Так, массив a_1, a_2, .., a_{100} будет описан следующим образом:

DIM A (1ØØ)

Максимальным индексом в языке Бейсик является число 7999. Одним оператором DIM можно описать несколько массивов, в этом случае описания различных массивов располагаются в строке оператора DIM через запятую. Так, например, описание

DIM T (5Ø), X (4), I (14, 26)

резервируют в памяти места для двух одномерных и одного двухмерного массива. Как и прочие строки программы, строки описаний получают свои порядковые номера.

Пример 4.5.6. Дан одномерный массив x_1, x_2, ..., x_{14}. Составить программу вычисления суммы $S=x_1+x_2+...+x_{14}$.

Ввод элементов массива в память ЭВМ, как и сам процесс вычислений суммы S, носит циклический характер. В целях сокращения программы эти два цикла объединены в один: и ввод очередного элемента x_k, и добавление его к искомой сумме S входят в тело одного и того же цикла (см. схему на рисунке 44). С помощью операторов FOR-NEXT этот алгоритм программируется следующим образом:

```
1Ø REM СУММА МАССИВА
2Ø DIM X (14)
3Ø S=Ø
4Ø FOR K=1 TO 14
5Ø READ X (K)
6Ø S=S+X (K)
7Ø NEXT K
8Ø PRINT "СУММА S="; S
9Ø DATA 1, 2, 3, 4, 5, 6, 7
1ØØ DATA 8, 9, 1Ø, 11, 12, 13, 14
```

В качестве конкретных значений в списки операторов DATA включены натуральные числа от 1 до 14. С помощью операторов

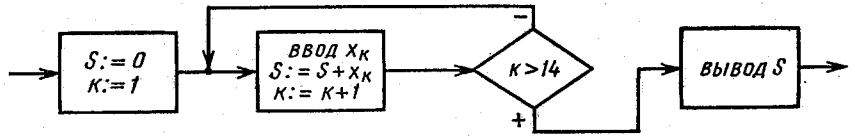


Рис. 44

FOR-NEXT особенно эффективно программируются алгоритмы с вложенными циклами.

Пример 4.5.7. Составить программу вычисления и печати элементов таблицы Пифагора.

Таблица Пифагора — это квадратная матрица из n строк и n столбцов. Первый столбец и первая строка состоят из натуральных чисел 1, 2, ..., n, так что левый угловой элемент a_{11} равен 1. Каждый элемент a_{ij} определяется формулой:

$$a_{ij}=i\times j.$$

Алгоритм можно построить следующим образом: сформировать искомый двухмерный массив A и напечатать его в форме квадратной матрицы (схема приведена на рисунке 45). Алгоритм содержит два цикла: внутренний цикл, определяющий элементы строки (при этом значение j остается постоянным, а i изменяется от 1 до n), и внешний цикл, осуществляющий переход от строки к строке (j изменяется от 1 до n). Искомая программа будет содержать два вложенных цикла FOR-NEXT, причем параметром внутреннего цикла служит переменная i, а внешнего — переменная j:

```
1Ø REM ТАБЛИЦА ПИФАГОРА
2Ø DIM A (1Ø, 1Ø)
3Ø FOR J=1 TO 1Ø
4Ø FOR I=1 TO 1Ø
5Ø A (I, J) = I * J
6Ø NEXT I
7Ø NEXT J
8Ø MATPRINT A
9Ø END
```

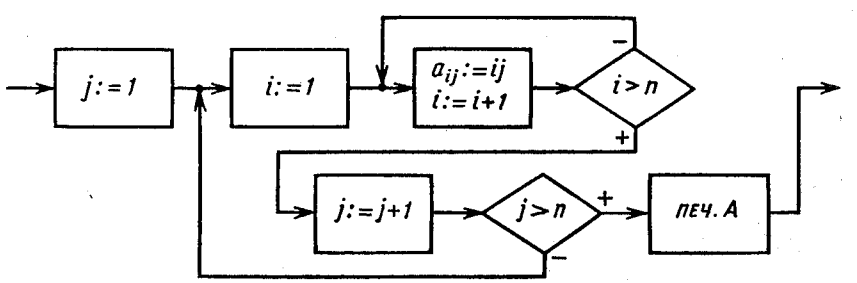


Рис. 45

Приведенная программа составлена для $n=10$ (10 строк и 10 столбцов). Строки 4Ø, 5Ø, 6Ø образуют тело внешнего цикла, которое выполняется 10 раз (J=1, 2, ..., 10). При этом строка 5Ø, являясь телом внутреннего цикла, выполняется 10 раз (I=1, 2, ..., 10) для каждого значения J. Таким образом, строка 5Ø в общей сложности выполняется $10\times10=100$ раз.

Печать результата осуществляется специальным оператором MATPRINT, который построчно выводит двухмерный массив (матрицу) A в виде:

1	2	3	4	5	6	7	8	9	1Ø
2	4	6	8	1Ø	12	14	16	18	2Ø
3	6	9	12	15	18	21	24	27	3Ø
4	8	12	16	2Ø	24	28	32	36	4Ø
5	10	15	20	25	30	35	40	45	5Ø
6	12	18	24	3Ø	36	42	48	54	6Ø
7	14	21	28	35	42	49	56	63	7Ø
8	16	24	32	4Ø	48	56	64	72	8Ø
9	18	27	36	45	54	63	72	81	9Ø
1Ø	2Ø	3Ø	4Ø	5Ø	6Ø	7Ø	8Ø	9Ø	1ØØ

Контрольные вопросы

1. Как программируются базовые структуры ЦИКЛ-ПОКА и ЦИКЛ-ДО в языке Бейсик с помощью оператора IF (все случаи)?
2. В чем заключаются особенности циклических программ, составленных на основе алгоритмических базовых структур ЦИКЛ-ПОКА и ЦИКЛ-ДО, и как это следует учитывать в программировании?
3. Как записываются и как действуют операторы цикла FOR-NEXT? Какая из структур циклических алгоритмов реализуется этими операторами?
4. Как составляются описания массивов в программах на языке Бейсик?

Упражнения

1. Составить программу генерирования 50 членов последовательности, заданной формулой общего члена $b_k=\frac{\sqrt{k}}{k+0{,}5}$:

а) с использованием оператора IF (по схемам ЦИКЛ-ПОКА и ЦИКЛ-ДО);

б) с использованием операторов FOR-NEXT.

2. Составить программы табулирования функций:

а) $f(x)=\frac{2\sin^3 x}{1+x^2}$ на отрезке $[-\pi, \pi]$ с шагом 0,1;

б) $f(x)=\begin{cases}\sqrt{2x^2+1}, & |x|<4\\ \ln(x-0{,}6), & |x|\geqslant 4\end{cases}$ для $x=-10, -9, \ldots, 10$.

3. Дан ряд:

$$\sum_{n=1}^{\infty}\frac{\sqrt{n}}{n^3+0{,}8}=\frac{1}{1{,}8}+\frac{\sqrt{2}}{8{,}8}+\ldots.$$

Составить программы:

а) вычисления суммы первых N членов ряда;

б) вычисления суммы всех первых членов ряда, величина которых не меньше заданного числа ε (значения параметров N и ε определяются при вводе).

4. Составить программу вычисления суммы:

$$\sum_{n=1}^{50}\frac{n+1}{n!}.$$

Примечание. Использовать рекуррентную формулу, связывающую два соседних члена ряда.

5. Составить программы поиска максимального элемента:

а) в одномерном массиве (см. схему на рисунке 27, с. 91);

б) в двухмерном массиве.

6. Составить программу вычисления значений функции $f(x, y)=\sqrt{x^2+y^2}$ для всевозможных значений x и y из последовательностей: $x_1, x_2, \ldots, x_m$ и $y_1, y_2, \ldots, y_n$.

4.6. Загрузка, редактирование и пуск программы

Рассмотрим перечень действий, которые должен проделать пользователь ЭВМ «Искра-226» для ввода и пуска программы.

После включения машины и загрузки необходимого математического обеспечения на экране дисплея фиксируется конфигурация:

```
READ Y
: _
```

Это означает, что ЭВМ не производит никаких операций и готова к приему информации. Символ _ (курсор) всегда показывает место экрана, на котором будет отображен очередной вводимый с клавишного устройства символ.

Для загрузки текста программы используется клавишное устройство, обеспечивающее возможность ввода любого символа алфавита языка Бейсик. Приведем краткое описание состава

клавиатуры и назначения отдельных клавиш. На клавишном устройстве условно выделяются 8 функциональных зон (рис. 35, с. 99). Клавиатура первой зоны работает в трехрегистровом режиме. Нижний регистр содержит операторы языка Бейсик. Верхний регистр зоны содержит прописные русские и латинские буквы (в зависимости от положения переключателя «лат/рус») и специальные символы. Переход на верхний и нижний регистры осуществляется соответственно клавишами [SHIFT] и [SHIFT LOCK], причем при переходе на нижний регистр загорается индикаторная лампочка. После нажатия клавиши [SHIFT LOCK] клавиатура первой зоны фиксируется в положении нижнего регистра, и эта фиксация снимается нажатием клавиши [SHIFT]. При переходе на верхний регистр нужно нажать клавишу [SHIFT] и, не отпуская ее, нажать требуемую клавишу.

Вторая зона, имеющая один регистр, содержит клавиши управления счетом и клавиши редактирования при вводе. Третья зона содержит стандартную цифровую клавиатуру для набора десятичных цифр и десятичной точки. Четвертая зона имеет один регистр и содержит часто используемые операторы языка. Пятая зона, имеющая два регистра, включает арифметические операции и стандартные математические функции. Шестая зона, имеющая один регистр, содержит клавиши редактирования текста программы. Седьмая зона имеет два регистра и 16 клавиш. Восьмая зона, имеющая один регистр, содержит шесть клавиш управления курсором.

Отображение вводимой с клавиатуры в память ЭВМ информации на экране дисплея дает возможность контролировать процесс ввода, а также редактировать вводимую информацию. Ввод готовой информации с клавиатуры ведется построчно, начиная каждый раз с номера строки. Номер строки можно вводить нажатием соответствующих цифровых клавиш, но можно воспользоваться и специальной клавишей [STMT NUMBER] (*statement number* — номер инструкции). Первое нажатие этой клавиши вводит номер строки, равный 10. При каждом последующем нажатии вводится номер очередной строки, на 10 больший предыдущего. Именно по этой причине при составлении программы строки удобно нумеровать десятками (разумеется, любой другой номер при необходимости может быть введен с помощью обычных цифровых клавиш). После набора полного текста строки нажимают клавишу перехода на новую строку [CR / LF].

Пока не нажата клавиша [CR / LF], текст вводимой строки можно отредактировать, используя клавиши [BACK SPACE] и [LINE ERASE].

С помощью клавиши [BACK SPACE] можно осуществлять посимвольное стирание текста в движении справа налево. Например, ввели на экран очередную строку программы:

```
18Ø A=5+SQR (2+COS (1_
```

и надо ошибочно введенное имя функции COS исправить на SIN. Тогда пятикратным нажатием клавиши [BACK SPACE] поочередно стираются символы

```
1( SOC
```

После чего строка имеет вид:

```
18Ø A=5+SQR (2+_
```

Вслед за этим обычным образом вводится правильный текст.

Стирание всей строки до нажатия клавиши [CR / LF] можно осуществить с помощью клавиши [LINE ERASE]. После нажатия этой клавиши текст программы исчезает, а курсор становится в начальное положение строки, что позволяет вводить любой желаемый текст. Отредактированная строка нажатием клавиши [CR / LF] вводится в память ЭВМ. Затем нажимают клавишу [STMT NUMBER] — и происходит переход к очередной строке программы.

При необходимости редактирования программной строки, уже введенной в память, необходимо перевести эту строку в состояние редактирования EDIT. Для этого надо набрать на цифровой клавиатуре номер строки и нажать клавишу [EDIT] — около набранного номера строки слева появится звездочка. Вслед за этим нажимают клавишу [RECALL] — и строка вызывается на экран. Для редактирования в режиме EDIT используются клавиши:

[DELETE] — стирается символ над курсором, а часть строки справа от курсора сдвигается влево на один символ;

[ERASE] — стирается часть строки, расположенная справа от местоположения курсора;

[INSERT] — раздвигается строка для вставки символа.

Для управления положением курсора на поле дисплея ис-

пользуются шесть клавиш. Клавишами [→] и [←] смещают курсор вправо или влево по строке программы на одно знакоместо. Для ускоренного перемещения курсора по горизонтали используют аналогичные клавиши [→→] и [←←], смещающие курсор сразу на пять знакомест. Смещение по вертикали (переход на строку выше или ниже) достигается нажатием клавиш [↑] и [↓]. Клавиши редактирования и управления курсором действуют в пределах редактируемой строки (программная строка может занимать три строки экрана). Отредактированная строка нажатием клавиши [CR / LF] вводится в память ЭВМ. Для полной замены строки, находящейся в памяти ЭВМ на новую, следует ввести номер этой строки, новый текст и нажать клавишу [CR / LF]. Чтобы введенную строку стереть из памяти, набирают на клавиатуре ее номер и нажимают [CR / LF].

После того как вся программа введена в память машины и отредактирована, можно приступить к ее выполнению. Пуск программы со строки с наименьшим номером осуществляется с помощью оператора RUN, для чего последовательно нажимаются клавиши [RUN] (зона 4) и [CR / LF]. В общем случае программа может быть пущена с любой строки, а не только с первой по порядку номеров, т. е. оператор RUN может использоваться в виде:

RUN < номер строки >,

где <номер строки> является целым числом, указывающим на номер начальной строки при пуске программы.

Текст программы при желании можно вывести на печатающее устройство, для этого надо нажать клавиши:

[LIST] [/] [Ø] [C] [CR / LF].

Для вывода результатов на печатающее устройство нажимаются клавиши:

[SELECT] [PRINT] [Ø] [C] [CR / LF].

Переключение вывода на экран производится клавишами:

[SELECT] [PRINT] [Ø] [5] [CR / LF].

Для очистки памяти машины от хранящихся в ней программ и данных нажимаются клавиши:

[CLEAR] [CR / LF].

На передней панели клавишного устройства расположена кнопка [RESET], используемая для подачи команды аварийного останова. Нажатие кнопки [RESET] прекращает выполнение оператора, действия устройств ввода-вывода и передает управление пользователю. При нарушении каких-либо правил обращения с ЭВМ она выдает сообщение об ошибке в виде текста ERROR XX, где XX — номер ошибки, по которому в технической документации машины можно найти объяснение причин ошибки и рекомендации по ее устранению. В приложении даны некоторые сведения об ошибках.

Правила загрузки и редактирования информации для разных микроЭВМ имеют много общего, но вместе с тем различны в зависимости от конфигурации клавиатуры, особенностей операционной системы и т. п. Необходимые уточнения всегда можно найти в технической документации микроЭВМ.

Контрольные вопросы

1. Каков состав клавиатуры микроЭВМ «Искра-226» и назначение отдельных групп клавиш?
2. Какие действия производятся при загрузке строки программы?
3. Как редактируется текст строки до загрузки ее в память?
4. Как редактируется текст строк, введенных в память ЭВМ?
5. Каким образом текст введенной в память строки заменяется полностью? Как стирается из программы строка программы?
6. Как осуществляется пуск введенной и отредактированной программы?
7. С помощью каких операций текст программы и результаты ее работы выводятся на печатающее устройство?

Глава 5
ЛАБОРАТОРНЫЙ ПРАКТИКУМ

Наряду с лекциями и практическими (семинарскими) занятиями в лабораторной части курса предусматривается выполнение индивидуальных заданий в форме двух лабораторных работ. Цель лабораторных занятий — закрепление и отработка в условиях постоянного доступа к вычислительной технике (микрокалькуляторы, микроЭВМ) основного содержания учебного материала.

В настоящей главе описываются задания к лабораторным работам и краткие указания к их выполнению.

5.1. Лабораторная работа 1

Тема. Методы приближенных вычислений.

Задание 1. Число x, все цифры которого верны в строгом смысле, округлить до трех значащих цифр. Для полученного числа $x_1 \approx x$ вычислить границы абсолютной и относительной погрешностей. В записи числа x_1 указать количество верных цифр.

Задание 2. Вычислить значение величины z с помощью МК при заданных значениях параметров a, b и c тремя способами:

а) по правилам подсчета цифр;

б) с систематическим учетом границ абсолютных погрешностей;

в) по способу границ.

Используя для вычислений микрокалькулятор, в каждом из способов вычислений применить методику пооперационной и итоговой оценки точности вычислений. Сравнить полученные результаты между собой, прокомментировать различие методов вычислений и смысл полученных числовых значений.

Задание 3. Вычислить с помощью ЭВМ величину z при заданных значениях a, b и c двумя способами:

а) с машинным округлением результата по правилам подсчета цифр с применением функции ROUND;

б) с подсчетом границы абсолютной погрешности результата и последующим ручным округлением. И в том и в другом случае сопоставить полученные ответы с результатом выполнения п. б) задания 2.

Пояснения к выполнению лабораторной работы 1. Исходные данные для выполнения заданий содержатся в таблице 5.1. (числа x, a, b, c — приближенные, в их записи все цифры верны в строгом смысле, коэффициенты — точные числа).

Таблица 5.1

№	x	z	a	b	c
1	3549	$\frac{c+\cos b}{2c-a^2}$	0,317	3,27	4,7561
2	7,32147	$\frac{\ln(b+c)}{b-ac}$	0,0399	4,83	0,072
3	35,085	$\frac{\sqrt{a+b}}{3a-c}$	1,574	1,40	1,1236
4	7,544	$\frac{ab-4c}{\ln a+3b}$	12,72	0,34	0,0290
5	198,745	$\frac{a-\operatorname{tg} b}{13c+b}$	3,49	0,845	0,0037
6	37,4781	$\frac{ac+3b}{\sqrt{b-c}}$	0,0976	2,371	1,15874
7	0,183814	$\frac{\ln(a-b)}{\sqrt{b+c}}$	82,3574	34,12	7,00493
8	0,009145	$\frac{b+\cos c}{3b+2a}$	0,11587	4,256	3,00971
9	11,3721	$\frac{a^2-b}{\sqrt{ab+c}}$	3,71452	3,03	0,765
10	0,2538	$\frac{\ln a+4b}{ab-c}$	7,345	0,31	0,09872
11	10,2118	$\frac{b^2+\ln c}{\sqrt{c-a}}$	0,038	3,9353	5,75
12	4,394	$\frac{2\operatorname{tg}(a+b)}{a^2c+b}$	0,2471	0,0948	37,84
13	0,8437	$\frac{4\sqrt{a^2+c}}{ab-c}$	1,284	4,009	3,2175
14	129,66	$\frac{\sin(a-\sqrt{b})}{c+a\ln b}$	18,407	149,12	2,3078
15	48,847	$\frac{0{,}8\ln b}{\sqrt{a+c}}$	29,49	87,878	4,403
16	9,2039	$\frac{\sqrt{a}}{bc-\ln b}$	74,079	5,3091	6,234
17	2,3143	$\frac{\sqrt{ab}}{b-2c}$	3,4	6,22	0,149

Продолжение

№	x	z	a	b	c
18	0,012147	$\frac{(b-c)^2}{2a+b}$	4,05	6,723	0,03254
19	0,86138	$\frac{\ln b-a}{a^2-12c}$	0,7219	135,347	0,013
20	0,1385	$\frac{b-\sin a}{a+3c}$	3,672	4,63	0,0278
21	23,394	$\frac{10c+\sqrt{b}}{a^2-b}$	1,24734	0,346	0,051
22	0,003775	$\frac{(a-c)^2}{\sqrt{a}+3b}$	11,7	0,0937	5,081
23	718,54	$\frac{a-\sin b}{b^2+6c}$	1,75	1,21	0,041
24	9,73491	$\frac{\sqrt{b-c}}{\ln a+b}$	18,0354	3,7251	0,071
25	11,456	$\frac{\ln c-10a}{\sqrt{bc}}$	0,113	0,1056	89,4

Для выполнения **заданий 1** и **2** необходимо проработать материал глав 1 и 2, выполняя с помощью МК приведенные в тексте примеры и упражнения в конце параграфов.

Для выполнения **задания 3** составляются две простые программы на ЭВМ.

В первом случае с учетом точности заданных значений a, b и c по правилам подсчета цифр делается прикидка точности результата и устанавливается уровень его округления n, который затем используется как второй аргумент функции ROUND. Пусть, к примеру, требуется вычислить значение выражения

$$z=\frac{\sqrt{x}}{\ln(y-t^2)}$$

для значений $x=3.91$, $y=4.08$, $t=0.826$, у которых все цифры верны в строгом смысле. Замечаем, что при этих условиях по правилам подсчета цифр значение аргумента логарифма, а следовательно и значение самого логарифма, будет иметь не менее трех значащих цифр. Такое же количество значащих цифр будет иметь и числитель. Отсюда следует, что и сам результат будет иметь не менее трех верных значащих цифр. Кроме этого, прикидка показывает, что целая часть результата будет цифрой однозначной. Отсюда следует, что по правилам подсчета цифр искомый результат z должен быть округлен до сотых. Программа:


```
1Ø REM Z=SQR (X)/LOG (Y—T ↑2)
2Ø REM ПРАВИЛА ПОДСЧЕТА ЦИФР
3Ø READ X, Y, T
4Ø Z=SQR (X)/LOG (Y—T ↑2)
5Ø PRINT "Z="; ROUND (Z, 2)
6Ø DATA 3.91, 4.Ø8, Ø.826
7Ø END
   RUN
   Z=1.62
```

Во втором случае составляется программа с вычислением и печатью z и Δz. Для вычисления значения Δz в программе используется методика итоговой оценки границы абсолютной погрешности результата.

Составим требуемую программу для вычисления значения выражения $z=\frac{\sqrt{x}}{\ln(y-t^2)}$ при тех же значениях исходных данных x, y, t. Применяя последовательно формулы для подсчета границ абсолютных погрешностей арифметических действий (см. Приложение II) и элементарных функций (табл. 2. 1, с. 65), получим:

$$\Delta z=\frac{\sqrt{x}\cdot\Delta\ln(y-t^2)+\ln(y-t^2)\cdot\Delta(\sqrt{x})}{\ln^2(y-t^2)}=$$
$$=\frac{2x(\Delta y+2t\,\Delta t)+\Delta x(y-t^2)\ln(y-t^2)}{2\sqrt{x}(y-t^2)\ln^2(y-t^2)}.$$

Ниже приведена программа на языке Бейсик с выведенными на печать значениями z и Δz (в программе погрешности значений X, Y, T и Z обозначены соответственно через XØ, YØ, TØ, ZØ, а значения XØ, YØ, TØ определены из условия, что X, Y, T заданы цифрами, верными в строгом смысле):

```
1Ø REM Z=SQR (X)/LOG (Y—T ↑2)
2Ø СТРОГИЙ УЧЕТ ПОГРЕШНОСТИ
3Ø READ X, XØ, Y, YØ, T, TØ
4Ø A=SQR (X):B=Y—T ↑2:C=LOG (B)
5Ø Z=A/C
6Ø ZØ=(2*X*(YØ+2*T*TØ)+XØ*B*C)/(2*A*B*C↑2)
7Ø PRINT "Z="; Z
8Ø PRINT "ZØ="; ZØ
9Ø DATA 3.91, .ØØ5, 4.Ø8, .ØØ5, .826, ØØØ5
1ØØ END
   RUN
   Z=1.616681Ø75569
   ZØ=3.3ØØ1156E — Ø3
```

Судя по величине погрешности, z имеет две верные в строгом смысле цифры. После округления получаем:

$$z=1{,}62\pm0{,}01.$$

5.2. Лабораторная работа 2

Тема. Решение задач на ПМК и микроЭВМ.

Задание 1. Для заданной функции $f(x)$ методом табулирования локализовать корни уравнения $f(x)=0$ (т. е. выделить по возможности наименьшие отрезки, содержащие по одному корню) и найти приближенные значения корней. Если корней бесконечное множество — взять два-три корня, ближайших к началу координат. Построить график функции на исследуемом участке.

Задание 2. Последовательность задана формулой общего члена a_k. Вычислить: а) значения первых n членов последовательности; б) сумму n первых членов последовательности; в) сумму всех членов последовательности, не меньших заданного числа ε.

Задание 3. Методом табулирования исследовать заданный предел функции (в точке или на бесконечности): а) установить предположительное значение предела (все заданные пределы функций существуют); б) исследовать поведение функции в окрестности точки (или, в зависимости от задания, на $+\infty$ и $-\infty$); в) сделать вывод о скорости и характере сходимости.

Пояснения к выполнению лабораторной работы 2.

Задания к выполнению лабораторной работы 2 приведены в таблице 5.2. Все три задания выполняются путем составления и реализации программ на ПМК или микроЭВМ.

Перед выполнением **заданий 1** и **2** следует подробно рассмотреть приведенные в тексте примеры (особенно в п. 1.5 главы 1 и п. 4.5 главы 4). Значения параметров n и ε для задания 2 выбирать произвольно в зависимости от характера формулы общего члена, а соответствующие программы вычислений составлять так, чтобы допускалось варьирование значений исходных данных.

Для выполнения **задания 3** составляется программа табулирования функции, находящейся под знаком предела. Участок табулирования выбирается так, чтобы, изменяя на нем соответствующим образом значение аргумента x, можно было бы ответить на поставленные в задании вопросы. При анализе предела функции в точке шаг табулирования выбирается так, чтобы значение аргумента приближалось к предельному (при желании можно получать значения функции для «сгущающихся» значений аргумента, если табулировать не с постоянным шагом, а изменять аргумент каким-либо подходящим способом).

Точно так же анализ поведения функции при $x \to \pm\infty$ проводится при построении такой последовательности значений x, с помощью которой можно было бы судить о поведении функции для достаточно больших по абсолютной величине значений аргумента*.

*При этом, разумеется, нужно помнить, что вывод о наличии предела функции нельзя делать исходя из последовательности значений x частного вида. Применяемый в данном случае прием оправдан, однако, тем, что по условию задания 3 все пределы функций существуют.

Таблица 5.2

№	Задание 1	Задание 2	Задание 3
1	$2^x-2\cos x$	$\frac{n+1}{2^n}$	$\lim\limits_{x\to\infty} e^{\frac{1}{2^{x+1}}}$
2	$\sqrt{x+1}-\frac{1}{x}$	$\frac{(-1)^n}{3^n}$	$\lim\limits_{x\to 3}\frac{x^2-5x+6}{x^2-9}$
3	$5\cos x-x$	$\frac{3n-1}{5n^2+1}$	$\lim\limits_{x\to\frac{\pi}{2}}\frac{\cos x}{\pi-2x}$
4	$\sin x-0{,}2x$	$\frac{2n^2-1}{n^n}$	$\lim\limits_{x\to\infty}\frac{3x^4-2}{\sqrt{x^8+3x+4}}$
5	$\operatorname{tg} x-0{,}5+1$	$\frac{2^n}{n!}$	$\lim\limits_{x\to 2}\frac{x^2-6x+8}{x^2-8x+12}$
6	$\cos x-x^2$	$\frac{1}{2^{\sqrt{n}}-1}$	$\lim\limits_{x\to\infty}\frac{2^x+3}{2^x-3}$
7	$x-\cos x-2$	$n-\sqrt[3]{1-n^3}$	$\lim\limits_{x\to 1}\frac{x^3-6x^2+11x-6}{x^2-3x+2}$
8	$2\sqrt{x}-\cos 0{,}5x$	$\frac{\sqrt{n}}{\sqrt{n^2+1}}$	$\lim\limits_{x\to\infty}(\sin\sqrt{x+1}-\sin\sqrt{x})$
9	$\frac{1}{x}-2\ln x$	$\frac{n^2}{n^3+2}$	$\lim\limits_{x\to\frac{\pi}{4}}\frac{\sin x-\cos x}{\pi-4x}$
10	$\sin 0{,}5x-\sqrt{1-x}$	$\frac{2n-1}{8n^2+3n}$	$\lim\limits_{x\to -1}\frac{\sqrt{4+x+x^2}-2}{x+1}$
11	$x\sin x-1$	$\frac{n}{n^2+3n}$	$\lim\limits_{x\to 0}(1+\operatorname{tg} x)^{\operatorname{ctg} x}$
12	$\operatorname{tg} 2x-3$	$\frac{n}{\sqrt[3]{n^4+n^2}}$	$\lim\limits_{x\to 0}\frac{\operatorname{tg} 4x}{\sin 2x}$
13	$1{,}7-2\cos 2x$	$\frac{2\sqrt{n}}{\sqrt{n^2+n}}$	$\lim\limits_{x\to 0}\frac{\ln\cos x}{\ln(1+x^2)}$
14	$2\operatorname{ctg} 3x+1$	$\frac{2n+3}{5-3n^3}$	$\lim\limits_{x\to\infty}\left(\frac{x+1}{x-1}\right)^x$
15	$\sin 4x-0{,}6$	$\frac{n+1}{\sqrt{n^3+1}}$	$\lim\limits_{x\to 0}\frac{1-\cos 5x}{x^2}$
16	e^x-x-4	$\frac{\sqrt{n^3+2}}{3n^4+2n}$	$\lim\limits_{x\to 0}\frac{\sin 2x}{x}$

Продолжение

№	Задание 1	Задание 2	Задание 3
17	$x^3-\sin x$	$\frac{5n+3}{3^n-n}$	$\lim\limits_{x\to 2}\frac{x^2-7x+10}{x^2-8x+12}$
18	$x\ln x-0{,}5$	$\frac{n^2+1}{n^4+2n}$	$\lim\limits_{x\to\infty}\left(1+\frac{1}{x^2}\right)^x$
19	$2^{-x}-\sin x$	$\frac{n^2+n}{e^n+1}$	$\lim\limits_{x\to 0}\frac{\sqrt{1+x\sin x}-1}{x^2}$
20	3^x-4x	$\frac{2n+1}{n^3-n^2}$	$\lim\limits_{x\to\infty}\left(\frac{2x-1}{2x+1}\right)^x$
21	$x-10\sin x$	$\frac{1}{\sqrt{n^2+1}}$	$\lim\limits_{x\to 0}\frac{x}{\sqrt[3]{1+x}-1}$
22	$x-\cos x-1$	$\frac{2n+1}{n^3+1}$	$\lim\limits_{x\to\infty}\operatorname{arctg} x$
23	$\ln x+\sqrt{x}$	$\frac{n\sqrt[3]{n+1}}{n^3+1}$	$\lim\limits_{x\to 0}\frac{\operatorname{tg} x-\sin x}{x^3}$
24	$10x=e^{-x}$	$\frac{n}{2^n+1}$	$\lim\limits_{x\to\infty}\left(\frac{x+2}{x+1}\right)^x$
25	$2-x=\lg x$	$\frac{3n-1}{5n^2+2}$	$\lim\limits_{x\to 1}\frac{x^5-1}{x^4-1}$

Пусть требуется исследовать предел:

$$\lim_{x\to 0}\frac{1-\cos x}{0{,}5x}.$$

Выражение под знаком предела при $x=0$ дает так называемую «неопределенность вида $\frac{0}{0}$». Составим программу табулирования функции $f(x)=\frac{1-\cos x}{0{,}5x}$ на МК-56 с начальным значением $x=2$ при условии, что все остальные значения x будут получаться по закону геометрической прогрессии со знаменателем 1/2 (т. е. уменьшаться вдвое). Будем предполагать, что начальное значение аргумента x находится в регистре X. Вычисление значения функции удобно оформить в виде подпрограммы. Искомая программа приведена в таблице 5.3.

Введя эту программу в память ПМК и начав вычисления с $x=2$, получим таблицу 5.4.

Продолжая подобные вычисления, следует иметь в виду, что при оперировании с существенно малыми числами калькулятор может давать результаты, не заслуживающие доверия. Из полученной таблицы можно сделать вывод, что пределом данной функции при

Таблица 5.3

Адрес	Команда	Выполняемое действие
00	[x → П] 1	Запись x из регистра X в R1
01	[ПП]	Переход на подпрограмму
02	11	Адрес подпрограммы
03	[С/П]	Остановка и индикация $f(x)$
04	[П → x] 1	Вызов x в регистр X из R1
05	2	Ввод числа 2 в регистр X
06	[÷]	Вычисление $x=x/2$
07	[x → П] 1	Запись x в регистр R1
08	[С/П]	Остановка и индикация x
09	[БП]	Безусловный переход
10	00	Адрес безусловного перехода
11	[F] [x^2]	Вычисление x^2
12	2	Ввод числа 2 в регистр X
13	[÷]	Вычисление $0{,}5x^2$
14	[x → П] 2	Запись $0{,}5x^2$ в регистр R2
15	[П → x] 1	Вызов x в регистр X из R1
16	[F] [cos]	Вычисление $\cos x$
17	1	Ввод числа 1 в регистр X
18	[−]	Вычисление $\cos x-1$
19	[/−/]	Вычисление $1-\cos x$
20	[П → x] 2	Вызов $0{,}5x^2$ в регистр X из R2
21	[÷]	Вычисление $f(x)=\frac{1-\cos x}{0{,}5x^2}$
22	[В/О]	Возврат к команде 03

Таблица 5.4

x	$f(x)$
2	0,70807345
1	0,9193954
0,5	0,9793392
0,25	0,9948032
0,125	0,9986944
0,03125	0,9998336

$x \to 0$ является, вероятно, число 1. Функция четная, поэтому пределы справа и слева будут одинаковы.

При использовании микроЭВМ подобные исследования проводятся с меньшими затратами времени и, что особенно важно, с автоматической выдачей результатов на печать, в том числе и в форме графиков.

ОТВЕТЫ И РЕШЕНИЯ К УПРАЖНЕНИЯМ

Глава 1

1.3

2. а) x [+] y [×] z [=]

x [−] y [×] z [=]

б) x [+] y [:] z [=]

x [−] y [÷] z [=]

в) x [×] y [÷] z [=]

г) x [+] y [×] z [+] u [=]

x [+] y [×] z [−] u [=]

x [−] y [×] z [+] u [=]

x [−] y [×] z [−] u [=]

3. а) y [×] z [+] [/−/] x [=]

б) y [÷] z [+] [/−/] x [=]

4. 1) а) y [+] z [÷] x [↔] [=]

y [−] z [÷] x [↔] [=]

б) y [÷] z [−] x [↔] [=]

в) y [×] z [−] x [↔] [=]

г) z [−] u [÷] y [↔] [−] x [↔] [=]

д) y [+] z [÷] u [−] x [↔] [−] v [=]

е) y [+] z [+] u [÷] x [↔] [−] v [÷] y [−] z [=]

2) а) y [+] z [=] [ЗАП] x [÷] [ИП] [=]

y [−] z [=] [ЗАП] x [÷] [ИП] [=]

б) y [÷] z [=] [ЗАП] x [−] [ИП] [=]

в) y [×] z [=] [ЗАП] x [−] [ИП] [=]

г) z [−] u [=] [ЗАП] y [÷] [ИП] [=] [ЗАП] x [−] [ИП] [=]

д) y [+] z [÷] u [=] [ЗАП] x [−] [ИП] [−] v [=]

е) y [×] z [+] u [=] [ЗАП] x [÷] [ИП] [−] v [÷] y [−] z [=]

6. а) a [×] [×] [×] [ЗАП] b [×] [ИП] [=]

б) y [×] [×] [×] [×] [×] [ЗАП] x [÷] [ИП] [=]

1.4

1. [F] [ДВ] [CF] 4

2. Для МК «Электроника Б3-18А»:

а) a [×] x [=] [F] [ЗАП] b [×] y [+] [F] [ИП] [=]

б) a [+] x [=] [F] [ЗАП] b [+] y [×] [F] [ИП] [=]

в) [C] [F] [ЗАП] a [+] x [=] [F] [П+x²] b [+] y [=] [F] [П+x²] [F] [ИП]

г) b [×] [×] [F] [ЗАП] y [×] [×] [F] [П−] a [+] x [=] [=] [÷] [F] [ИП] [=]

Для МК-41:

а) a [×] x [+] b [×] y [=]

б) [(] a [+] x [)] x [(] b [+] y [)] [=]

в) [(] a [+] x [)] [y^x] 2 [)] [+] [(] b [+] y [)] [y^x] 2 [=]

г) [(] a [+] x [)] [y^x] 2 [)] [÷] [(] b [y^x] 2 [−] y [y^x] 2 [)] [=]

3. На МК-41:

$e^{\pi} - 1$ [e^x] [y^x] [π] [=] (Ответ: 23,14.)

$\pi^e -$ [π] [y^x] 1 [e^x] [=] (Ответ: 22,46.)

4. На Б3-18А:

а) y [F] [cos] [F] [ЗАП] x [F] [ИП] [=]

б) y [F] [cos] [F] [ЗАП] x [F] [√] [F] [ИП] [=]

в) b [F] [√] [F] [ЗАП] a [F] [x^y] [F] [ИП] [=]

г) [C] [F] [ЗАП] x [F] [arc] [cos] [F] [П+] y [F] [e^x] [П+] y [×] [=] [+] x [=] [F] [ln] [F] [ИП] [−] [=]

На МК-41:

а) x [÷] y [cos] [=]

б) x [√] [÷] y [cos] [=]

в) a [y^x] b [√] [=]

г) (x arc cos + y e^x) ÷ (x + y y^x 2) =

1.5

1. а) 3,18 B↑ 24,76 F $\sqrt{}$ ×

б) 0,9 B↑ B↑ × × 0,84 F sin ×

в) 0,7 B↑ 24,3 F x^y x→П 0 0,261 F lg 2,8 − F 1/x П→x 0 +

г) 1,5 B↑ 2,1 F x^y 0,64 F $\sqrt{}$ F sin B↑ 9,71 ×

2. d B↑ c B↑ b B↑ a

3. R_1 F 1/x R_2 F 1/x + R_3 F 1/x + R_4 F 1/x + F 1/x

4.

Адрес команды	Команда	Выполняемое действие
00	B↑	Запись числа R в регистр Y
01	×	Вычисление R^2
02	F π	Ввод числа π в регистр X
03	×	Вычисление πR^2
04	С/П	Остановка и индикация ответа

5. Память: $x=0$ | R0
$h=0{,}1$ | R1

Программа табулирования:

Адрес	Команда
00	П→x 0
01	ПП
02	11
03	С/П
04	П→x 0
05	П→x 1
06	+
07	x→П 0
08	С/П
09	БП
10	00

Адрес	Команда
11	B↑
12	4
13	×
14	x→П 2
15	П→x 0
16	B↑
17	3
18	E x^y
19	П→x 2
20	−
21	B/0

Таблица функции $f(x)=3^x-4x$ на отрезке [0; 2]

x	$f(x)$
0	1
0,1	0,7161232
0,2	0,4457309
0,3	0,1903891
0,4	−0,0481545
0,5	−0,2679493
0,7	−0,6423309
0,8	−0,7917755
0,9	−0,9121248
1	−1,0000006

x	$f(x)$
1,1	−1,0516311
1,2	1,0628078
1,3	−1,0288334
1,4	−0,9444645
1,5	−0,8038484
1,6	−0,6004551
1,7	−0,3269937
1,8	−0,0246724
1,9	0,4636239
2,0	0,999984

Как следует из таблицы, уравнение $3^x-4x=0$ имеет на отрезке [0; 2] два корня: $x_1\in[0{,}3;\ 0{,}4]$ и $x_2\in[1{,}8;\ 1{,}9]$.

6. Значение ε перед пуском программы вводится на индикатор (регистр X). Программа:

Адрес	Команда
00	x→П 0
01	0
02	x→П 1
03	x→П 2
04	1
05	П→x 2
06	+
07	x→П 2
08	F e^x
09	F 1/x
10	П→x 2
11	×
12	x→П 3

Адрес	Команда
13	П→x 0
14	−
15	F x⩾0
16	23
17	П→x 1
18	П→x 3
19	+
20	x→П 1
21	БП
22	04
23	П→x 1
24	С/П

7. Исходные данные (числа a и b) помещаются в регистры R_a и R_b.

Адрес	Команда
00	П→x a
01	П→x b
02	−
03	F x<0
04	12
05	П→x b
06	П→x a
07	x→П b

Адрес	Команда
08	↔
09	x→П a
10	БП
11	00
12	F x=0
13	09
14	П→x b
15	С/П

Глава 2

2.2

1. $\Delta l = 0{,}5$ см; $\delta l = \frac{0{,}5}{63{,}7} \approx 0{,}79\%$.

2. $\Delta(36{,}7) = 0{,}05$; $\delta(36{,}7) = \frac{0{,}05}{36{,}7} \approx 0{,}14\%$;

$\Delta(2489) = 0{,}5$; $\delta(2489) = \frac{0{,}5}{2489} \approx 0{,}02\%$;

$\Delta(31{,}010) = 0{,}005$; $\delta(31{,}010) = \frac{0{,}0005}{31{,}010} \approx 0{,}0016\%$;

$\Delta(0{,}031) = 0{,}0005$; $\delta(0{,}031) = \frac{0{,}0005}{0{,}031} \approx 1{,}6\%$.

3. $0{,}310 \approx 0{,}31$; $\Delta(0{,}31) = 0{,}0005$; 3 цифры;
$8{,}495 \approx 0{,}50$; $\Delta(0{,}50) = 0{,}0005 + 0{,}005 = 0{,}0055$; 2 цифры слева;
$24{,}3790 \approx 24{,}38$; $\Delta(24{,}38) = 0{,}00005 + 0{,}001 = 0{,}00105$; 4 цифры.

4. $\delta(M) = \frac{0{,}01}{5{,}98} \approx 0{,}17\%$;

$\delta(m) = \frac{1}{16} = 6{,}25\%$.

2.3

1. а) $0{,}0007 < 10^{-3}$; 2 цифры; $x \approx 2{,}36$;

б) $0{,}0004 < \frac{1}{2} \cdot 10^{-3}$; 3 цифры; $y = 109{,}6$;

в) $0{,}00005 < 10^{-4}$; 3 цифры; $z = 24{,}31$.

2. а) Один; б) два; в) один.

2.4

1. а) 15,19; $\Delta(15{,}19) = 0{,}01$; $\delta(15{,}19) \approx 0{,}066\%$;
б) 43,337; $\Delta(43{,}337) = 0{,}101$; $\delta(43{,}337) \approx 0{,}015\%$;
г) 0,002; $\Delta(0{,}002) = 0{,}002$; $\delta(0{,}002) = 100\%$.

2.5

1. а) $x = 0{,}47$; $\Delta x = 0{,}005$; $\Delta(\cos x) = |\sin x| \cdot \Delta x \approx 0{,}002$; в значении $\cos 0{,}47 = 0{,}891568288$ верны в строгом смысле 2 значащие цифры; округление по одной запасной: $\cos 0{,}47 \approx 0{,}892$.

б) $x = 0{,}024$; $\Delta x = 0{,}0005$; $\Delta\left(\frac{1}{x}\right) = \frac{\Delta x}{x^2} \approx 0{,}87$;

в значении $\frac{1}{0{,}024} = 41{,}66666666$ верна в строгом смысле 1 зна-

чащая цифра; округление до одной запасной: $\frac{1}{0{,}024} \approx 42$.

е) $x=2{,}6$; $y=1{,}21$; $\Delta x=0{,}05$; $\Delta y=0{,}005$;

$\Delta(x^y)=x^y\left(|y|\frac{\Delta x}{x}+|\ln x|\cdot\Delta y\right)=0{,}028$;

в значении $2{,}6^{1{,}21}=3{,}177735539$ верны в строгом смысле 2 значащие цифры; округление до одной запасной: $2{,}6^{1{,}21}\approx 3{,}18$.

2. а) $\delta(\sqrt{2{,}701})=\frac{1}{2}\cdot\delta(2{,}701)=\frac{1}{2}\cdot 0{,}0001=0{,}00005$. Поскольку $0{,}00005<10^{-4}$, заключаем, что в значении $\sqrt{2{,}701}$ по крайней мере 3 значащие цифры верны в строгом смысле.

Глава 3

3.2

2. а) Схема алгоритма изображена на рисунке 46.

1. чтение x
2. если $x\geqslant 0$, идти к 5
3. $y:=-x$
4. идти к 6
5. $y:=x$
6. запись y
7. конец

б) Схема алгоритма изображена на рисунке 47.

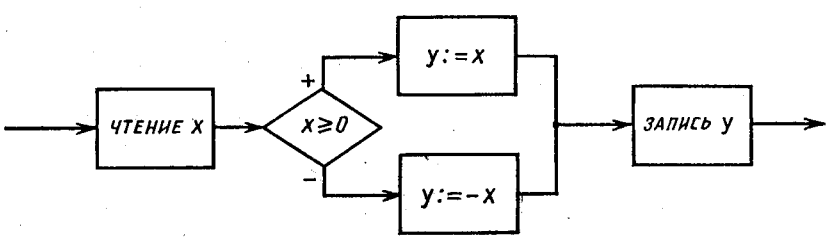


Рис. 46

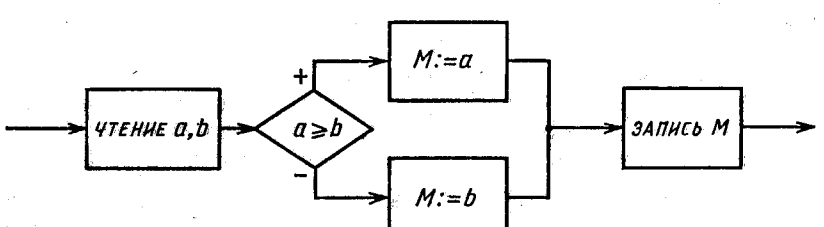


Рис. 47

1. чтение a, b
2. если $a\geqslant b$, идти к 5
3. $M:=b$
4. идти к 6
5. $M:=a$
6. запись M
7. конец

3.3

1. Результатом решения квадратного уравнения $ax^2+bx+c=0$ (в множестве действительных чисел) может быть: а) два корня (при $D=b^2-4ac\geqslant 0$); б) отсутствие корней (при $D<0$). Схема алгоритма решения квадратного уравнения изображена на рисунке 48.
2. а) Схема алгоритма изображена на рисунке 49; б) схема алгоритма изображена на рисунке 50.
3. а) Схема алгоритма изображена на рисунке 51; б) схема алгоритма изображена на рисунке 52.
4. Схема алгоритма изображена на рисунке 53.
5. Схема алгоритма изображена на рисунке 54.
6. Схема алгоритма изображена на рисунке 55.

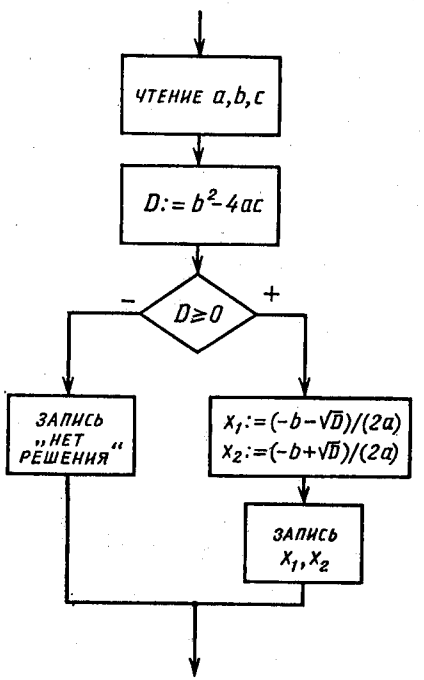


Рис. 48

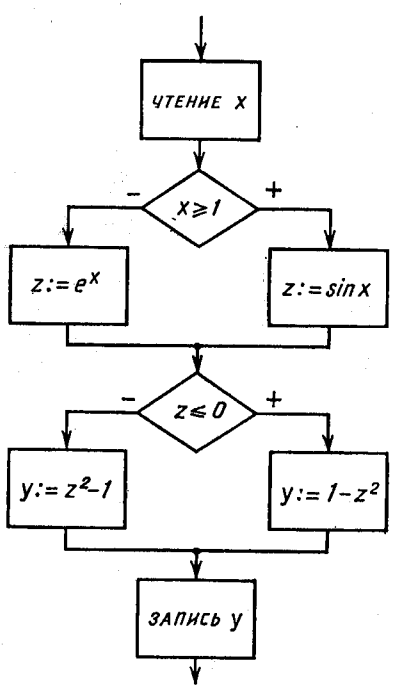


Рис. 49

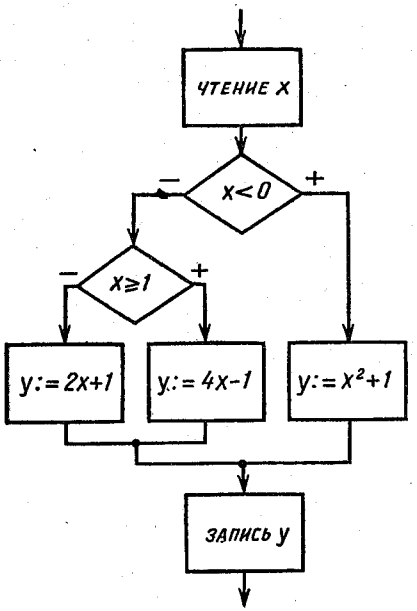


Рис. 50

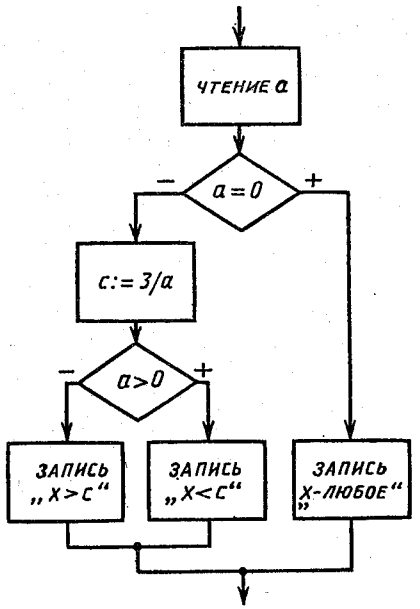


Рис. 51

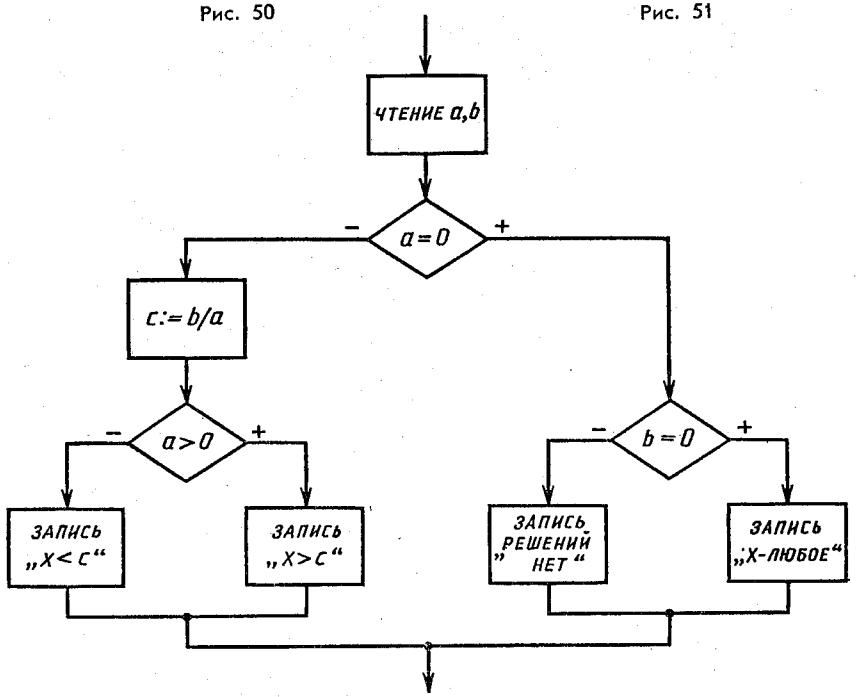


Рис. 52

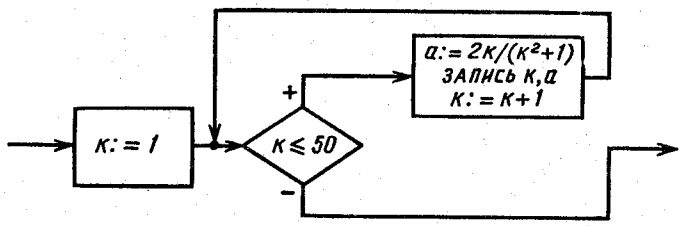


Рис. 53

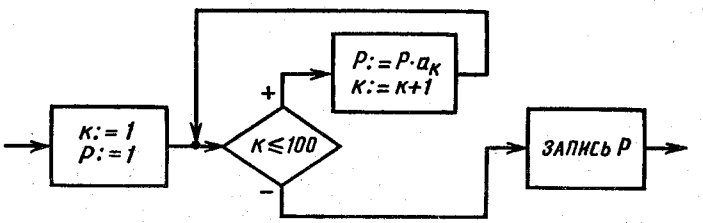


Рис. 54

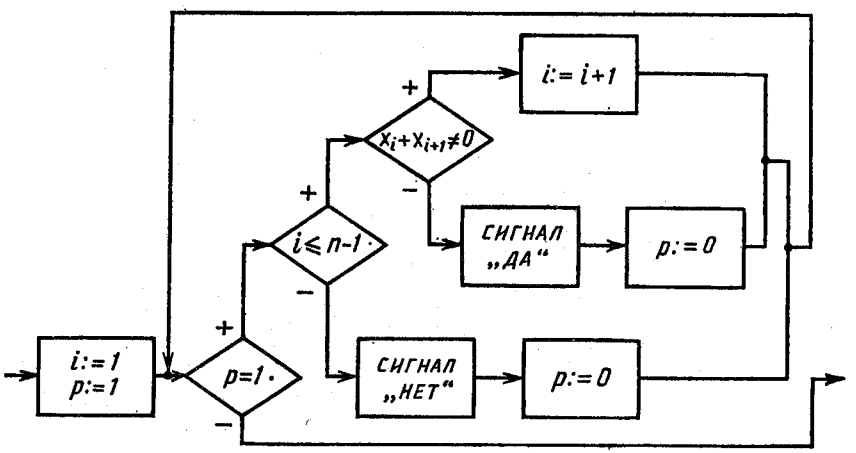


Рис. 55

Глава 4

4.2

1. а) .ØØ94; в) 1.E—3;
   б) —13.Ø37; г) —5.86E4.

2. а), б), в), г), е) действительные; д) целое.

3. а) X↑1.37;
б) A0+A1 * X+A2 * X↑2;
в) (.1275+2 * SIN (SQR (A))) ↑3+2 ↑COS (B);
г) SQR (SIN (X−Y) ↑2−1)/(1+ABS (X ↑2−Y ↑2)).

4. а) $\frac{S-ax}{b}$; в) $\frac{|x-y|}{(1+x)^a}$;
б) $\frac{ade}{bc}$; г) $\frac{4}{a^2ij \cdot \ln |x|}$.

5. а) Истинно; б) ложно; в) истинно; г) истинно.

6. а) Истинно; б) ложно.

4.3

1. а)
```
10 INPUT D, L
20 S=#PI * D * (D+2 * L)/4
30 V=#PI * D ↑2 * SQR (4 * L ↑2−D ↑2)/24
40 PRINT "S="; S, "V="; V
50 GOTO 20
```

б)
```
10 READ D, L
20 S=#PI * D * (D+2 * L)/4
30 V=#PI * D ↑2 * SQR (4 * L ↑2−D ↑2)/24
40 PRINT "S="; S, "V="; V
50 GOTO 20
60 DATA. 24, 1.06, 2.48, 1.84, 12.09, 28.3
```

2.
```
10 REM ТРЕУГОЛЬНИК
20 INPUT R
30 A=2 * R * SQR (3)
40 H=3 * R
50 S=3 * R ↑2 * SQR (3)
60 PRINT "СТОРОНА="; A
70 PRINT "ВЫСОТА="; H
80 PRINT "ПЛОЩАДЬ="; S
90 GOTO 20
```

3.
```
10 REM КОНУС — ШАР
20 PRINT "ВВЕДИТЕ ЗНАЧЕНИЕ РАДИУСА"
30 INPUT P
40 PRINT "ВВЕДИТЕ УРОВЕНЬ ОКРУГЛЕНИЯ"
50 INPUT N
60 REM ОБЪЕМ V1 КОНУСА С ШАРОМ
70 V1=3 * #PI * R ↑3
80 REM ОБЪЕМ V2 КОНУСА БЕЗ ШАРА
90 V2=5 * #PI * R ↑3/3
100 REM ВЫСОТА H КОНУСА БЕЗ ШАРА
110 H=3 * P * SQR (5)
120 PRINT "УРОВЕНЬ ВОДЫ="; ROUND (H, N)
130 GOTO 20
```


4.4

1. а)
```
10 REM F (Z)
20 INPUT Z
30 IF Z<1THEN 50
40 F=Z ↑2:GOTO 60
50 F=1−Z
60 PRINT "F ("; Z; ") ="; F
70 GOTO 20
```

б)
```
10 REM F (X)
20 INPUT X
30 IF X−INT (X) =0 THEN 50
40 F=1/TAN (X):GOTO 60
50 F=0
60 PRINT "F ("; X; ") ="; F
70 GOTO 20
```

2.
```
10 REM БИТ
20 INPUT A, B, C
30 IF A>B THEN 50
40 Y=B:GOTO 60
50 Y=A
60 IF Y>C THEN 80
70 Y=C
80 PRINT "MAX ("; A; ", "; B; ", "; C·") ="; Y
90 GOTO 20
```

3.
```
10 REM КВУРДЕЙСТВ
20 INPUT A, B, C
30 D=B ↑2−4 * A * C
40 IF D<0 THEN 90
50 X1=(−B−SQR (D))/(2 * A)
60 X2=(−B+SQR (D))/(2 * A)
70 PRINT "X1="; X1; "X2="; X2
80 GOTO
90 PRINT "РЕШЕНИЙ НЕТ"
100 GOTO 20
```

4.
```
10 REM КВУРКОМПЛЕКС
20 INPUT A, B, C
30 D=B ↑2−4 * A * C
40 IF D<0 THEN 80
50 R1=(−B−SQR (D))/(2 * A)
60 R2=(−B+SQR (D))/(2 * A)
```

```
7Ø I1, I2=Ø:GOTO 1ØØ
8Ø R1, R2=-B/(2*A)
9Ø I1=SQR (-D)/(2*A):I2=-I1
1ØØ PRINT "X1=("; R1; ")+I* ("; I1; ")"
11Ø PRINT "X2=("; R2; ")+I* ("; I2; ")"
12Ø GOTO 2Ø
```

5. а)
```
1Ø REM F (X)
2Ø INPUT X
3Ø IF X<-1 THEN 7Ø
4Ø IF X<1 THEN 6Ø
5Ø F=ARCCOS (X↑2):GOTO 8Ø
6Ø F=LOC (X+.8):GOTO 8Ø
7Ø F=SIN (X)/(1-X↑3)
8Ø PRINT "F ("; X; ")="; F
9Ø GOTO 2Ø
```

б)
```
1Ø REM F (V (X))
2Ø INPUT X
3Ø IF X<Ø THEN 5Ø
4Ø U=SQR (X):GOTO 6Ø
5Ø U=LOG (-X)
6Ø IF U<=Ø THEN 8Ø
7Ø F=COS (U):GOTO 9Ø
8Ø F=2*U+1
9Ø PRINT "F (U ("; X; "))="; F
1ØØ GOTO 2Ø
```

4.5

1. а)
```
1Ø REM IF (ЦИКЛ-ПОКА)
2Ø K=1
3Ø IF K>5Ø THEN 8Ø
4Ø B=SQR (K)/(K+.5)
5Ø PRINT "B ("; K; ")="; B
6Ø K=K+1
7Ø GOTO 3Ø
8Ø END
```

```
1Ø REM IF (ЦИКЛ-ДО)
2Ø K=1
3Ø B=SQR (K)/(K+.5)
4Ø PRINT "B ("; K; ")="; B
5Ø K=K+1
6Ø IF K<=5Ø THEN 3Ø
7Ø END
```

б)
```
1Ø REM FOR-NEXT
2Ø FOR K=1 TO 5Ø
3Ø B=SQR (K)/(K+.5)
4Ø PRINT "B ("; K; ")="; B
5Ø NEXT K
```


2. а)
```
1Ø REM TAB F1 (X)
2Ø PRINT "X", "F1 (X)"
3Ø FOR X=-1 TO 1 STEP .1
4Ø F1=2*(SIN (X)↑3)/(1+X↑2)
5Ø PRINT X, F1
6Ø NEXT X
```

б)
```
1Ø REM TAB F2 (X)
2Ø PRINT "X", "F2 (X)"
3Ø FOR X=-10 TO 10
4Ø IF ABS (X)<4 THEN 6Ø
5Ø F2=LOG (ABS (X-.6)):GOTO7Ø
6Ø F2=SQR (2*X↑2+1)
7Ø PRINT X, F2
8Ø NEXT X
```

3. а)
```
1Ø СУММА S (N)
2Ø INPUT N
3Ø K=1:S=Ø
4Ø IF K>N THEN 7Ø
5Ø S=8+SQR (K)/(K↑3+.8)
6Ø K=K+1:GOTO 4Ø
7Ø PRINT "СУММА S (N)="; S
8Ø END
```

б)
```
1Ø REM СУММА S (E)
2Ø INPUT E
3Ø K=1:=0:A=1/1.8
4Ø IF A<E THEN 8Ø
5Ø S=S+A:K=K+1
6Ø A=SQR (K)/(K↑3+.8)
7Ø GOTO 40
8Ø PRINT "СУММА S (E)="; S
9Ø END
```

4. $a_n=\frac{n+1}{n!}$, $a_{n+1}=\frac{n+2}{(n+1)!}$, $\frac{a_{n+1}}{a_n}=\frac{(n+2)\cdot n!}{(n+1)!\,(n+1)}=\frac{n+2}{(n+1)^2}$,

т. е. $a_{n+1}=\frac{n+2}{(n+1)^2}\cdot a_n$.

Учитывая, что $a_1=2$, все члены заданного ряда, начиная со второго, могут быть получены по формуле $a_{n+1}=\frac{n+2}{(n+1)^2}\cdot a_n$ при $n=1, 2, \ldots, 49$. Программа:

```
1Ø REM РЕКУРР
2Ø C=2:N=1
3Ø IF N>49 THEN 7Ø
4Ø A=A*(N+2)/(N+1)↑2
5Ø S=S+A:N=N+1
```

```
60 GOTO 30
70 PRINT "S="; S
80 END
```

5. а)
```
10 REM ПОИСК ОДНОМЕРНЫЙ
20 DIM X (10)
30 FOR K=1 TO 10
40 INPUT X (K)
50 NEXT K
60 M=X (I)
70 FOR K≠2 TO 10
80 IF M>=X (K) THEN 100
90 M=X (K)
100 NEXT K
110 PRINT "M="; M
120 END
```

6.
```
10 REM F (X, Y)
20 DIM X (10), Y (20)
30 FOR K=1 TO 10
40 INPUT X (K)
50 NEXT K
60 FOR K=1 TO 20
70 INPUT Y (K)
80 NEXT K
90 PRINT "X", "Y", "F (X, Y)"
100 FOR I=1 TO 10
110 FORJ=1 TO 20
120 F=SQR (X (I) ↑2+Y (J) ↑2)
130 PRINT X (I), Y (J), F
140 NEXT J: NEXT I
```


ПРИЛОЖЕНИЕ I

Перечень сообщений об ошибках для ЭВМ «Искра-226»*

Номер ошибки	Причины, рекомендации по устранению
SYSTEM ERR 00 SYSTEM ERR 01 SYSTEM ERR 02	Системные ошибки. При случайном системном сбое пользователю рекомендуется попытаться устранить причину, изменив последовательность операций, приведших к сбою, используя кнопку RESET, оператор CLEAR, либо выключить и еще раз включить машину. Если машину не удалось вывести из этого состояния и сбой носит постоянный характер, то необходим ремонт
ERR 01	Текстовое переполнение. Использован весь доступный пользователю объем оперативной памяти. Укоротить программу и (или) разбить ее на части
ERR 02	Переполнение таблиц. Отведенное в ОЗУ место на запоминание внутренних операционных таблиц и идентификаторов переменных полностью использовано либо недопустимая глубина вложений программ и циклов. Укоротить, отредактировать и (или) разбить программу на сегменты
ERR 03	Математическая ошибка: нарушение областей определения математических функций, деление на нуль и т. п.
ERR 06	Синтаксическая ошибка при вводе с клавиатуры. Исправить текст строки
ERR 11	Не определена строка, на которую произведена ссылка. Исправить текст программы
ERR 12	При счете по программе встретился ошибочный оператор. Исправить текст оператора
ERR 18	Использование недопустимой величины (числа, размерности, массива, индекса). Например, размерность больше, чем 7999, либо индекс массива превышает заданную размерность
ERR 20	Недопустимый формат числа, порядок больше 99
ERR 22	При выполнении программы встретились неопределенный массив или переменная

* Полный перечень сообщений об ошибках можно найти в технической документации микроЭВМ «Искра-226».

Продолжение

Номер ошибки	Причины, рекомендации по устранению
ERR 25	Нарушение в организации пар сопряженных операторов FOR/NEXT (или COSUB/RETURN). Например, для встретившегося в программе оператора NEXT отсутствует спряженный оператор FOR или осуществлен переход внутрь цикла (подпрограммы)
ERR 27	При исполнении оператора READ данных, перечисленных в DATA, не хватило или данные, указанные в операторе RESTORE, находились за допустимыми пределами. Исправить текст программы, дополнив ее нужным количеством данных, либо исправить оператор RESTORE
ERR 29	Недопустимый формат данных в операторе INPUT
ERR 42	Недопустимое присвоение целой переменной (число большее, чем 7999)
ERR 45	Длина оператора превышает 255 байтов. Исправить текст оператора

ПРИЛОЖЕНИЕ II

$$|X-x| \leqslant \Delta x \qquad \delta x = \frac{\Delta x}{|x|}$$

Δx — граница абсолютной погрешности приближения x

δx — граница относительной погрешности приближения x

Формулы погрешностей арифметических действий

$x * y$	$\Delta(x * y)$	$\delta(x * y)$
$x+y$	$\Delta x+\Delta y$	$\frac{\lvert x\rvert}{\lvert x+y\rvert}\delta x+\frac{\lvert y\rvert}{\lvert x+y\rvert}\delta y$
$x-y$	$\Delta x+\Delta y$	$\frac{\lvert x\rvert}{\lvert x-y\rvert}\delta x+\frac{\lvert y\rvert}{\lvert x-y\rvert}\delta y$
$x\cdot y$	$x\cdot\Delta y+y\cdot\Delta x$	$\delta x+\delta y$
x/y	$\frac{x\cdot\Delta y+y\cdot\Delta x}{y^2}$	$\delta x+\delta y$

ПРИЛОЖЕНИЕ III

Служебные слова языка Бейсик

Английские служебные слова	Русское произношение	Смысловой перевод
LET	лет	пусть
GO TO	гоуту	перейти на
IF	иф	если
THEN	зен	то
FOR	фо	для
TO	ту	до
STEP	стэп	шаг
NEXT	некст	следующий
DATA	дейт	данные
READ	рид	читать
INPUT	инпут	ввести
PRINT	принт	печатать
END	энд	конец
DIM(ENSION)	димэнш	размерность
RUN	ран	пуск
ERROR	эрро	ошибка
REM(ARK)	римак	примечание
BACK SPACE	бэк спэйс	обратный ход
LINE	лайн	линия
EDIT	эдит	редактирование
RECALL	рикол	отзывать
DELETE	дилит	вычеркивать
ERASE	ирэйз	стирать
INSERT	инсет	вставить
CLEAR	клиэ	очищать
ROUND	раунд	округлять
LIST	лист	список
SELECT	селект	выбирать

ЛИТЕРАТУРА

1. Антипов И. Н. Основные приемы вычислений на микрокалькуляторе «Электроника Б3-18А».— М.: Высшая школа, 1980.

2. Белый Ю. А. Считающая микроэлектроника.— М.: Наука, 1984.

3. Блох А. Ш., Павловский А. И., Пенкрат В. В. Программирование на микрокалькуляторах.— Минск: Вышэйшая школа, 1981.

4. Гильде В., Альтрихтер З. С микрокалькулятором в руках / Пер. с нем. Ю. А. Данилова.— М.: Мир, 1980.

5. Гутер Р. С., Полунов Ю. Л. От абака до компьютера.— М.: Знание, 1975.

6. Иванова Т. П., Пухова Г. В. Вычислительная математика и программирование.— М.: Просвещение, 1979.

7. Ковалев М. П., Шварцбурд С. И. Электроника помогает считать.— М.: Просвещение, 1978.

8. Машина вычислительная электронная клавишная программируемая «Искра-226». Инструкция по программированию (базовый объем) 1.320.136.Д14-1. Часть I, 1983.

9. Основы информатики и вычислительной техники: Проб. учеб. пособие для сред. учеб. заведений. Ч. I, II / Под ред. А. П. Ершова, В. М. Монахова.— М.: Просвещение, 1985, 1986.

10. Техническое описание и инструкции по эксплуатации микрокалькуляторов Б3-26, МК-44, Б3-18А, МК-41, МК-56 и др.

ОГЛАВЛЕНИЕ

Предисловие 3
Глава 1. Техника вычислений на микрокалькуляторах 5
1.1. Общие сведения о микрокалькуляторах . . . . . —
1.2. Способы представления чисел 7
1.3. Вычисления на арифметических микрокалькуляторах 8
1.4. Микрокалькуляторы инженерного типа 22
1.5. Программируемые микрокалькуляторы 36
Глава 2. Методы оценки погрешностей 52
2.1. Источники ошибок в вычислениях по формуле . . . . . —
2.2. Основные понятия теории приближенных вычислений 55
2.3. Определение количества верных цифр по относительной погрешности приближенного числа 57
2.4. Формулы для подсчета погрешностей арифметических действий 60
2.5. Формулы для подсчета погрешностей значений элементарных функций 63
2.6. Практика вычислений 69
Глава 3. Алгоритмы 78
3.1. Принцип алгоритмизации в использовании вычислительной техники —
3.2. Алгоритмы и способы их представления 79
3.3. Конструирование алгоритмов 85
Глава 4. Элементы программирования на языке Бейсик 95
4.1. Основные устройства ЭВМ . . . . . —
4.2. Алфавит и простейшие конструкции языка Бейсик 97
4.3. Организация программы 103
4.4. Программирование ветвлений 109
4.5. Программирование циклов 114
4.6. Загрузка, редактирование и пуск программы 125
Глава 5. Лабораторный практикум 130
5.1. Лабораторная работа 1 . . . . . —
5.2. Лабораторная работа 2 134
Ответы и решения к упражнениям 139
Приложения. 155
Литература 158

Валерий Михайлович Заварыкин
Владимир Габриэлович Житомирский
Михаил Павлович Лапчик

ТЕХНИКА ВЫЧИСЛЕНИЙ И АЛГОРИТМИЗАЦИЯ

Вводный курс

Зав. редакцией Р. А. Хабиб
Редактор Н. А. Песина
Младшие редакторы Л. Е. Козырева, Е. А. Сафронова
Художественный редактор Е. Н. Карасик
Технические редакторы Н. Т. Щербак, Р. С. Невретдинова
Корректор М. Ю. Сергеева

ИБ № 10498

Сдано в набор 30.07.86. Подписано к печати 24.03.87. Формат 60 × 90¹/₁₆. Бум. типограф. № 2. Гарнит. литер. Печать офсетная. Усл. печ. л. 10. Усл. кр. отт. 10,25. Уч.-изд. л. 8,03. Тираж 98 000 экз. Заказ № 385. Цена 25 коп.

Ордена Трудового Красного Знамени издательство «Просвещение» Государственного комитета РСФСР по делам издательств, полиграфии и книжной торговли. 129846, Москва, 3-й проезд Марьиной рощи, 41.

Саратовский ордена Трудового Красного Знамени полиграфический комбинат Росглавполиграфпрома Государственного комитета РСФСР по делам издательств, полиграфии и книжной торговли. 410004, Саратов, ул. Чернышевского, 59.