МИКРОКОМПЬЮТЕР "ЭЛЕКТРОНИКА МК 90"

РУКОВОДСТВО ПО ПРОГРАММИРОВАНИЮ

1988

Заказ 1734 Тираж 1500

Ротапринт областного управления статистики

МИКРОКОМПЬЮТЕР

"Электроника МК 90"

Руководство по программированию

1988

I. ПОРТАТИВНАЯ МИКРО-ЭВМ ЭЛЕКТРОНИКА МК 90

I.I. Назначение ЭВМ

Портативная микро-ЭВМ "Электроника МК 90" предназначена для выполнения научных, инженерных и статистических расчетов, оперативной обработки текстовой и графической информации.

Отличительной чертой ПЭВМ является компактность, обеспечивающая выполнение работы там, где это необходимо пользователю и так, как ему удобно. Это позволяет исключить пользователя из числа периферийных устройств ЭВМ, расширить область применения ЭВМ.

Компактность особенно важна для целей обучения. Снижается психологический барьер между человеком и ЭВМ, облегчается усвоение составляющих вычислительного процесса, подчеркивается их единство и ведущая роль математического моделирования.

I.2. Общие технические характеристики ЭВМ

Микропроцессор - I6-разрядный, совместимый по системе команд с микро-ЭВМ "Электроника-60".

Оперативное запоминающее устройство - I6 Кбайт.

Постоянное запоминающее устройство - 32 Кбайт.

Внешняя память - сменные модули I0 Кбайт ОЗУ и I92 Кбайт ПЗУ, подключаемые по отдельности или в любой попарной комбинации.

Язык программирования - Бейсик.

Дисплей (ЖКИ) матричного типа с количеством точек изображения I20x64 или 8 строк по 20 символов.

Звуковой сигнал, программируемый по продолжительности и току в диапазоне от I00 Гц до I кГц.

Габаритные размеры 30 мм x 250 мм x I00 мм.

Масса не более 0,7 кг.

Потребляемая мощность не более 0,2 Вт.

I.3. Язык программирования

Для пользователя языком программирования и общения с ЭВМ является Бейсик (BASIC) - наиболее популярный язык, используемый для работы на малых вычислительных машинах. Он универсален, довольно прост и его легко выучить.

Имеется несколько вариантов языка Бейсик, учитывающих особенности

ЭВМ. В данной ЭВМ используется вариант языка с развитыми средствами обработки графической информации. Полное описание языка приведено в Руководстве по эксплуатации ЭВМ.

Основой языка ПЭВМ является язык ее вычислительного устройства – микропроцессора. Команды такого машинного языка – это последовательности нулей и единиц, которые представляются комбинациями электрических импульсов. Машинный язык непосредственно не используется – работать с ним для пользователя утомительно.

Специальная программа – "Интерпретатор Бейсика", находящаяся в ПЗУ ЭВМ, преобразует (интерпретирует) Бейсик – программу пользователя в программу на машинном языке. Затем программа выполняется.

I.4. Клавиатура

На передней панели ПЭВМ с правой стороны расположены 63 клавиши: 7 первых рядов по 8 клавиш и последний, восьмой ряд – 7 клавиш.

Клавиши работают от легкого и четкого нажатия до упора, различаются цветом и символикой.

Для лучшего ориентирования пронумеруем столбцы (вертикальные ряды) клавиш от 1 до 8. Условимся при обозначении клавиши указывать номера ее столбца и ряда. Например, клавиша 7 [ВК] (8) – односимвольная клавиша 8 столбца, 7 ряда. Она обеспечивает Выполнение любой Команды, заканчивая последовательность нажимаемых клавиш.

Односимвольных клавиш – 13.

19 клавиш – двухсимвольные, у них один символ нанесен на клавише, а второй – над клавишей. Эти клавиши "работают" в сочетании с переключателями символов – клавишами 8 [Р/Л] (1) и 8 [В/Н] (8)

Р/Л – установка Русского / Латинского шрифтов для букв

В/Н – установка Верхнего / Нижнего регистра для букв и символов.

Одна широкая клавиша нижнего ряда 8 [ ] (4,5) – бессимвольная, обеспечивает пропуск (пробел) между символами при их наборе.

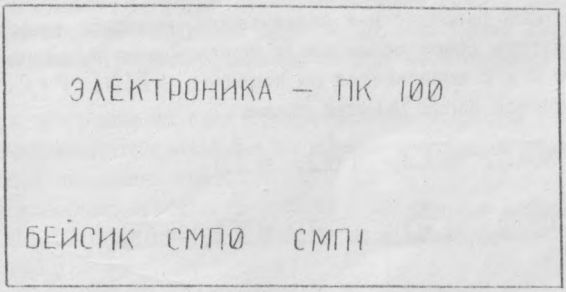
Клавиша включения ПЭВМ 2 [⏻] (1), выключения 1 [⏻] (1).

I.5. Дисплей

Основным средством взаимодействия пользователя с ПЭВМ является экран видеодисплея, расположенный на передней панели слева. Он используется для считывания результатов расчета и контроля при вводе информации.

На экране можно разместить до 8 строк текста, максимальная длина строки 20 символов.

При включении ПЭВМ (2 [Ш]) на экране отображается информация



ЭЛЕКТРОНИКА – ПК 100 – название ПЭВМ

БЕЙСИК – язык программирования

СМП0, СМП1 – сменные модули памяти с номерами 0 и 1.

Дополнительные пояснения для этой строки приводятся ниже.

После включения нажимается клавиша [ВК] и на экране появляется информация

```
Basic V 1.0  бр лат
BASIC V 1.0 (c) 1986
Готов
_
```

ПЭВМ готова к работе (ГОТОВ).

Первая строка экрана – служебная, используется для указания положения переключателей символов:

ВР – установлен верхний регистр для букв и символов.

лат – установлен латинский шрифт для букв.

При нажатии клавиши 8 [F] в первой строке слева от ВР появляется знак ФК (нажата Функциональная Клавиша). Эта клавиша обеспечивает сокращенный набор на клавиатуре команд Бейсика.

Черточка под словом ГОТОВ указывает положение курсора, т.е. место, с которого будет отображаться дальнейшая информация.

I.6. Режимы работы

Основными рабочими режимами ПЭВМ являются:

I. Режим непосредственных вычислений.

В этом режиме ПЭВМ работает как калькулятор, выполняя вычисления и отображая результаты сразу после ввода вычисляемого выражения. Например, выражение 2 x 2 вычисляется по команде: PRINT 2*2 и результат (4) появляется после нажатия клавиш:

^{8}[P], ^{1}[R], ^{1}[I], ^{6}[N], ^{3}[T],

^{3}[2], ^{8}[B/H], ^{7}[*], ^{8}[B/H], ^{3}[2], ^{8}[BK].

2. Режим вычислений по программе.

В этом режиме ПЭВМ работает как ЭВМ, выполняя вычисления и отображая результаты после ввода программы в оперативное запоминающее устройство (ОЗУ) и запуска ее на счет.

Программа вычисления выражения 2 x 2 принимает вид:

```
10 PRINT 2*2: END
```

а ее запуск на счет обеспечивается командой RUN

Последовательность нажимаемых клавиш:

1) [1] [0] (номер строки программы)
2) [F] [P] [2] [B/H] [*] [B/H] [2] (первый оператор)
3) [:] (разделитель операторов)
4) [F] [E] (второй и последний оператор программы)
5) [BK] (ввод однострочной программы в ОЗУ)
6) [F] [R] [BK] (запуск программы на счет)

Сделаем пояснения.

I. В отличие от предыдущего примера здесь использован сокращенный набор команд PRINT , END , RUN по первым буквам, совместно с функциональной клавишей [F] Полный список команд приведен в Руководстве по эксплуатации ПЭВМ.

2. Указанием программного режима является номер (I0) строки. Строки можно нумеровать от I до 8I9I с любым, удобным для пользователя, шагом.

Если строку набрать без номера, то после нажатия клавиши [BK] она вы-

полнится немедленно, т.е. в режиме непосредственных вычислений.

3. Окончанием набора строки и ввода ее в ОЗУ служит нажатие клавиши [ВК].

4. Ввод программы и запуск ее на выполнение (счет) раздельные операции, не обязательно следующие друг за другом. После ввода возможна проверка программы (редактирование), запись ее в сменный модуль памяти.

5. Запуск программы на счет осуществляется командой RUN. Результаты вычислений отображаются на экране в процессе выполнения программы и после окончания счета.

6. После завершения счета на экране отображается информация:

ОСТ В СТРОКЕ <N> (N = I0 в примере)

ГОТОВ

ОСТ(ановка) В СТРОКЕ <N>(программы), подтверждающая выполнение последней строки с номером N и всей программы.

Повторный запуск программы на счет осуществляется командой RUN

7. Для исправления ошибок при наборе используется клавиша $\overset{7}{\leftarrow}$[ЗБ]. Однократное ее нажатие "стирает" последний введенный символ, нажимая несколько раз можно стереть всю "фразу".

I.7. Средства памяти

В комплекте ПЭВМ имеются запоминающие устройства (ЗУ) различного назначения и емкости. Емкость ЗУ измеряется в единицах, называемых байтами. Один байт памяти ЭВМ способен хранить только один символ, поэтому можно рассматривать байты как информационные символы.

2^{10} = I024 байт называют килобайтом и обозначают I Кбайт.

ПЭВМ оснащена постоянной (неизменяемой) памятью называемой постоянным запоминающим устройством (ПЗУ), емкостью 32 Кбайт. В ПЗУ хранится информация, нужная для обработки (интерпретирования) команд на языке Бейсик и для выполнения других функций.

Эта информация записывается в ПЗУ в нестираемой форме в процессе его изготовления и не изменяется даже при выключении ПЭВМ. Пользователь прямого доступа к ПЗУ не имеет.

Динамическая (изменяющаяся) память ПЭВМ называется оперативным запоминающим устройством (ОЗУ) Емкость ОЗУ - I6 Кбайт.

В ОЗУ информация может храниться и обрабатываться в любой момент работы ПЭВМ, но при выключении ПЭВМ информация уничтожается.

Главное достоинство ОЗУ - изменчивость, благодаря которой ее ячейки можно многократно использовать для различных целей. Например, в

разные моменты выполнения программы хранить в ячейке разные числа. В ОЗУ пользователь записывает программу для последующего ее выполнения.

Внешняя память - дополнительные запоминающие устройства, выполняющие роль банков для хранения информации. Например, программ пользователя.

В качестве внешней памяти в ПЭВМ используются сменные модули памяти (СМП) емкостью 10 Кбайт и 192 Кбайт.

СМП 10 Кбайт - пользовательские модули. В них пользователь может переписать из ОЗУ программы для длительного хранения и, соответственно, вызвать их в ОЗУ для выполнения.

СМП 192 Кбайт - внешнее ПЗУ, информация в этот модуль записывается в нестираемой форме в процессе его изготовления. Обмен информацией между этим модулем и ОЗУ - односторонний: считывание из модуля в ОЗУ.

Модули 10 Кбайт и 192 Кбайт могут подключаться к ПЭВМ в отдельности или в любой попарной комбинации, устанавливаются в отсек, находящийся с задней стороны ПЭВМ. Отсек закрывается крышкой с надписью "Блок памяти".

1.8. Управление звуковым сигналом

ПЭВМ может воспроизводить звуки и музыку при помощи встроенного звукогенератора, используя команду PLAY . Эта команда управляет частотой и продолжительностью звука, но не его громкостью.

Генерируются только чистые тона; непосредственного способа их целевого искажения для различных акустических эффектов не существует. Например, тон соль диез первой октавы воспроизводится по команде:

PLAY 27, 128, где 27 - номер тона, 128 - длительность звучания (~ целая нота).

Выполним команду, нажимая клавиши:

1) [F] 5 [8] (сокращенный набор PLAY)
2) [2] [7] (номер тона)
3) 7 [4] (разделитель)
4) [1] [2] [8] (длительность звучания)
5) [BR] (выполнение команды)

Всего воспроизводится 41 тон от фа большой октавы (номер 0) до ля второй октавы (номер 40). Нумерация тонов приведена в Руководстве по эксплуатации ПЭВМ.

Воспроизведение всего звукоряда можно выполнить по программе

```
10 FOR K=0 TO 40
20 PLAY K, 128
30 NEXT K
40 END
```

2. РЕЖИМ НЕПОСРЕДСТВЕННЫХ ВЫЧИСЛЕНИЙ

В режиме непосредственных вычислений команды Бейсика выполняются сразу после ввода их в ОЗУ и результат отображается на экране. Это дает возможность производить несложные численные расчеты так же, как на калькуляторе.

2.1. Ввод, запись и чтение чисел

Ввод чисел в ОЗУ производится нажатием цифровых клавиш 1 [1], 1 [2], ..., 2 [9], 2 [0] в порядке следования цифр при чтении числа.

Если число дробное, то в начале вводится целая часть, нажатием клавиши 7 [.] устанавливается десятичная точка, затем вводится дробная часть. Введенное так число - положительное.

При вводе отрицательного числа, вначале, нажатием клавиши 2 [–] устанавливается отрицательный знак числа, затем вводится мантисса числа.

При вводе чисел с десятичным порядком сначала устанавливается знак числа, вводится мантисса числа, затем нажимается клавиша 3 [E] (установка основания порядка - 10) и вводится значение порядка.

Все указанные элементы числа набираются при установке переключателя В/Н в положение ВР.

Ограничения при вводе чисел:

1. Максимальное число значащих цифр числа - 7. Это означает, что наибольшее целое число, вводимое в ПЭВМ, равно 9999999. Наименьшее целое число равно - 9999999.

Наибольшая правильная дробь: .9999999, наименьшая положительная правильная дробь: .0000001. Целая часть (0) правильной дроби может не указываться.

Если число не целое, то сумма цифр целой и дробной его части не

должна превышать 7.

При невыполнении указанного ограничения производится округление числа, причем, если первая отбрасываемая цифра (восьмая по счету) больше 4, то последняя сохраняемая цифра (седьмая по счету) увеличивается на единицу.

2. Максимальное число, воспринимаемое ПЭВМ, равно $9999999\ 10^{980}$

3. Минимальное число для ПЭВМ равно $-\ 99999999\ 10^{-980}$

Запись числа в ОЗУ связана с запоминанием его в какой-либо ячейке памяти. Указанием конкретной ячейки служит ее имя, выполняющее роль номера ячейки. Имя ячейки составляется из больших латинских букв и цифр:

A , A1 , A9 , Z0 и т.д.

При составлении имени действуют правила:

- имя должно начинаться с буквы и содержать одну букву.
- за буквой в имени может следовать не более одной цифры.

Так, используя букву T , можно составить 11 разных имен ячеек:

T , T0, T1, T2, T3, T4, T5, T6, T7, T8, T9.

Команда записи числа < N > в ячейку памяти с именем A пишется так:

LET A = N

Например, LET A = 1 - запись числа 1 в ячейку памяти с именем A реализуется нажатием клавиш: (В/Н - ВР)

[F] 4[L]	- сокращенный набор команды LET
3[A]	- набор имени ячейки
[В/Н]	- переход на нижний регистр
2[=]	- установка связи числа с ячейкой
[В/Н]	- переход на верхний регистр
[1]	- набор числа
[ВК]	- запись числа в ячейку памяти.

Подтверждением записи является перемещение курсора влево вниз относительно набранной команды.

Чтение числа - это отображение его на экране, либо вызов из ячейки памяти для выполнения вычислительной операции.

Для отображения числа на экран используется команда PRINT . Команда применяется при отображении вводимого числа или числа, записан-

ного предварительно в ячейку памяти.

Например, команда PRINT I совмещает ввод и отображение числа.

Или: I) LETA=1 – ввод и запись числа I в ячейку A ;

2) PRINTA – отображение на экран содержимого A (I).

Константа π , часто используемая в расчетах, хранится в памяти ПЭВМ и отображается на экран командой PRINT PI : (В/Н-ВР)

[F] [P] набор команды PRINT

[P] [I] – набор имя ячейки π

[ВК] – отображение на экране π =3.I4I593 с округлением

2.2. Арифметические операции

ПЭВМ выполняет с числами пять арифметических опрраций, приведенных в таблице (A , B числа):

Операция	Клавиша	В/Н	Приоритет	Пример
Возведение в степень	8 5 [¬] 7	ВР	Первый	A ¬ B
Деление	2 [/] 7	ВР	Второй	A / B
Умножение	1 [*] 7	НР	Второй	A * B
Вычитание	2 [–] 3	ВР	Третий	A – B
Сложение	1 [+] 3	НР	Третий	A + B

Приоритет операций определяет порядок их выполнения при вычислении арифметических выражений. Он может быть изменен вводом скобок, так как выражения в скобках вычисляются в первую очередь.

К моменту выполнения операции величины A , B должны иметь конкретные числовые значения.

При вычислении арифметических выражений используются команды PRINT , LET

Пример: вычислить выражение $\frac{(5+3)\cdot(8-6)}{4} - 0.7$.

Приведем выражение к "вычислительному" для ПЭВМ виду:

(5+3)*(8-6)/4-.7

Этим устанавливается порядок выполнения операций

I) 5 + 3 = 8

2) 8 - 6 = 2

3) 8 x 2 = I6

4) I6/ 4 = 4

5) 4 - .7 = 3.3

Запишем два варианта вычислительных команд:

I. PRINT (5+3)*(8-6)/4-.7

Набор клавиш (В/Н-ВР)

а) [F] [P] [В/Н]

б) 2 [(]4 [В/Н] [5] [В/Н] [+] [В/Н] [3] [В/Н]

в) 2 [)]5 [*] [(] [В/Н] [8] [-] [6] [В/Н]

г) [)] [В/Н] [/] [4] [-] [.] [7]

- набор команды PRINT (5+3)*(8-6)/4-.7

д) [ВК] - выполнение команды PRINT, результат на экране 3.3

2. LETA=(5+3)*(8-6)/4-.7 , PRINTA

Набор клавиш: (В/Н-ВР)

а) [F] [L] LET

б) [A] [В/Н] [=] A=

в) нажать клавиши п п. б), в), г) первого варианта

г) [ВК] - выполнение команды LET

д) [F] [P] [A] PRINTA

е) [ВК] - выполнение команды PRINT, результат на экране 3.3.

Отличие вариантов не только в числе нажимаемых клавиш.

В первом случае, отобразив результат, ПЭВМ тут же о нем "забывает".

Во втором - результат хранится в ячейке А, может использоваться в последующих операциях до момента выключения ПЭВМ, или записи в эту ячейку нового числа (старое "стирается").

Например, во втором случае можно выполнить третью команду:

LETA=A-1 и отобразить результат: PRINTA (2.3)

Выражение A=A-1 "понимается" ПЭВМ так:

Из содержимого ячейки A вычти единицу и результат запиши в эту ячейку, "стирая" старое значение.

Сделаем выводы.

I. При вычислении выражений необходимо их математическую запись преобразовывать в "вычислительный" для ПЭВМ вид.

2. Простые выражения можно вычислять с помощью команды PRINT.

3. Сложные выражения лучше разбивать на ряд простых и для вычисления использовать команды LET . Промежуточные и конечные результаты вычислений отображать командой PRINT .

При этом можно выводить на экран сразу несколько результатов: PRINT A , B , C отображаются результаты, находящиеся в ячейках A , B , C .

Имена отображаемых ячеек в команде PRINT разделяются запятой.

2.3. Вычисление функций

В состав счетных операций ПЭВМ включены вычисления значений десяти математических функций.

Команда вычислений составляется из имени функции (три прописные латинские буквы), за которым в круглых скобках указывается аргумент функции.

Например, $\sqrt{25}$ вычисляется по команде SQR (25) , где SQR - имя функции "корень квадратный".

Вычисление $\sqrt{25}$ можно реализовать командой PRINT SQR (25).

Набор клавиш (В/Н - ВР):

а) [F] [P] PRINT

б) [S] [Q] [R] SQR

в) [В/Н] [(] [В/Н] [2] [5] [В/Н] [)] (25)

г) [ВК] - выполнение команды, результат на экране 5.

Набор имени функции можно выполнить сокращенно:

б) [F] [4] SQR(- имя функции и левая скобка аргумента.

Вычисляются следующие функции:

Наименование	Обозначение	Команда
Синус X	$\sin x$	SIN (X)
Косинус X	$\cos x$	COS (X)
Арктангенс X	$arctg\, x$	ATN (X)
Экспонента X	$exp\, x$	EXP (X)
Натуральный логарифм X	$ln\, x$	LOG (X)
Корень квадратный из X	$\sqrt{x}$	SQR (X)
Абсолютное значение X (модуль)	$\lvert x \rvert$	ABS (X)
Целая часть X	$int\, x$	INT (X)
Сигнум - функция (знак) X	$sign\, x$	SGN (X)
Датчик случайных чисел X	$rnd\, x$	RND (X)

Аргумент функций $\sin x$, $\cos x$ задается в радианах. Аргумент функции $rnd\ x$ – любое число.

Значение функции $rnd\ x$ – случайные числа, равномерно распределенные в интервале (0,1).

Значение функции $arctg\ x$ вычисляется в пределах от $-\pi/2$ до $\pi/2$.

Значения функций $\sin x, \cos x$, $arctg\ x$, $\sqrt{x}$, $\exp x$, $\ln x$ вычисляются с точностью до 10^{-8}.

Используя клавишу [F] и одну из цифровых клавиш, можно выполнять сокращенный набор имен функций.

Упражнения

Выполнить вычисления:

1. 3.8133 + 1.6785 – 2.915 + 7.236 – 1.2157

2. $\frac{(6-9)\cdot 7+48}{2} \cdot 3$

3. $\cfrac{25}{4+\cfrac{5}{3+\cfrac{7}{8-\frac{11}{2}}}}$

4. $16.4^{3} + 5.1^{2} - (3.14)^{-2.71}$

Задана арифметическая прогрессия:

$a_1 = 4, \quad d = 3$

Найти первые десять членов и их сумму .

Вычислить значение функций:

1. $y = \frac{1}{\sqrt{2\pi}} \cdot e^{-\frac{x^2}{2}}$, $x = 0.1,\ 1,\ 10$

2. $y = arctg\ \frac{\pi + x}{2} + arcctg\ \frac{\pi - 2}{2}$, $x = \frac{\pi}{6}$, $\frac{\pi}{4}$.

3. $y = \frac{\cos^2(1+2x) + \sqrt{3x^2 + \sqrt{x}}}{\ln(2x+1) - 2e^{\sqrt{x}}}$, $x = 11.2$

4. $y = \frac{|x - 0.5| + \sin(x - 0.5)}{int\ (1/x)}$, x – случайное число из интервала (0,1)

3. ГРАФИЧЕСКИЕ ВОЗМОЖНОСТИ ЭВМ

Видеодисплей ПЭВМ графического типа – его экран представляет матрицу независимо управляемых точек. Это позволяет выполнять построение мозаичных изображений, графиков, диаграмм и т.д.

В язык ПЭВМ включены специальные команды, облегчающие построение графических изображений.

3.I. Разметка экрана

Экран имеет размеры I20 мм х 64 мм, разделен на горизонтальные и вертикальные полосы (линии) толщиной I мм.

Число вертикальных линий (вертикалей) – I20, горизонтальных (горизонталей) – 64.

Пересечение вертикали и горизонтали образует квадрат со стороной I мм – точку экрана.

Точка – неделимый, независимо управляемый графический элемент, основа компьютерной графики.

На вертикали располагается, образуя ее, 64 точки, на горизонтали – I20 точек, а весь экран – это матрица 64 х I20 = 7680 точек.

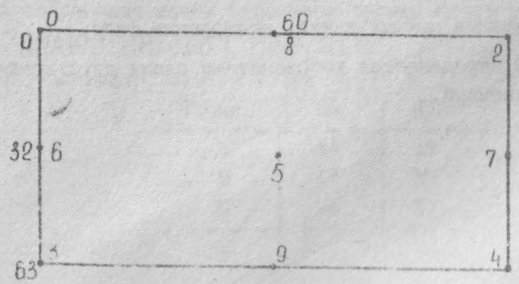
3.2. Отображение точек

Вертикали пронумерованы слева направо от 0 до II9, горизонтали – сверху вниз от 0 до 63. Любая точка экрана определяется номером ее вертикали (НВ) и горизонтали (НГ).

НВ, НГ – координаты точки.

Например, точка на пересечении первой вертикали и первой горизонтали имеет координаты НВ = 0 , НГ = 0 . Она расположена в левом верхнем углу экрана и является началом координат.

Характерные точки экрана показаны на рисунке, а их координаты приведены в таблице.



№	Характерные точки экрана	НВ	НГ
I	Верхняя левая	0	0
2.	Верхняя правая	II9	0
3	Нижняя левая	0	63
4	Нижняя правая	II9	63
5	Центральная	60	32
6	Средняя первой вертикали	0	32
7	Средняя последней вертикали	II9	32
8	Средняя первой горизонтали	60	0
9	Средняя последней горизонтали	60	63

Команда на отображение точки: DRAW Н НВ, НГ. DRAWH - имя команды, НВ , НГ - координаты точки. Например, DRAWH 60, 32- отображение точки в центре экрана.

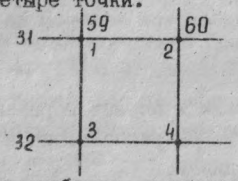
Набор клавиш: (В/Н - ВР, Р/Л - ЛАТ)

а) [F] [B] [H] набор DRAWH

б) [6] [0] [,] [3] [2] - координаты точки, разделяемые запятой

в) [ВК] - выполнение команды.

Центр здесь понимается условно, поскольку фактически центральными являются четыре точки:



1. НВ = 59, НГ = 3I
2. НВ = 60, НГ = 3I
3. НВ = 59, НГ = 32
4. НВ = 60, НГ = 32

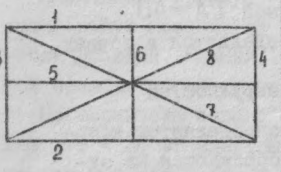
Примем для удобства, в качестве центральной, точку с координатами НВ = 60, НГ = 32. Вертикаль 60 будем считать средней вертикалью, а горизонталь 32 - средней горизонталью.

3.3. Вычерчивание прямых линий, построение фигур

Прямые линии на экране определяются координатами своих двух крайних точек: начальной и конечной.

Начальная точка из двух выбирается произвольно, самим пользователем.

Характерные прямые линии экрана показаны на рисунке, координаты их крайних точек приведены в таблице



N п/п	Характерные прямые линии экрана	координаты			
		НВ1	НГ1	НВ2	НГ2
1	Первая горизонталь	0	0	119	0
2	Последняя горизонталь	0	63	119	63
3	Первая вертикаль	0	0	0	63
4	Последняя вертикаль	119	0	119	63
5	Средняя горизонталь	0	32	119	32
6	Средняя вертикаль	60	0	60	63
7	Диагональ \	0	0	119	63
8	Диагональ /	119	0	0	63

Команда на вычерчивание прямой

DRAWD НВ1, НГ1, НВ2, НГ2.

Например, DRAWD 0,32,119,32 – вычерчивание средней горизонтали.

Набор клавиш: (В/Н – ВР, Р/Л – ЛАТ)

а) [F] [B] [D]

б) [0] [,] [3] [2] [,] [1] [1] [9] [,] [3] [2]

в) [ВК] – выполнение команды.

"Стирание" (гашение) прямой линии выполняется по команде:

DRAW E НВ1, НГ1, НВ2, НГ2.

Этой командой можно "стереть" (погасить) точку: DRAWE НВ,НГ,НВ,НГ. считая, что точка – прямая нулевой длины.

Используя координаты нескольких точек (НВ1, НГ1), (НВ2,НГ2), ..., ($НВ_К$, $НГ_К$), можно соединить их прямыми линиями, построить фигуру. Команда соединения точек отрезками прямых имеет вид:

DRAWD НВ1, НГ1, НВ2, НГ2, ..., $НВ_К$, $НГ_К$.

Пусть, например, заданы точки A, B, C :

Точки	НВ	НГ
A	44	13
B	44	37
C	76	37

Прямые, соединяющие точки, определяются координатами:

прямая	НВ1	НГ1	НВ2	НГ2
АВ	44	13	44	37
ВС	44	37	76	37
СА	76	37	44	13

При вычерчивании прямые образуют прямоугольный треугольник АВС с катетами АВ=24(37-13); ВС=32(76-44)и гипотенузой СА=40 . Длина сторон треугольника, как и вообще прямых, определяется в точках.

3.4. Вычерчивание кривых линий, построение окружностей

Вычерчивание кривых линий выполняется поточечно: определяются координаты каждой точки, затем точки последовательно отображаются на экране.

Окружности чертить проще. Нужно определить координаты центра окружности НВ , НГ и размер (в точках) радиуса R . Команда вычерчивания:

DRAWC НВ, НГ, R

Например, DRAWC 60, 32, 31 - вычерчивание окружности с центром НВ=60, НГ=32 (центр экрана) и радиусом R=31. Это значение радиуса является максимальным (31=63-32). Минимальное значение радиуса окружности R=1 .

Набор клавиш (В/Н-ВР, Р/Л-ЛАТ):

а) [F] [B] [C] - набор DRAWC

б) [6] [0] [,] [3] [2] [,] [3] [1] - данные окружности

в) [ВК] - выполнение команды

Полный перечень "графических" команд ПЭВМ приведен в Руководстве по эксплуатации.

Упражнения

1. Отобразить характерные точки экрана, используя приведенные в таблице их координаты.

2. Отобразить характерные прямые линии экрана, используя приведенные в таблице координаты их крайних точек.

3. В каждом углу экрана отобразить окружности радиуса R=10 , так чтобы они касались угловых линий.

4. Отобразить параболу $y = x^2$, расположив оси координат так:
   - ось OY - средняя вертикаль
   - ось OX - нижняя горизонталь

4. ОТОБРАЖЕНИЕ СИМВОЛЬНОЙ ИНФОРМАЦИИ

Символы: буквы, цифры, знаки-отображаются на экране дисплея набором точек. Матрица 64 x I20 точек дисплея при этом преобразуется в матрицу позиций.

Позиция – это прямоугольник регулируемой высоты и ширины, составленный из точек. Она является неделимым, независимо управляемым элементом отображения символьной информации.

Наименьшие размеры – 8 точек по высоте, 6 – по ширине определяют стандартную позицию из 8 x 6 = 48 точек. Символ формируется из 7 x 5 точек позиции. Нижний (8-й) ряд точек и правый (6-й) столбец стандартной позиции используются для разделения символов.

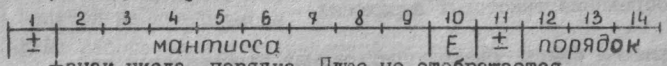
По вертикали экрана располагается 64 : 8 = 8, а по горизонтали I20 : 6 = 20 стандартных позиций. Их общее число 8 x 20 = I60. Горизонтальный ряд позиций называется строкой.

Символы отображаются на экране по команде PRINT .

4.I. Вывод данных на экран

Численные данные выводятся на экран как результаты расчетов, при проверке ввода исходных данных и в других случаях.

Общий вид отображения числа:



±знак числа, порядка. Плюс не отображается.

Мантисса числа – до семи цифр и десятичная точка, разделяющая целую и дробную части числа.

E – указатель десятичного порядка числа .

Порядок – значение порядка числа.

Отдельные элементы числа могут отсутствовать, например, порядок при отображении числа с фиксированной запятой.

Примеры: I) PRINT I – во второй позиции строки отображается число I. В первой позиции знак плюс не отображается.

2) PRINT PI , начиная со второй позиции строки отображается константа $\pi = 3.141593$. В третьей позиции находится десятичная точка, разделяющая целую и дробную части константы.

3) PRINT -.1234567E-980 . Через I0 секунд после набора команды отображается с первой позиции число – $0.1234567\ I0^{-980}$, занимая I4 позиций в строке.

Перечислив несколько чисел в команде PRINT , можно отобразить их на экране. Интервалы между отображенными числами определяются ви-

дом разделителя в перечислении.

Если разделитель – точка с запятой (;), то интервал составляет одну позицию.

Например, PRINT 1;-2 – отображаются два числа 1, -2 , разделяемые одной пустой позицией. Этот интервал является минимальным.

Второй разделитель – запятая расширяет интервал между числами до трех позиций. Например, команда PRINT 1,-2 устанавливает три позиции между числами 1, -2 .

При перечислении чисел в PRINT можно использовать оба разделителя, повторять запятую.

Примеры:

1) PRINT 1,-2;-3 – первый интервал три позиции, второй – одна.

2) PRINT 1,-2,,-3 – первый интервал три позиции, второй – пять.

Каждая дополнительная запятая увеличивает интервал между числами на две позиции.

При отображении положительных чисел интервалы между ними увеличиваются на одну (знаковую) позицию.

4.2. Отображение текстовых выражений

Текстовое выражение – буква, слово, предложение отображается по команде PRINT . Само выражение заключается в кавычки

Пример: PRINT "BASIC" (В/Н-ВР, Р/Л-ЛАТ)

1) [F] [P] PRINT .

2) [В/Н] [Ъ] [”] – открывающаяся кавычка

3) [В/Н] [В] [А] [S] [I] [С]

4) [В/Н] [Ъ] [”] – закрывающаяся кавычка

5) [ВК] – выполнение команды.

Нажатие клавиши [В/Н] в п.3) обеспечивает набор прописных букв. Если ее не использовать, набираемые буквы будут строчными: basic . Строчные буквы отображаются уменьшенным числом точек.

В качестве разделителя слов в предложении используется клавиша "Пробел": [ ] . Однократное ее нажатие устанавливает интервал в одну позицию, повторное – увеличивает его на одну позицию.

Например, PRINT "A␣␣B" , где ␣ – знак пробела шириной в одну позицию.

Набор команды: (В/Н - ВР, Р/Л - ЛАТ):

1) [F] [P] PRINT
2) [В/Н] [,,] ,,
3) [В/Н] [A] A
4) [ ] [ ] - два пробела
5) [B] B
6) [В/Н] [,,] ,,
7) [ВК] - выполнение команды

Отображаются буквы A и B с интервалом в две позиции.

Большое предложение при отображении размещается в нескольких строках по 20 символов в строке, включая пробелы.

В выражении кроме букв можно использовать цифры, знаки (кроме кавычек), а также совместно отображать выражения и числа.

Пример: A = - 1 можно отобразить командой:

PRINT "A = " ; - 1 где "A=" - выражение, - 1 -число.

Разделитель-точка с запятой-отображает выражение слитно, а разделитель-запятая - с интервалом в две позиции, увеличивая его при повторении запятой на две позиции.

Кроме рассмотренной стандартной формы информацию на экран можно отображать и в специальных формах, используя спецификации оператора PRINT , приведенные в Руководстве по эксплуатации ПЭВМ.

Упражнения

1. Отобразите самое маленькое и самое большое(для ПЭВМ) числа.
2. Отобразите десять цифр 0, 1, ..., 9:
   а) в две строчки:
      0 1 2 3 4
      5 6 7 8 9
   б) в два столбца:
      0 5
      1 6
      2 7
      3 8
      4 9
3. Отобразите Вашу фамилию, имя, отчество в виде: Фамилия
   Имя
   Отчество

5. ВЫЧИСЛЕНИЯ ПО ПРОГРАММЕ

Команды в режиме непосредственных вычислений выполняются немедленно после их ввода. Вместо этого можно составить из них программу, которая будет выполняться в нужный момент времени.

Такой способ работы называется программным режимом вычислений. Его основная особенность - возможность многократно выполнять команды, не набирая их повторно.

В программном режиме команды называются операторами. Описание всех операторов Бейсика для данной ПЭВМ приведено в Руководстве по эксплуатации ПЭВМ.

При записи операторы программы могут указываться отдельной строкой либо объединяться полем строки. Характерной чертой программы является нумерация ее строк.

Номером строки может быть любое целое число от I до 8I9I; оно ставится перед первым оператором. Разделитель между номером и оператором (пробел) не обязателен, разделитель операторов в строке - двоеточие (:). В строке программы может быть до 80 символов, включая номер и разделители.

Вычисления по программе - комплексная работа, предусматривающая организацию взаимодействия пользователя и ЭВМ.

Ее основные этапы:

- составление программы;
- ввод и редактирование программы;
- отладка программы;
- счет по программе.

5.I. Составление программы

Составление программы - основной этап вычислений с использованием программ. Он начинается с анализа вычислительного алгоритма с целью его упрощения, сокращения числа выполняемых операций.

Для этого необходимо:

I) по возможности упростить рабочий алгоритм, привести его к вычислительному для ЭВМ виду;
2) выделить в алгоритме повторяющиеся выражения, предусмотрев их однократное вычисление;
3) определить порядок выполнения операций;
4) установить тип используемых в алгоритме величин (массивы, простые переменные).

5) определить форму ввода исходных данных и вывода результатов вычислений;

После этого выполняется запись программы, заканчивающаяся указанием оператора END - конец программы. Для записи программы удобно использовать специальные бланки программы.

В бланке программы должно быть указано название программы, записана последовательность пронумерованных строк, даны краткие пояснения алгоритма - комментарии программы.

Для учебных целей лучше использовать бланки следующего вида:

ЭЛЕКТРОНИКА МК 90

Программа: Лист:

Листов:

Строка	Оператор	Выражение

Бланк имеет три трафаретные колонки и рассчитан на запись 22 строк программы (один лист). Вверху над трафаретом записывается наименование программы, номер листа и общее количество листов заданной программы. В первой колонке трафарета указывается номер строки программы. В колонке "Оператор" записывается оператор Бейсика. Разметка колонки выполнена под оператор RESTORE (7 позиций). В последней колонке указываются вычислительные или текстовые выражения, комментарии программы. Колонка размечена на 20 позиций - строка символов на экране.

Заполнение бланка лучше производить карандашом - легче исправляются ошибки, убираются помарки. Применение бланков улучшает наглядность записи программы, уменьшает число возможных ошибок, повышает доступность программы для других пользователей.

В качестве примера составим программу для вычисления выражения:

$$y = tg(x^2 - 1) + \frac{\sqrt{x^4 - 1}}{\sqrt{1 + x^2}}.$$

Анализ выражения показывает, что:

1) его можно упростить, так как $\sqrt{x^4-1} = \sqrt{x^2-1}\cdot\sqrt{x^2+1}$

$$y = tg(x^2-1)+\sqrt{x^2-1}$$

затем привести к вычислительному для ПЭВМ виду, поскольку функции "тангенс" нет в составе ее счетных операций.

$$y = sin(x^2-1)/cos(x^2-1) + \sqrt{x^2-1}\ ;$$

2) выделим повторяющееся выражение:

$$x^2-1\ ;$$

3) порядок выполнения операций определим так:

а) $x^2-1=z$ б) $\sqrt{z}=B$

б) $sin(z)/cos(z)=A$ г) $y=A+B$;

4) тип используемых величин

y, x, z, A, B – простые переменные

y – результат

x – исходное данное

z, A, B – промежуточные результаты ;

5) исходное данное – X может иметь несколько значений. Это естественное предположение реализуется последовательным вводом значения X (оператор INPUT) и, соответственно, выводом результата y.

Запишем программу на бланке

ЭЛЕКТРОНИКА МК90

Программа: $y=tg(x^2-1)+\sqrt{x^2-1}$ Лист: 1

Листов: 1

Строка	Оператор	Выражение
,5	INPUT	X
1,0	LET Z=	X↑2-1
1,5	LET A=	SIN(Z)/COS(Z)
2,0	LET B=	SQR(Z)
2,5	LET Y=	A+B
3,0	PRINT	Y
3,5	END	

5.2. Ввод и редактирование программы

Ввод программы, т.е. запись в оперативную память ЭВМ выполняется последовательным набором ее строк.

Вначале набирается номер строки, затем оператор и следующее за ним выражение. Нажатием клавиши [ВК] производится ввод строки в ОЗУ.

Набирать и вводить строки программы можно в любой последовательности - в ОЗУ они располагаются согласно нумерации.

При вводе программы можно с помощью команды AUTO установить режим автоматической нумерации строк.

Команда AUTO A, B присваивает набираемым строкам номера, начиная с числа A и далее по возрастанию с шагом B: AUTO 5,5 - строки нумеруются начиная с 5(A), увеличивая следующий номер на 5(B): 5, 10, 15, ...

Числа A, B могут не указываться:

AUTO - стандартная нумерация: 10, 20, 30, ...

AUTO 5 - нумерация с шагом 10: 5, 15, 25, ...

AUTO,5 - нумерация от 10 с шагом 5: 10, 15, 20, ...

Пример: Выполнить ввод программы предыдущего примера. В качестве "исходных данных" имеем текст программы на бланке.

Выполняем ввод программы:

Включаем ЭВМ: 2 [↑] [ВК]

Устанавливаем режим автоматической нумерации строк:

[F] [A] AUTO

[5] [,] [5] [ВК] - нумерация типа: 5, 10, 15, 20, ...

Вводим программу по строкам:

[F] [4] [1] [X] [ВК] - ввод первой строки

- - - - - - - - - - - - -

[F] [E] [ВК] - ввод последней строки .

Отменяем режим нумерации:

[7] [СЧ] [4] [Р] [ВК] - сообщение на экране: -

подтверждает отмену.

Ввод программы выполнен.

Редактирование программы, т.е. устранение ошибок, допущенных при вводе, начинается с просмотра программы. Просмотр (чтение) программы выполняется с помощью команды LIST.

Команда LIST A , B отображает на экран программу, начиная со строки с номером A до строки с номером B . После отображения первых восьми строк, точнее заполнения восьми строк экрана, вывод приостанавливается и возобновляется нажатием клавиши [ВК] Каждое ее нажатие выводит на экран одну строку программы. Заканчивается просмотр программы дополнительным нажатием клавиши [ВК] (сообщение "Готов").

Числа A и B в команде LIST могут отсутствовать.

Команда LIST A отображает строку программы с номером A .

LIST,B - отображается программа, начиная с первой строки до строки с номером B .

LIST - отображается вся программа.

После просмотра строки возможные ошибки исправляются ее повторным набором. Ненужные (лишние) строки можно исключить из программы, используя команду DELETE . Ее общая форма:

DELETE A,B - исключаются строки программы, начиная с номера A и до номера B включительно.

Сокращенные формы:

DELETE A - исключается строка с номером A ;

DELETE,B - исключаются строки от начала программы до строки с номером B .

Перед вводом программы полезно выполнить команду DELETE 1, 8191, очищая ОЗУ.

В качестве примера отобразим на экран введенную ранее программу.

1. Отобразим первую строку:

[F] [M] [5] [ВК] - набор команды LIST 5 и ее выполнение.

Отображение на экране:

```
5 INPUT X
Готов
_
```

2. Отобразим две последние строки:

[F] [M] [3] [0] [,] [3] [5] [ВК]

Информация на экране:

```
30 PRINT Y
35 END
Готов
_
```

3. Исключим последнюю строку программы:

[F] [→] [3] [5] [ВК] – набор команды DELETE 35и ее выполнение.

Сообщение на экране: Готов

4. Выполним повторно п.2. – сообщение на экране:

```
30 PRINT Y
Готов
```

подтверждает отсутствие в программе строки с номером 35.

5. Восстановим эту строку:

[3] [5] [F] [E] [ВК]

Информация на экране:

```
35 END
```

Просмотр и редактирование строки можно выполнить с помощью команды EDIT .

Команда EDIT A отображает на экране строку с номером A и устанавливает маркер (-) под первой цифрой номера. Используя клавиши 7[3←] и 7[6→] можно подвести маркер под любой символ строки, нажатием клавиши [ЗБ] стереть его и затем набрать нужный символ. Ускоряет перемещение маркера по строке нажатие клавиш:

[СУ] [A] – маркер устанавливается под первой цифрой номера строки

[СУ] [X] – маркер устанавливается за последним символом строки

[СУ] [B] – маркер перемещается на начало предыдущего, относительно текущего, слова строки

[СУ] 5[5F] – маркер перемещается на начало последующего, относительно текущего, слова строки.

Ускоренное стирание символов строки выполняется нажатием клавиш:

[СУ] [E] – удаляются все символы строки, расположенные правее маркера.

После исправления строка записывается в ОЗУ нажатием клавиши [ВК]. Действие команды EDIT отменяется нажатием клавиш [СУ] [P].

Пример: Отредактируем первую строку программы:

I. [F] [V] [5] [ВК] – набор и выполнение команды EDIT5.

Информация на экране:

5 INPUT X , маркер под цифрой 5, т.е. в начале строки.

2.Переместим маркер в конец строки:

[СЧ] [X] - маркер занял позицию за последним символом:

5 INPUT X _

3. Нажав клавишу [←] подведем маркер к символу X (справа):

5 INPUT X_

4. Нажмем клавишу [ЗБ] - символ X исчез.

5. Восстановим символ, нажав клавишу: [X] .

6. Введем строку в ОЗУ, нажав клавишу: [ВК] - редактируемая строка останется на экране, маркер перемещается в начало следующей строки.

Команда EDIT не выполняет "редакторские" функции при исправлении русского текста.

5.3. Ввод исходных данных. Отладка программы. Счет по программе.

Исходные данные, обеспечивающие получение результатов вычислений, входят в программу в виде констант или переменных. Набор константы означает и ввод ее в ОЗУ. Для переменных их численные значения задаются тремя способами:

1. С помощью оператора INPUT
   Например: 5 INPUT X
2. С помощью оператора LET
   Например: 5 LET X = 5
3. С помощью операторов READ, DATA :

```
5 READ X
7 DATA 5
```

Не рассматривая подробно особенностей способов, отметим, что ввод данных с использованием оператора INPUT предпочтительнее при выполнении вариантных расчетов. Два других способа в таких случаях требуют изменения текста программы.

Отладка программы, т.е. решение тестовых примеров с известными результатами может, по необходимости, производится и для отдельных ее частей.

Если тестируется вся программа, то она запускается на счет коман-

дой PUN

Результаты вычислений отображаются на экране в процессе работы программы.

Окончанием счета служит сообщение на экране:

ОСТ В СТРОКЕ N
Готов

где $\overline{N}$ – номер последней строки программы.

Пример: Выполним тестирование программы п.5.2.

Известно, что при $X=2$ $y=1.59$

Порядок действий:

1. Вводим программу.
2. Включаем счет: [F] [R] [ВК] – набор команды RUN и выполнение .
3. По запросу программы (?) вводим значение $X=2$: [2] и запускаем программу на продолжение счета [ВК] .
4. Считываем результат $y=1.59$.

Если результаты счета не совпадают с ответом, проверяются промежуточные результаты. Их значения можно отобразить на экране командой PRINT .

В нашем примере возможен пункт:

5. Отображаем промежуточные результаты Z, A, B :

[F] [P] [Z] [,] [A] [,] [B] [ВК]

$Z=3$, $A=-0.14$, $B=1.73$.

Отметим, что разделение алгоритма на отдельные операции, облегчающие составление программы, упрощает и ее отладку, ускоряет поиск ошибок, используя значения промежуточных результатов.

Отладка программы по частям выполняется с использованием операторов STOP и GOTO .

Оператор STOP останавливает вычисление в любом месте программы. Его включают в программу при составлении или в процессе отладки.

Например, дополним программу п.5.2. строкой

12 STOP набор клавиш: [1] [2] [F] [U] [ВК] .

При выполнении программы произойдет остановка после выполнения первых двух строк. Проверим значение Z , как указано выше:

[F] [P] [Z]

Информация на экране: 3 .

Операции проверки и остановки можно объединить:

```
12 PRINT Z:STOP.
```

После выполнения части программы (строки 5, I0, I2) продолжение счета обеспечивает команда GOTO N , где N - номер строки, с которой необходимо продолжить вычисления.

В нашем примере, набрав после остановки команду

[F] [G] набор GOTO
[1] [5] [ВК]

продолжаем вычисления с строки I5 до получения результата.

После выполнения отладки дополнительные операторы STOP, PRINT исключаются из программы с помощью команды DELETE .

Запуск программы на счет выполняется командой RUN . Необходимо приблизительно оценить время выполнения программы, поскольку возможны случаи зацикливания, т.е. безостановочного выполнения программы. При выполнении программы могут появляться сообщения об ошибках счета, коды ошибок приведены в Руководстве по эксплуатации. Ошибки необходимо исправить и вновь включить счет.

Остановка счета на любом шаге выполняется нажатием клавиш [СУ] [Р] . При выключении ЭВМ программа в ОЗУ не сохраняется.

5.4. Сменный модуль памяти. Запись и считывание файлов

Сменные модули памяти (СМП ОЗУ) емкостью I0 Кбайт используются для хранения файлов.

Два модуля: СМП0, СМП1 подключаются к ПЭВМ в блоке памяти, причем один модуль - рабочий, второй - резервный. Рабочий модуль участвует в обмене информацией с ОЗУ, резервный - хранит информацию.

Рабочим модуль объявляется по команде DEV :

```
DEV"SM0:"   -СМП0
DEV"SM1:"   -СМП1
```

После названия команды указывается имя модуля и двоеточие (:), заключаемые в кавычки.

Если имя модуля не указано: DEV , то рабочим объявляется СМП0 . При включении ПЭВМ рабочим модулем также является СМП0.

Информация в модуле хранится в виде файлов.

Файл - упорядоченный поименованный набор однотипных записей, хранимый на внешних запоминающих устройствах.

Именование файла облегчает поиск и обращение к нему среди других

файлов, а упорядочение (нумерация) записей в файле играет ту же роль для отдельной записи файла.

По виду хранимой информации файлы классифицируются на программные и информационные.

Программные файлы содержат строки (операторы) программы.

Информационные файлы - наборы фактографических или числовых данных.

В данной ПЭВМ используются программные файлы, обмен которыми между ОЗУ и СМП означает:

- запись в СМП файла, находящегося в ОЗУ.
- считывание в ОЗУ файла, находящегося в СМП.

Операции выполняются копированием файла и передачей копии от одного устройства памяти к другому, общее название операций - загрузка файла.

Если файл копированием из ОЗУ загружается в СМП (запись файла), то этим обеспечивается его сохранение и последующее использование.

Запись файла в СМП выполняется по команде SAVE (хранить). Очевидно, в команде должно быть указано, что хранить (имя файла) и где хранить (имя СМП).

Полный вид команды:

SAVE " <имя модуля> <имя файла> "

Выражение в кавычках называется спецификацией файла. Первым в спецификации указывается имя модуля - SM0 : или SM1 :. Имя модуля может отсутствовать - файл запишется в рабочий модуль. В конце имени файла может присутствовать указатель типа файла. Указатель состоит из точки, за которой следует не более трех символов.

Пример: SAVE " SM0 : GRAF . BAS " - запись в СМПО файла с именем GRAF . BAS, где . BAS - указатель типа файла. В данном случае - это программа на Бейсике.

Тип такого файла можно не указывать: SAVE" SM0: GRAF "

Можно не указывать номер СМП, если он рабочий:

SAVE " GRAF "

Записываемая программа размещается в СМП блоками по 512 байт каждый, время записи зависит от ее объема.

Минимальный объем размещения в СМП для Бейсик - программы - один блок, максимальный - 16 блоков.

Считывание файла из СМП в ОЗУ выполняется по команде LOAD (загрузить).

Полный вид команды:

LOAD " <имя модуля> <имя файла> "

Структура спецификации файла такая же, как в команде SAVE .

Пример: LOAD " SM0 : GRAF . BAS" – загрузка в ОЗУ файла GRAF . BAS находящегося в СМП0.

Краткая запись: LOAD "GRAF" означает загрузку Бейсик программы из рабочего СМП.

После загрузки программа по команде RUN может быть запущена на счет.

Операции загрузки и счета можно объединить в одной команде:

LOAD "<специф. файла>", R

добавляя после указания спецификации файла запятую и букву R (первую букву RUN).В этом случае после загрузки начнется выполнение программы.

В качестве примера выполним запись и считывание программы п.5.2 с использованием СМП0.

Выберем имя файла FY , имея в виду, что в программе вычисляются значения функции $y = f(x)$ После ввода программы в ОЗУ (см. п.5.2) выполним запись ее в СМП0:

1. Объявляем СМП0 рабочим : DEV "SM0:"

2. Выполняем запись: SAVE "FY"

Напоминаем, п.1 не является обязательным.

Запись произведена копированием и программа (оригинал) сохранится в ОЗУ.

Выключим ЭВМ и снова ее включим.

Загрузим программу FY из СМП0 в ОЗУ:

LOAD "FY"

Если нужна загрузка и счет, выполним команду:

LOAD "FY", R

Файлы можно удалять из СМП по команде:

KILL " <имя модуля> <имя файла> "

При выполнении команды из указанного модуля удаляется поименованный файл.

Например, KILL " SM0 : FY – удаляется файл FY находящийся в модуле СМП0.

Удаление всех файлов из модуля – чистка модуля – выполняется по команде:

INIT " <имя модуля> "

Если чистится рабочий модуль, имя не указывается: INIT

Напоминаем, после набора любой из перечисленных команд нажимается

клавиша [ВК] .

В СМП могут быть записаны несколько файлов, их имена составляют каталог модуля.

Каталог модуля отображается на экран командой:

FILES " <имя модуля> "

Для рабочего СМП имя не указывается.

После выполнения команды на экране появляется сообщение:

Справочник	<имя модуля>	
<имя файла>	L_1	N_1
<имя файла>	L_2	N_2
свободно	L	N

где L_i - длина файла в блоках по 512 байт, N_i - начальный адрес файла на СМП (номер первого блока).

Последовательно нажимая клавишу [ВК] можно просмотреть весь каталог. Заканчивается сообщение фразой:

A файлов B блоков

C свободных блоков

где A, B, C - количество файлов и блоков.

Упражнения

1. Составить программу вычисления значений функции:

$$y = \frac{x^3 \cdot \cos^2(3x-5) + arctg\sqrt{2x^3-1}}{tg(2x^3-1) + (3x-5)^2}$$

для значений x = 1.9, 1.1, 1.3, 1.5, 1.7.

2. Составить программу вычисления площади S треугольника, если заданы:

а) сторона a и прилежащие к ней углы B, C:

$$S = \frac{1}{2} a^2 \cdot \frac{\sin B \cdot \sin C}{\sin(B+C)} ;$$

б) стороны a, b и угол C между ними:

$$S = \frac{1}{2} a \cdot b \cdot \sin C ;$$

в) стороны a, b, c :

$$S = \sqrt{p(p-a)(p-b)(p-c)} , \quad p = \frac{1}{2}(a+b+c) .$$

Вычислить для трех случаев площадь равностороннего треугольника со стороной, равной 5.

3. Составить программу вычисления значений обратной тангенс-тригонометрической функции $arctg\ z$ комплексного аргумента $z = \alpha + j\beta$ ($j = \sqrt{-1}$ – мнимая единица)

$$arctg\ z = \frac{1}{2}\left(\pi - arctg\ \frac{1+\beta}{\alpha} - arctg\ \frac{1-\beta}{\alpha}\right) + \frac{j}{4}\ln\frac{(1+\beta)^2+\alpha^2}{(1-\beta)^2+\alpha^2}.$$

Вычислить $arctg\,(0.7 + j\,0.96)$.

4. Составить программу вычисления векторного произведения двух векторов $A(a_1, a_2, a_3)$, $B(b_1, b_2, b_3)$

$$A \times B = C(c_1, c_2, c_3),$$

где $c_1 = a_2 b_3 - a_3 b_2$,

$c_2 = a_3 b_1 - a_1 b_3$,

$c_3 = a_1 b_2 - a_2 b_1$.

Вычислить C при $A(7, 8, 9)$, $B(4, 5, 6)$.

5. Составить программу решения системы двух линейных алгебраических уравнений:

$$a_{11}x_1 + a_{12}x_2 = b_1;$$

$$a_{21}x_1 + a_{22}x_2 = b_2.$$

Алгоритм решения:

$$x_2 = \frac{a_{11}b_2 - a_{21}b_1}{a_{11}a_{22} - a_{21}a_{12}};$$

$$x_1 = \frac{1}{a_{11}}(b_1 - a_{12}x_2).$$

Записать программу в модуль СМП.

Решить систему:

$$5x_1 - 8x_2 = -7$$

$$2x_1 + 6x_2 = 9.$$

6. АЛГОРИТМЫ И ПРОГРАММЫ

Отвлекаясь от содержательного смысла алгоритмов и учитывая только структуру их исполнения, все многообразие численных алгоритмов, реализуемых на ЭВМ, можно подразделить на несколько основных типов:

- линейные алгоритмы
- блочные алгоритмы (блок-алгоритмы)
- разветвляющиеся алгоритмы
- циклические алгоритмы
- алгоритмы подпрограммного типа

Такое выделение типов алгоритмов отражает и специфику их программной реализации, ориентированную на языковые средства ЭВМ, в данном случае программные средства Бейсика.

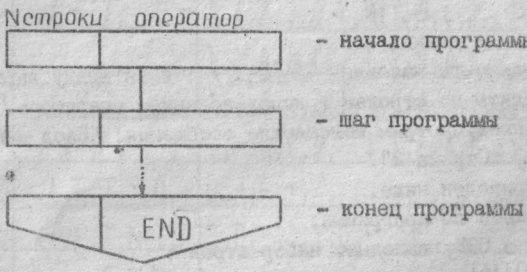
Поэтому приведенный в этой главе материал дает описание основных свойств указанных типов алгоритмов и иллюстрирует технику составления программ для ЭВМ "Электроника МК 90".

6.1. Линейный алгоритм

Линейный алгоритм обеспечивает решение задачи в порядке следования шагов алгоритма (операторов), выстраивая процесс вычислений "в линию".

Его программной реализацией является линейная программа.

Вычисления по линейной программе ЭВМ производит, переходя от одной строки программы к другой, и заканчивает оператором END по схеме:



Линейная программа сокращает время расчета в тех случаях, когда производится серия вычислений с разными исходными данными.

Приведенная в п.5.2. программа является линейной.

Пример: Составить программу вычисления определителя третьего порядка:

$$D = \begin{vmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ a_{31} & a_{32} & a_{33} \end{vmatrix} = C + E + F ,$$

$$C = a_{31}(a_{12}a_{23} - a_{22}a_{13}) ,$$

$$E = a_{32}(a_{21}a_{13} - a_{11}a_{23}) ,$$

$$F = a_{33}(a_{11}a_{22} - a_{21}a_{12}) ,$$

где a_{ij} - элементы определителя (заданные числа)

Определим порядок вычислений:

1) C

2) E

3) F

4) $C + E + F = D$

Распределим память для используемых в алгоритме численных величин:

1) исходные данные - элементы определителя образуют таблицу из трех строк и трех столбцов - двумерный массив A .

Чтобы использовать массив в вычислениях нужно зарезервировать для него место в памяти.

В языке Бейсик это делается с помощью оператора DIM A(3,3) ;

2) промежуточные и конечный результаты вычислений определим как простые переменные: C , E , F , D .

Определим порядок ввода данных:

Исходные данные - элементы массива $A(3,3)$, имея ввиду вариантные расчеты, будем вводить по строкам с использованием оператора INPUT.

Для контроля ввода предусмотрим поясняющие сообщения: "Ввод данных", "Строка 1", "Строка 2", "Строка 3".

Текст программы - приведен ниже.

Определим порядок счета по программе:

1) ввести программу в ОЗУ, выполнив набор строк ;

2) включить счет: RUN ;

3) последовательно, по запросу (?) ввести строки элементов (три числа, разделяемые запятой) ;

4) считать результат D .

Тестовый пример: $\begin{vmatrix} 1 & 4 & -1 \\ 5 & 0 & 3 \\ 6 & 9 & -7 \end{vmatrix} = 140$

Программа : Определитель - 3

Лист : 1

Листов : 1

Строка	Оператор	Выражение
15	DIM	A(3,3)
10	PRINT	"Вводим данные"
15	PRINT	"Строка 1"
20	INPUT	A(1,1),A(1,2),A(1,3)
25	PRINT	"Строка 2"
30	INPUT	A(2,1),A(2,2),A(2,3)
35	PRINT	"Строка 3"
40	INPUT	A(3,1),A(3,2),A(3,3)
45	LET	C=A(3,1)*(A(1,2)*A(2,3)-A(2,2)*A(1,3))
50	LET	E=A(3,2)*(A(2,1)*A(1,3)-A(1,1)*A(2,3))
55	LET	F=A(3,3)*(A(1,1)*A(2,2)-A(2,1)*A(1,2))
60	LET	D=C+E+F
65	PRINT	"D=";D
70	END	

Последовательность вычисления примера:

- ввести элементы первой строки:
  сообщение "Строка I",
  ответ на запрос - ? :

  1, 4, -1 [ВК]

- ввести элементы второй строки:
  сообщение "Строка 2",
  ответ на запрос - ?

  5, 0, 3 [ВК]

- ввести элементы третьей строки:
  сообщение "Строка 3"
  ответ на запрос - ?

  6, 9, -7 [ВК]

- считать результат:

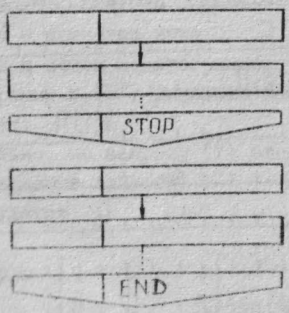
  D = 140

6.2. Блок-алгоритм

Блок-алгоритм объединяет несколько алгоритмов (блоков) в единую расчетную схему. В этом случае к исходным данным расчета добавляется указание порядка работы отдельных блоков программы.

Блок-алгоритму соответствует блок программа, т.е. программа, каждый блок которой является собственно программой. Блок имеет свой заголовок (начало), в конце его может ставиться команда STOP .

В программу блоки объединяются полем программной памяти ЭВМ по схеме:



- начало блока I
- шаг
- конец блока I
- начало блока 2
- шаг
- конец блока 2

Пример: Составить программу, выполняющую арифметические операции с комплексными числами.

Пусть даны два комплексных числа

$c_1 = a_1 + jb_1$, $c_2 = a_2 + jb_2$, j -мнимая единица .

Результат операций - комплексное число $c_3 = a_3 + jb_3$ вычисляется по формулам:

сложение: $c_3 = c_1 + c_2$; $a_3 = a_1 + a_2$, $b_3 = b_1 + b_2$

вычитание: $c_3 = c_1 - c_2$; $a_3 = a_1 - a_2$, $b_3 = b_1 - b_2$

умножение: $c_3 = c_1 \cdot c_2$; $a_3 = a_1 \cdot a_2 - b_1 \cdot b_2$

$b_3 = a_2 \cdot b_1 + a_1 \cdot b_2$

деление: $c_3 = \frac{c_1}{c_2}$; $r = a_2^2 + b_2^2$, $a_3 = (a_1 \cdot a_2 + b_1 \cdot b_2)/r$

$b_3 = (a_2 \cdot b_1 - a_1 \cdot b_2)/r$

Выделим в алгоритме отдельные блоки:

1) ввод исходных данных
2) сложение
3) вычитание
4) умножение
5) деление

Распределим память: исходные данные $a_1 b_1$; $a_2 b_2$ и результат a_3 , b_3 - простые переменные.

Порядок ввода данных: расчеты вариантные, ввод по INPUT .

Контроль ввода - выдача поясняющих сообщений.

Текст программы: приведен ниже .

Пояснение к программе. Программа позволяет выполнять арифметические операции с комплексными числами в любом порядке, превращая ЭВМ в "комплексное" вычислительное устройство.

При цепочном выполнении операций результат каждого вычисления может быть исходным данным для следующей операции, поэтому он (a_3, b_3) преобразуется в первое комплексное число (строка 40).

Первый блок программы (строки 5 ÷ 25) выполняет ввод данных. Следующие за ним блоки - вычислительные отмечены своими названиями. Порядок их включения определяется пользователем в зависимости от расчетной ситуации:

1) сложение - нажать клавиши: [F] [G] [5] [0] [ВК]
2) вычитание: [F] [G] [6] [0] [ВК]
3) умножение: [F] [G] [7] [5] [ВК]
4) деление: [F] [G] [9] [0] [ВК]

ЭЛЕКТРОНИКА МК 90

Программа : Комплексная арифметика

Лист : 1

Листов : 1

Строка	Оператор	Выражение
1,5	PRINT	"Ввод a1, b1"
1,0	INPUT	A, B
1,5	PRINT	"Ввод a2, b2"
2,0	INPUT	C, D
2,5	STOP	
3,0	PRINT	"Сложение"
3,5	LET	E = A + C : LET F = B + D
4,0	LET	E : LET B = F
4,5	PRINT	"a3 = " ; E
5,0	PRINT	"b3 = " ; F
5,5	STOP	
6,0	PRINT	"Вычитание"
6,5	LET	E = A - C : LET F = B - D
7,0	GOTO	4,0
7,5	PRINT	"Умножение"
8,0	LET	E = A * C - B * D : LET F = C * B + A * D
8,5	GOTO	4,0
9,0	PRINT	"Деление"
9,5	LET	H = C ↑ 2 + D ↑ 2
10,0	LET	E = (A * C + B * D) / H
10,5	LET	F = (C * B - A * D) / H
11,0	GOTO	4,0

Здесь [F] [G] – сокращенный набор команды GOTO , обеспечивающей запуск на счет с указанной строки программы.

Порядок счета по программе:

1) ввести программу
2) включить счет: RUN
3) ввести исходные данные: a_1, b_1 , затем a_2, b_2
4) включить вычислительный блок (см. выше)
5) считать результаты: a_3, b_3
6) при цепочном расчете:
   - нажать клавиши [F] [G] [1] [5] [ВК]
   - ввести данные: a_2, b_2 (ввод второго числа)
   - включить вычислительный блок;
7) для расчета с двумя данными перейти к п.2)

Тестовый пример: $\frac{3+j4}{7-j2} + (7.4-j5.6)\cdot(6.4+j8) - (2.6-j1.9) = 89.8 + j25.9.$

Последовательность вычисления примера:

- выполнить деление $\frac{3+j4}{7-j2}$ и записать результат $a_1=3,\ b_1=4,\ a_2=7,\ a_3=0.245,\ b_3=0.641$;
- выполнить умножение $(7.4-j5.6)\cdot(6.4+j8)$ $a_1=7.4,\ b_1=-5.6,\ a_2=6.4,\ b_2=8$;
- выполнить сложение результатов, предыдущих операций : $(a_2=0.245,\ b_2=0.641)$ – второе исходное данное ;
- выполнить вычитание: результат предыдущей операции – уменьшаемое, $(a_2=0.26,\ b_2=-1.9)$ – второе исходное данное ;
- считать результаты: $a_3=89.8,\ b_3=25.9.$

6.3. Разветвляющийся алгоритм

Разветвляющийся алгоритм составляется для задач, процесс вычисления в которых зависит от выполнения некоторых условий. Включение в состав разветвляющегося алгоритма условий изменяет естественный (линейный) ход выполнения задачи.

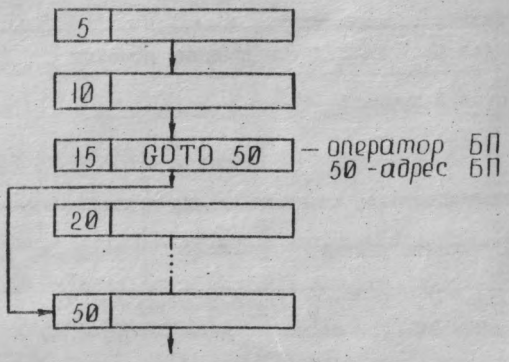
Исполнение алгоритма производится в этом случае с использованием переходов – безусловных и условных, определяющих порядок выполнения команд.

Для программной реализации переходов в состав языка Бейсик включены операторы безусловного и условного переходов. Они формируются из команды перехода и адреса перехода. Команда определяет тип перехода, адрес указывает номер строки программы, на которую требуется перейти.

Оператор безусловного перехода имеет вид: GOTO N , где N, $1 \leqslant N \leqslant 8191$ - номер строки.

Безусло́вный переход позволяет пропустить часть программы, не выполняя и не возвращаясь к ней, как это показано на схеме.

Схема "Безусловный переход"



Как пример, схема иллюстрирует программу, в которой после выполнения первых двух строк безусловным переходом управление передается на строку 50.

Условный переход передает управление в зависимости от результатов проверки некоторого условия. Простейшими условиями являются соотношения между числами A, B вида:

Соотношение:	Запись на Бейсике:
$A < B$	A < B
$A \leqslant B$	A < = B
$A \geqslant B$	A > = B
$A > B$	A > B
$A = B$	A = B
$A \neq B$	A < > B

Знаки соотношений набираются клавишами:

(В/Н - НР)

<	7 [4 <] :	[3]	[В/Н]	3<5 [<]	[В/Н]	[5]
>	7 [5 >] :	[5]	[В/Н]	5>3 [>]	[В/Н]	[3]
=	2 [8 =] :	[5]	[В/Н]	5=5 [=]	[В/Н]	[5]

Остальные соотношения формируются комбинированным набором указанных клавиш. Оператор условного перехода включает одно из соотношений в виде:

IF < соотношение > THEN < операторы >

Если соотношение истинно, то выполняются операторы (оператор), находящиеся в этой строке за THEN . При невыполнении условия управление передается на следующую строку программы.

Поясним процесс передачи управления при условном переходе схемой.

Схема "Условный переход"

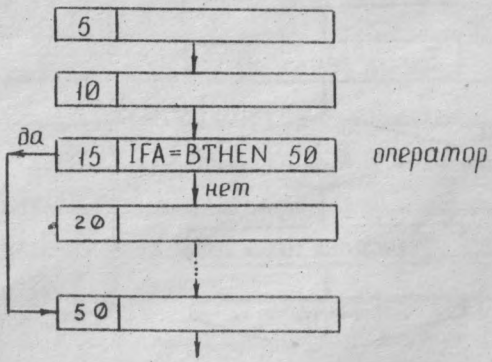


Схема иллюстрирует программу, в которой после выполнения двух строк проверяется условие A = B равенства двух переменных (арифметических выражений). Если условие выполняется, управление передается на строку 50, так как она (ее номер) указана в адресе перехода. При невыполнении условия управление передается на строку 20.

В качестве примера составим разветвляющуюся программу для вычисления корней квадратного уравнения:

$$aX^2 + bX + c = 0 \quad .$$

Как известно, корни уравнения вычисляются в зависимости от знака дискриминанта $D = b^2 - 4ac$. Если $D \geq 0$, корни X_1, X_2 – вещественные и вычисляются по формуле:

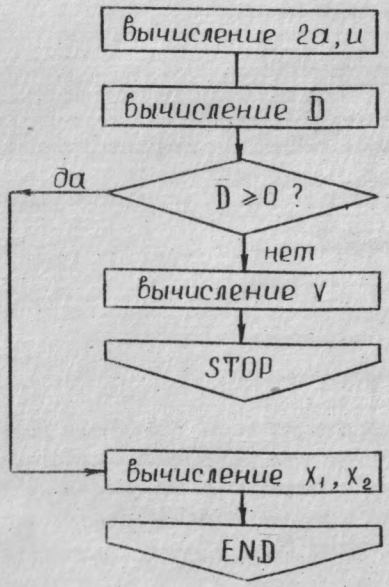
$$X_1, X_2 = \frac{-b \pm \sqrt{D}}{2a} \quad .$$

При $D < 0$ комплексные корни вычисляются в виде:

$$U \pm jV = -\frac{b}{2a} \pm j\frac{\sqrt{-D}}{2a} \quad , \quad j = \sqrt{-1} \quad .$$

Обозначим $U = -\frac{b}{2a}$ и выделим в расчетном алгоритме повторяющиеся выражения: $2a, U, D$.

Представим алгоритм в виде блок-схемы:



ЭЛЕКТРОНИКА МК 90

Программа : Квадратное уравнение

Лист : 1

Листов : 1

Строка	Оператор	Выражение
5	PRINT	"Вводи a, b, c"
10	INPUT	A, B, C
15	LET	E=2*A
20	LET	U=-B/E
25	LET	D=B↑2-4*A*C
30	IF	D>=0 THEN 60
35	LET	F=SQR(-D)/E
40	PRINT	"Корни мнимые."
45	PRINT	"U=";U
50	PRINT	"V=";F
55	STOP	
60	LET	F=SQR(D)/E
65	LET	E=U+F
70	LET	F=U-F
75	PRINT	"Корни вещ."
80	PRINT	"x1=";E
85	PRINT	"x2=";F
90	END	

Распределим память: исходные данные a, b, c и результаты D, u, v, x_1, x_2 – простые переменные.

Порядок ввода данных: ввод по INPUT

Текст программы: приведен выше.

Пояснение к программе. Программа начинается с ввода коэффициентов уравнения a, b, c и вычисления значений $2a, u, D$. Далее проверяется условие $D \geqslant 0$ (строка 30) и при его выполнении управление передается на строку 60 – вычисление вещественных корней x_1, x_2.

При невыполнении условия (т.е. при $D < 0$) управление передается на следующую строку 35 – вычисляются комплексные корни.

Порядок счета по программе:

1) ввести программу

2) включить счет: RUN

3) ввести коэффициенты уравнения a, b, c

4) считать результаты

Тестовые примеры:

1) $x^2 + x - 6 = 0$; $a=1$, $b=1$, $c=-6$; $x_1 = 2$, $x_2 = -3$.

2) $2x^2 - 3x + 5 = 0$; $a=2$, $b=-3$, $c=5$; $u = 0.75$, $v = 1.39$.

6.4. Циклический алгоритм

Циклические алгоритмы составляются для задач, использующих однородные, повторяющиеся (циклические) вычисления: вычисления сумм, произведений, итерационные вычисления. Для организации таких вычислений в состав циклического алгоритма включают блоки, выполняющие следующие функции:

- блок рабочих команд, выполняющий вычисления;
- блок управления циклом, регулирующий число повторений блока рабочих команд.

Состав рабочих команд определяется содержанием конкретной задачи, блок управления формирует безусловные и условные переходы, а также специальные циклические операторы – циклы.

Схема циклической программы имеет вид:

Схема "Циклическая программа"

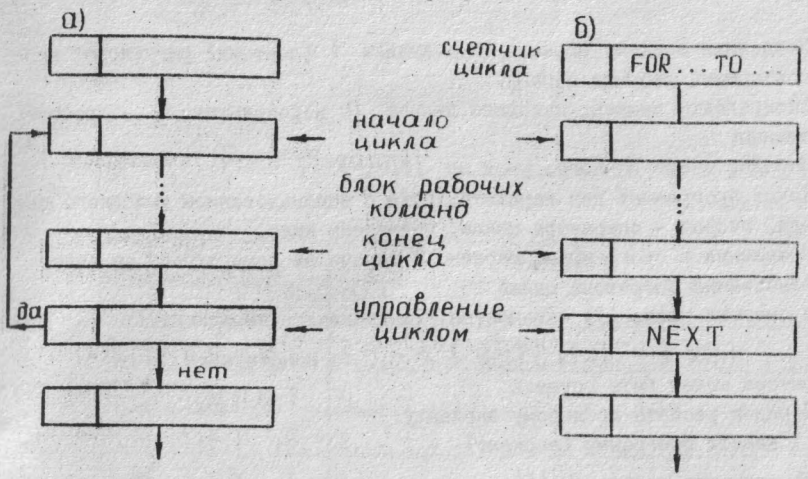


Схема а) иллюстрирует выполнение циклических вычислений с использованием условного перехода. В состав блока управления здесь включено условие, регулирующее число повторений рабочего блока (счетчик повторений) или условие, определяющее окончание вычислений.

В схеме б) счетчик повторений реализуется с помощью операторов

FOR ... TO ... STEP и NEXT .

По типу окончаний циклические алгоритмы подразделяются на алгоритмы с заданным числом повторений (конечные) и итеративные.

В конечном алгоритме число повторений рабочего блока фиксировано. Примером такого алгоритма является вычисление факториала:

$$n! = 1 \cdot 2 \cdot 3 \cdot \ldots \cdot (n-1) \cdot n = F \quad .$$

Для его вычисления необходимо перемножить n первых натуральных чисел, причем число перемножений $n-1$ является заданной величиной.

Расчетный алгоритм определяется вычислительной схемой:

1) зададим $F = 1$, $i = 1$

2) вычислим $F = F \cdot i$

3) вычислим $i = i + 1$, т.е. к значению i прибавим единицу и полученный результат припишем снова i

4) проверим условие $i \leq n$, если оно выполняется, перейдем к п.2)

5) стоп

Введенная в схему переменная величина I (счетчик) регулирует число повторений рабочих команд.

Распределим память: исходное данное N и результат F – простые переменные.

Порядок ввода данного: ввод по INPUT

Текст программы: два варианта, один с использованием условного перехода, второй – оператора цикла, приведены ниже.

Пояснения к программам. Вариант II короче на одну строку за счет использования оператора цикла

Если счетчик цикла (I) изменяется, увеличиваясь на единицу:

FOR I = TO N STEP 1 , то конструкция STEP 1 в операторе может быть опущена.

Порядок расчета по любому варианту:

I) ввести программу (вариант)

2) включить счет: RUN

3) по запросу ввести N

4) считать результат

Тестовый пример: $5! = 120$

Итеративные циклические алгоритмы включают в состав блока управления проверку условий на окончание повторяющихся вычислений. Часто таким условием является требование достижения заданной точности результата вычислений. При программной реализации таких алгоритмов используются условные переходы.

В качестве примера рассмотрим вычисление функции e^{-x}, используя разложение в ряд:

$$e^{-x} = 1 - \frac{x}{1!} + \frac{x^2}{2!} - \frac{x^3}{3!} + \dots$$

Расчетная формула представляет бесконечный сходящийся ряд, поэтому первый шаг в разработке алгоритма – заменить бесконечный ряд подходящим конечным приближением:

$$S_n = 1 - \frac{x}{1!} + \frac{x^2}{2!} + \dots + (-1)^n \frac{x^n}{n!}$$

Для заданной точности ε вычисления заканчиваются при выполнении условия:

$$|S_n - S_{n-1}| = |U_n| < \varepsilon$$

Следующий шаг – запись текущего члена ряда с использованием известных величин:

ЭЛЕКТРОНИКА МК 90

Программа : Факториал

Строка	Оператор	Выражение
		Вариант I
5	PRINT	"Введите n"
10	INPUT	N
15	LET	F=1
20	LET	I=1
25	LET	F=F*I
30	LET	I=I+1
35	IF	I<=N THEN 25
40	PRINT	"F=";F
45	END	
		Вариант II
5	PRINT	"Введите n"
10	INPUT	N
15	LET	F=1
20	FOR	I=1 TO N
25	LET	F=F*I
30	NEXT	I
35	PRINT	"F=";F
40	END	

$$u_k = (-1)^k \cdot \frac{x^k}{k!}, \quad k = 1, 2, 3, \ldots, n.$$

Заметим, что $u_k = -u_{k-1} \cdot \frac{x}{k}$, $u_0 = 1$ и эта рекуррентная формула существенно упрощает расчетный алгоритм.

Определим порядок вычислений:

1) $S = 0$, $u = 1$, $k = 0$ (начальные значения)

2) $S = S + u$

3) $k = k + 1$

4) $u = -\frac{x}{k} \cdot u$

5) если $|u| > \varepsilon$, то перейти к п.2

6) стоп

Распределим память: исходные данные - аргумент x, точность ε и результаты S, k, u - простые переменные.

Порядок ввода данных: ввод по INPUT

Текст программы приведен ниже.

Пояснения к программе. Программа начинается с ввода данных и присваивания начальных значений величинам S, u, k.

Затем вычисляется сумма S и текущий член ряда u - строки 40÷50 (блок рабочих команд цикла).

Строка 55 - управление циклом, в нем содержится условие $|u| > \varepsilon$, определяющее окончание цикла. Пока условие выполняется (заданная точность не достигнута), вычисление суммы ряда продолжается.

После окончания цикла на экран отображаются результаты:

S - приближенное значение e^{-x}

k - число членов ряда для оценки его сходимости.

Порядок расчета по программе:

1) ввести программу

2) включить счет: RUN

3) ввести исходные данные x, ε

4) считать результаты S, k

Тестовый пример: $x = 2$, $\varepsilon = 10^{-4}$

$S = 0.13538$, $k = 11$

$e^{-2} = 0.1353353$.

ЭЛЕКТРОНИКА МК 90

Программа : Вычисление e^{-x}

Лист : 1

Листов : 1

Строка	Оператор	Выражение
5	PRINT	"ВВОД X"
10	INPUT	X
15	PRINT	"ВВОД E"
20	INPUT	E
25	LET	S=0
30	LET	U=1
35	LET	K=0
40	LET	S=S+U
45	LET	K=K+1
50	LET	U=-X/K*U
55	IF	ABS(U)>E THEN 40
60	PRINT	"S=";S
65	PRINT	"K=";K
70	END	

6.5. Алгоритм подпрограммного типа

Алгоритмы подпрограммного типа - это участки вычислений, повторяющиеся в разных местах расчетного (общего) алгоритма. Повторение часто обусловлено изменением "входных" данных участка. Повторяющиеся участки вычислений оформляются в программе в виде подпрограммы.

Подпрограмма - это выделенная в отдельный блок часть программы, начинающаяся с любой команды и заканчивающаяся командой "возврат в программу" (оператор RETURN).

Приставка "под" в слове подпрограмма указывает на подчиненность ее основной программе, которую в этом случае называют головной программой.

Структурно программа состоит из двух частей:

- обращение к подпрограмме ;
- блок рабочих команд подпрограммы.

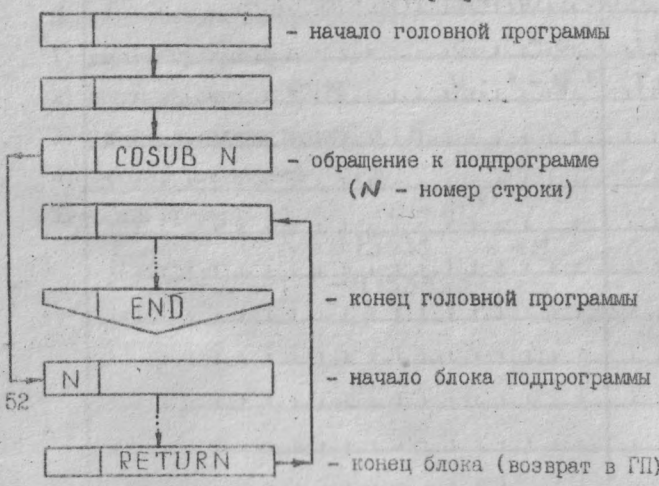
Блок рабочих команд - это программная реализация повторяющегося участка алгоритма, выделение его из состава головной программы.

Обращение к подпрограмме - это указание места в головной программе, в котором должна "сработать" данная подпрограмма.

Обращение к подпрограмме, вызывающее ее работу, составляется из команды "Подпрограмма" (оператор GOSUB) и адреса перехода.

Адрес перехода указывает номер первой строки блока рабочих команд подпрограммы.

Последний оператор блока RETURN после выполнения блока передает управление на строку головной программы, следующую за обращением к подпрограмме, как показано на схеме:



При выполнении обращения к подпрограмме:

1. Управление передается на первую строку блока подпрограммы так же, как при безусловном переходе.
2. Запоминается номер следующей за обращением строки головной программы, на которую осуществляется возврат после выполнения блока подпрограммы.

В головной программе может быть несколько подпрограмм, а в блоке рабочих команд подпрограммы может находится обращение к другой подпрограмме.

Применение подпрограммы позволяет обращаться к ней из разных мест программы, легко заменять блок рабочих команд подпрограммы.

Рассмотрим два примера составления подпрограммы.

1. Составить программу решения системы трех линейных алгебраических уравнений:

$$\begin{aligned} a_{11}x_1 + a_{12}x_2 + a_{13}x_3 &= b_1 \\ a_{21}x_1 + a_{22}x_2 + a_{23}x_3 &= b_2 \\ a_{31}x_1 + a_{32}x_2 + a_{33}x_3 &= b_3 \end{aligned}$$

по методу Крамера:

$$x_1 = D_1/D\,, \quad x_2 = D_2/D\,, \quad x_3 = D_3/D\,,$$

вычисляя последовательно четыре определителя третьего порядка:

$$D_1 = \begin{vmatrix} b_1 & a_{12} & a_{13} \\ b_2 & a_{22} & a_{23} \\ b_3 & a_{32} & a_{33} \end{vmatrix}, \qquad D_2 = \begin{vmatrix} a_{11} & b_1 & a_{13} \\ a_{21} & b_2 & a_{23} \\ a_{31} & b_3 & a_{33} \end{vmatrix}$$

$$D_3 = \begin{vmatrix} a_{11} & a_{12} & b_1 \\ a_{21} & a_{22} & b_2 \\ a_{31} & a_{32} & b_3 \end{vmatrix} \qquad D = \begin{vmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ a_{31} & a_{32} & a_{33} \end{vmatrix}$$

Используем программу "Определитель" (см. п.6.I.) для составления подпрограммы вычисления определителя.

Определим порядок вычислений:

1) D_1 2) D_2 3) D_3
4) D 5) x_1 6) x_2
7) x_3

Распределим память:

1) исходные данные представляют собой матрицу из трех строк и четырех столбцов:

$$\begin{pmatrix} a_{11} & a_{12} & a_{13} & b_1 \\ a_{21} & a_{22} & a_{23} & b_2 \\ a_{31} & a_{32} & a_{33} & b_3 \end{pmatrix}$$

Зарезервируем для нее место в памяти, определив двумерный массив A(3,3). Уточним, в массиве 4 строки (0 ÷ 3) и 4 столбца (0 ÷ 3) - объем памяти выбран с "запасом";

2) результаты вычислений - простые переменные.

Порядок ввода данных.

Исходные данные введем построчно, начиная с первой строки. Данные строки вводим по одному слева направо, например:

Строка I

a_{11}

a_{12}

a_{13}

b_1

Ввод выполняем по оператору INPUT

Текст программы приведен ниже.

Пояснение к программе. После ввода данных производится первое обращение к подпрограмме - строка 45 GOSUB 165 . При этом запоминается номер следующей строки (50) головной программы и передается управление на начало блока подпрограммы (строка 165).

Блок подпрограммы располагается за последней командой головной программы. С учетом ввода данных первое выполнение блока вызовет вычисление определителя D_1 и по команде RETURN - возврат в головную программу.

Значение первого определителя запоминается (строка 50 LETD1 = D) и производится подготовка элементов для вычисления второго определителя. Суть подготовки - перестановка столбцов в массиве A . Ее лучше уяснить, делая перестановку по программе "вручную".

Для вычисления четырех определителей выполняются три перестановки столбцов массива A и четыре обращения к подпрограмме.

Порядок расчета по программе:

1) ввести программу

2) включить счет: RUN

ЭЛЕКТРОНИКА МК 90

Программа : СЛАУ -3

Строка	Оператор	Выражение
5	DIM	A(3,3)
10	PRINT	"Введите данные"
15	FOR I	=1 TO 3
20	PRINT	"Строка", I
25	FOR J	=0 TO 3
30	INPUT	A(I,J)
35	NEXT J	
40	NEXT I	
45	GOSUB	165
50	LET D1	=D
55	FOR I	=1 TO 3
60	LET E=	A(I,0)
65	LET A(	I,0)=A(I,1)
70	LET A(	I,1)=E
75	NEXT I	
80	GOSUB	165
85	LET D2	=-D
90	FOR I	=1 TO 3
95	LET E=	A(I,0)
100	LET A(	I,0)=A(I,2)

Программа : СЛАУ-3 Лист : 2

Листов : 2

Строка	Оператор	Выражение
1.0.5	LET A(	I,,2)=E
1.1.0	NEXT I	
1.1.5	GOSUB	1.6.5
1.2.0	LET D.3	=D
1.2.5	FOR I	=1 TO 3
1.3.0	LET A(	I,,3)=A(I,,0)
1.3.5	NEXT I	
1.4.0	GOSUB	1.6.5
1.4.5	PRINT	"X1=";D1/D
1.5.0	PRINT	"X2=";D2/D
1.5.5	PRINT	"X3=";D3/D
1.6.0	END	
1.6.5	LET C=	A(3,,1)*(A(1,,2)*A(2,,3)-A(2,,2)*A(1,,3))
1.7.0	LET E=	A(3,,2)*(A(2,,1)*A(1,,3)-A(1,,1)*A(2,,3))
1.7.5	LET F=	A(3,,3)*(A(1,,1)*A(2,,2)-A(2,,1)*A(1,,2))
1.8.0	LET D=	C+E+F
1.8.5	RETURN	

3) ввести исходные данные

4) считать результаты

Тестовый пример:

$$X_1 + 2X_2 + 3X_3 = 14 \quad , \quad X_1 = 1$$
$$4X_1 + 7X_2 + X_3 = 21 \quad , \quad X_2 = 2$$
$$3X_1 + X_2 - X_3 = 2 \quad , \quad X_3 = 3$$

Последовательность вычисления примера:

- ввести элементы первой строки:
  сообщение "Строка I"
  ответ на запрос - ?
  I, 2, 3, I4 [ВК]
- ввести элементы второй строки:
  сообщение "Строка 2"
  ответ на запрос - ?
  4, 7, I, 2I [ВК]
- ввести элементы третьей строки:
  сообщение "Строка 3"
  ответ на запрос - ?
  3, I, -I, 2 [ВК]
- считать результаты:

$$X_1 = 1$$
$$X_2 = 2$$
$$X_3 = 3$$

Второй пример. Составить программу приближенного вычисления интеграла по формуле Симпсона:

$$I = \int_a^b y(x)\,dx \approx \frac{h}{3} \cdot \sum_{j=0}^{m} w_j \cdot y(x_j)$$

$y(x)$ - подынтегральная функция, заданная на отрезке $[a, b]$,

$h = (b-a)/m$ - шаг интегрирования, m - четное число узлов интегрирования,

$$x_0 = a \quad , \quad x_m = b \quad , \quad x_j = a + jh \; ;$$
$$w_0 = 1 \quad , \quad w_m = 1 \quad , \quad w_{2j-1} = 4 \quad , \quad w_{2j} = 2 \; .$$

В этом примере применение подпрограммы обусловлено необходимостью вычисления значений подынтегральной функции $y(x)$, различной в конкретных расчетах, и возможностью составления общей программы.

Определим порядок вычислений:

1) $h = (b-a)/m$

2) $x = b$, ПП: y_m, $I = y_m$

3) $x = a$, ПП: y_0, $I = I + y_0$

4) $W = 4$, $n = m-1$

5) $j = 1$

6) $x = x + h$, ПП: y

7) $I = I + W \cdot y$

8) $W = 6 - W$

9) $j = j + 1$, если $j \leq n$ то перейти к п.6)

10) $I = I \cdot h/3$, стоп

Распределим память:

а) исходные данные:

a – нижний предел интегрирования

b – верхний предел интегрирования

m – число (четное) узлов интегрирования, простые переменные

б) результаты: h, x, y, I, W, n – простые переменные.

Порядок ввода данных: исходные данные вводим по оператору INPUT в указанном порядке.

Текст программы приведен ниже.

Пояснения к программе. Для вычисления интеграла необходимо составить блок рабочих команд подпрограммы расчета значения y_j конкретной подынтегральной функции $y(x)$. Блок располагается после окончания головной программы, начиная со строки 90. При составлении блока подпрограммы нужно учитывать, что текущее значение аргумента X_j формируется в головной программе (переменная X). Вычисленное значение функции должно присваиваться переменной Y, используемой в головной программе.

ЭЛЕКТРОНИКА МК 90

Программа : Вычисление интеграла методом Симпсона.

Лист : 1

Листов : 1

Строка	Оператор	Выражение
1,5	PRINT	"BBOD A, B, M"
1,0	INPUT	A, B, M
1,5	LET	H=(B-A)/M
2,0	LET	X=B:GOSUB 9,0
2,5	LET	I=Y
3,0	LET	X=A:GOSUB 9,0
3,5	LET	I=I+Y
4,0	LET	W=4
4,5	LET	N=M-1
5,0	FOR	J=1 TO N
5,5	LET	X=X+H:GOSUB 9,0
6,0	LET	I=I+Y*W
6,5	LET	W=6-W
7,0	NEXT	J
7,5	LET	I=I*H/3
8,0	PRINT	"I=";I
8,5	END	
	Пример ПП :	$y = 1/(x^2 + x + 1)$
9,0	LET	Y=1/(X*X+X+1)
9,5	RETURN	

Порядок расчета по программе:

1) составить блок подпрограммы вычисления y_i и включить его в текст программы

2) ввести программу

3) включить счет: RUN

4) ввести исходные данные: a, b, m (три числа через запятую)

5) считать результат I

Тестовый пример: $\int_0^1 \frac{dx}{(x^2+x+1)}$, $a=0$, $b=1$.

$m=10$, $I=0.6045961$

$m=100$, $I=0.6045998$

Точное значение $I=0.60459978$

Последовательность выполнения примера:

- ввести исходные данные:
  сообщение ВВод A, B, M
  ответ на запрос ? ?
  0, I, I0 (или I00) [ВК]
- считать результат:

$I=0.6045961$, если $m=10$

или $I=0.6045998$, если $m=100$

Результаты расчетов примера подтверждают высокую точность метода Симпсона при вычислении интегралов от непрерывных функций. Значение m для таких случаев можно выбрать в пределах I0÷20.

Упражнения

I. Составить программу вычисления значения функции $y(x)$ для заданного аргумента x по формуле квадратичной интерполяции:

$$y(x)=\frac{(x-x_2)(x-x_3)}{(x_1-x_2)(x_1-x_3)}\cdot y_1+\frac{(x-x_1)(x-x_3)}{(x_2-x_1)(x_2-x_3)}\cdot y_2+\frac{(x-x_1)(x-x_2)}{(x_3-x_1)(x_3-x_2)}\cdot y_3,$$

где $x_1, x_2, x_3, y_1, y_2, y_3$ – заданные величины.

С использованием программы получить таблицу значений функции Бесселя $J_0(x)$ для $0 \leqslant x \leqslant 1$ с шагом 0.2:

$x_1=0$, $x_2=0.5$, $x_3=1$

$y_1=1$, $y_2=0.9385$, $y_3=0.7652$.

2. Составить программу вычисления числа сочетаний C_n^m из n элементов по m элементов:

$$C_n^m = \frac{n!}{m!(n-m)!}, \qquad k! = 1 \cdot 2 \cdot 3 \cdots k$$

Вычислить отношение C_{36}^5 / C_{45}^6

3. Составить программу вычисления приближенного значения корня нелинейного уравнения $f(x) = 0$, используя метод Ньютона:

$$x_{i+1} = x_i - \frac{f(x_i)}{f'(x_i)}, \quad i = 0, 1, 2, \cdots, \quad x_i, x_{i+1} \text{ – два}$$

последовательных значения корня, x_0 задано,

$$f'(x_i) = \left.\frac{df(x)}{dx}\right|_{x = x_i}$$

Точность вычисления корня определяется заданной погрешностью ε:

$$|x_{i+1} - x_i| < \varepsilon$$

Найти наибольший корень уравнения $\sin x - x \cos x = 0$.

4. Составить программу приближенного вычисления интеграла по формуле трапеций:

$$\int_a^b y(x)dx \approx h \cdot \left(\frac{y_0 + y_n}{2} + y_1 + y_2 + \cdots + y_{n-1} \right)$$

$y_0 = y(a)$, $y_n = y(b)$, $y_i = y(x_i)$ – значения подынтегральной функции, $h = (b-a)/n$, n – число узлов интегрирования.

Вычислить $\int_0^{\pi/2} \frac{dx}{(\sin x + 2\cos x)^2}$ с точностью $\varepsilon = 10^{-4}$.

7. ТЕХНОЛОГИЯ ПРОГРАММИРОВАНИЯ

Множество различных аспектов принимаются во внимание при составлении программы и сама программа – их "равнодействующая".

Основная цель создания программы – получение решения задачи с ее помощью. В этом и суть термина "безбумажное программирование".

Но было бы слишком просто полагать, что программа, вырабатывающая нужные результаты, всегда является хорошей, а программист, который изготавливает программу вовремя, – хороший программист.

Основные требования к качеству программ, следующие:

- минимальный объем привлекаемых программой средств памяти;
- минимальное время работы программы;
- минимальное время изготовления программы;
- простота эксплуатации программы и достоверность получаемых результатов.

Требования эти в известном смысле противоречивы, приоритет их выполнения зависит от конкретной задачи и возможностей ЭВМ.

Рассмотрим подробнее практическую реализацию перечисленных требований.

7.1. Оптимизация программы

Сокращение объема программы и памяти, занимаемой данными, как правило, уменьшает время ее работы, упрощает структуру программы.

Исходными данными при оптимизации являются рабочая программа и полученные с ее использованием результаты расчетов. Их анализ показывает, насколько эффективна предполагаемая оптимизация программы, намечает план ее конкретной реализации.

Существуют два подхода к оптимизации рабочих программ: чистка и перепрограммирование.

Первый подход заключается в изменении самой программы с целью ее сокращения. Для этого можно использовать любые особенности языка программирования, рабочей программы, может быть, расчетного алгоритма.

Достоинство подхода в том, что он требует мало времени. Результаты повышения эффективности программы могут быть столь же малыми.

В качестве примера рассмотрим программу решения системы линейных уравнений методом Крамера, приведенную в п.6.5.

Три перестановки столбцов массива данных, реализованных в программе как независимые ее участки (строки 55÷75, 90÷110, 125÷135), можно

Программа : СЛАУ-3. Метод Крамера.

Лист : 1

Листов : 2

Строка	Оператор	Выражение
15	DIM	A(3,3)
10	PRINT	"Ввод данных"
15	FOR I	=1 TO 3
20	PRINT	"Строка",I
25	FOR J	=0 TO 3
30	INPUT	A(I,J)
35	NEXT J	
40	NEXT I	
45	GOSUB	140
50	LET D1	=D
55	LET K=	1
60	GOSUB	115
65	LET D2	=-D
70	LET K=	2
75	GOSUB	115
80	LET D3	=D
85	LET K=	3
90	GOSUB	115
95	PRINT	"X1=";D1/D
100	PRINT	"X2=";D2/D

Программа : СЛАУ - 3. Метод Крамера.

Строка	Оператор	Выражение
1 0 5	PRINT	"X3=";D3/D
1 1 0	END	
1 1 5	FOR I=1 TO 3	
1 2 0	LET E=A(I,0)	
1 2 5	LET A(I,0)=A(I,K)	
1 3 0	LET A(I,K)=E	
1 3 5	NEXT I	
1 4 0	LET C=A(3,1)*(A(1,2)*A(2,3)-A(2,2)*A(1,3))	
1 4 5	LET E=A(3,2)*(A(2,1)*A(1,3)-A(1,1)*A(2,3))	
1 5 0	LET F=A(3,3)*(A(1,1)*A(2,2)-A(2,1)*A(1,2))	
1 5 5	LET D=C+E+F	
1 6 0	RETURN	

выполнить, включив один участок в подпрограмму и сократив остальные.

Текст программы приведен выше.

Цена оптимизации в данном случае – уменьшение программы на пять строк.

Второй подход состоит в переделке рабочей программы на основе изменения (оптимизации) алгоритма решения задачи, либо замены алгоритма на новый, более эффективный. Этот подход более трудоемкий, требует тщательного изучения расчетного алгоритма и, возможно, его пересмотра, уточнение исходной математической модели.

Конечно, тщательную проработку модели и расчетного алгоритма нужно делать сразу, приступая к решению задачи – без этого не получить и рабочей программы. Подчеркивая эту мысль, опытные программисты призывают обучающихся новичков не жалеть времени и сил на такую проработку, составить детальный и наилучший план предстоящей работы – решение задачи на ЭВМ.

Однако на практике, зачастую, самый хороший план работы можно составить после ее выполнения. Это связано прежде всего с тем, что при составлении плана теперь можно учесть практический опыт, накопившийся в процессе выполнения работы. Говоря в этом смысле: "Первый блин комом", имеют в виду именно первый блин, а не любой из последующих.

Итак, реализацию второго подхода лучше выполнять в обратной последовательности этапов решения задачи на ЭВМ: программирование, разработка расчетного алгоритма, составление математической модели.

Рассмотрим пример перепрограммирования решения системы линейных алгебраических уравнений. Предыдущее сокращение программы не затрагивало самого алгоритма.

Известно, что метод Крамера по числу выполняемых операций является наиболее трудоемким. Более эффективен алгоритм последовательного исключения неизвестных – метод Гаусса.

Для системы трех уравнений расчетный алгоритм имеет вид:

$$1)\ m = a_{21} \cdot a_{32} - a_{22} \cdot a_{31}$$

$$2)\ n = a_{12} \cdot a_{31} - a_{11} \cdot a_{32}$$

$$3)\ p = a_{11} \cdot a_{22} - a_{12} \cdot a_{21}$$

$$4)\ x_3 = \frac{b_1 \cdot m + b_2 \cdot n + b_3 \cdot p}{a_{13} \cdot m + a_{23} \cdot n + a_{33} \cdot p}$$

$$5)\ x_2 = \frac{(a_{13} \cdot a_{21} - a_{11} \cdot a_{23}) x_3 + a_{11} \cdot b_2 - a_{21} \cdot b_1}{p}$$

$$6)\ x_1 = (b_1 - a_{12} x_2 - a_{13} \cdot x_3) / a_{11}$$

Программа : СЛАУ-3. Метод Гаусса

Строка	Оператор	Выражение
5	DIM	A(3,3)
10	PRINT	"Введите коэффициенты"
15	FOR I	=1 TO 3
20	PRINT	"строка",I
25	FOR J	=0 TO 3
30	INPUT	A(I,J)
35	NEXT J	
40	NEXT I	
45	LET M	=A(2,0)*A(3,1)-A(2,1)*A(3,0)
50	LET N	=A(1,1)*A(3,0)-A(1,0)*A(3,1)
55	LET P	=A(1,0)*A(2,1)-A(1,1)*A(2,0)
60	LET X	=A(1,3)*M+A(2,3)*N+A(3,3)*P
65	LET Z	=X/(A(1,2)*M+A(2,2)*N+A(3,2)*P)
70	LET X	=(A(1,2)*A(2,0)-A(1,0)*A(2,2))*Z
75	LET Y	=(X+A(1,0)*A(2,3)-A(2,0)*A(1,3))/P

Программа : СЛАУ-3. Метод Гаусса. Лист : 2

Листов : 2

Строка	Оператор	Выражение
80	LET	X=(A(1,3)-A(1,1)*Y-A(1,2)*Z)/A(1,0)
85	PRINT	"X1=";X
90	PRINT	"X2=";Y
95	PRINT	"X3=";Z
100	END	

Распределение памяти и порядок ввода данных оставим прежними.

Текст программы приведен выше.

Результат оптимизации: программа короче предыдущей на 12 строк - показатель относительный.

Абсолютный показатель - сокращение объема ОЗУ, занимаемой программой, составляет 36 байт (~ 10%).

Более значительно (более чем в 2 раза) сокращение времени счета по программе - и это основной результат оптимизации.

7.2. Тестирование программ

Тестирование программы - это испытание на правильность ее работы, оно призвано указывать на наличие ошибок, а не на их отсутствие.

Если тестирование программы проводится интуитивно, без какого-либо плана испытаний, трудно рассчитывать на хорошие результаты. Их обеспечивает тщательный подбор данных для контрольных примеров и заблаговременный выбор элементов (участков) программы, подлежащих проверке.

Само тестирование должно выполняться последовательно и аккуратно с наибольшей полнотой.

Поспешным является вывод о правильности программы лишь на том основании, что она воспринята ЭВМ и выдает численные результаты. Все, что достигнуто в данном случае - это получение некоторой выходной информации, не обязательно правильной.

В программе все еще могут быть "счетные" ошибки, а между тем задача, которая ставится при написании программы - это не просто получение ответов, но получение правильных ответов. Поэтому обычно бывает необходима"ручная" проверка машинных результатов.

Способы такой проверки - прикидка результатов, использование, как контрольных, некоторых расчетных соотношений, выполнение программы в режиме "ручного счета".

Прикидка результатов выполняется с учетом области допустимых значений решения задачи, которая определяется в процессе построения математической модели. Анализируя область допустимых значений решения задачи, можно оценить знак решения, его величину (хотя бы порядок величины) и т.д.

Полезной является подстановка оцениваемых результатов в какое-либо из расчетных соотношений, и проверка их достоверности.

Примером служит программа решения системы трех линейных алгебраических уравнений (п.6.5). Полученное решение системы $X_1 = 1$,

$X_2 = 2$, $X_3 = 3$ можно проверить подстановкой в любое из уравнений системы, скажем в первое:

$$1 + 2 \cdot 2 + 3 \cdot 3 = 14$$

Основной метод тестирования программы – решение контрольных примеров с различными вариантами исходных данных.

Хорошо подобранные исходные данные из области своих допустимых значений обеспечивают необходимую полноту тестирования программы, устанавливают ограничения ее применения.

В качестве второго примера тестирования рассмотрим задачу нахождения корней квадратного уравнения (п.6.3). Программа решения задачи использует в качестве исходных данных коэффициенты a , b , c и в зависимости от их значений определяет действительные или комплексные корни уравнения.

Варианты выбранных исходных данных и расчетные ситуации отобразим в таблице.

Таблица тестирования квадратного уравнения

№ п/п	Коэффициенты			Корни	Примечание
	a	b	c		
I	I	I	-6	$X_1 = 2, X_2 = -3$	Хороший начальный тест
2	2	-3	5	$0.75 \pm j\,1.39$	Комплексные корни
3	I	0	0.25	$0 \pm j\,0.5$	Мнимые корни
4	2	I	0	$X_1 = 0, X_2 = -0.5$	Все нормально
5	0	2	I	нет	Должен быть один корень Почему нет корней?
6	2	0	0	$X_1 = 0, X_2 = 0$	Все нормально
7	0	2	0	нет	Должен быть один корень (что показывает дисплей?)
8	0	0	2	нет	Неправильное уравнение (что показывает дисплей?)
9	0	0	0	нет	Почему? (что показывает дисплей?)

Результаты тестирования показывают, что программа не учитывает случай $a = 0$ (тесты 5, 7), т.е. не предусматривает решение линей-

ного уравнения: $bx + c = 0$

Выявленное ограничение $a \neq 0$ необходимо указать в инструкции к программе, либо дополнить программу блоком решения линейного уравнения.

В остальных случаях программа выдает правильные результаты и это дает основание надеяться, что она будет работать надежно.

7.3. Документирование программы

Документирование программы решает следующие задачи:

1. Объяснить, для чего предназначена программа, каковы ее ограничения, какие данные являются исходными и какие результаты могут быть получены.

2. Проинформировать о порядке выполнения программы, последовательности ввода исходных данных, особенностях режима счета, хранения и выдачи результатов.

3. Облегчить тестирование и эксплуатацию программы, поиск содержащихся в ней ошибок и возможность расширения программы, если изменяются требования к ней.

Показателем квалификации программиста является степень понимания им всего процесса программирования. При этом важный и часто упускаемый из виду аспект состоит в передаче другим программистам деталей данной программы.

Документирование не должно начинаться тогда, когда разработка программы закончена. Оно должно выполняться одновременно с разработкой программы, начиная с этапа постановки задачи.

Главная цель документации состоит в том, чтобы помочь пользователю и даже самому разработчику понять программу. Хорошая документация расширяет круг пользователей программой, увеличивает срок ее эксплуатации.

Для всех программ, кроме простейших, необходима как внешняя, так и программная документация.

Внешняя документация представляет собой описание программы, не содержащееся в программных комментариях. Она оформляется в виде руководства для пользователя, либо инструкции к выполнению программы.

Независимо от того, насколько хороша программа, она будет работать только в том случае, если для ее использования имеется надлежащая инструкция.

Даже если программа предназначена для индивидуального пользования,

следует написать инструкцию, чтобы не забыть некоторых деталей, касающихся ее работы. Если же программа предназначена для непрограммистов, необходима хорошо составленная подробная инструкция.

Программная документация - это текст программы, предназначенный не только для чтения, но и для ввода программы. Текст должен быть аккуратно оформленным, наглядным, достаточно комментированным. Большое значение при оформлении имеют бланки программы, о которых говорилось в п.5.I.

Очень важны комментарии в программе. Кроме своей информативной функции, они позволяют быстро ориентироваться в программе, выделяя в ней отдельные блоки.

Разработка программы - творческая задача, предполагающая, как правило, вариантную реализацию для получения хорошего результата. При этом нередко выясняется, что первая попытка была неудачной. Повторение работы с учетом накопленного опыта помогает создать оптимальный, легко отлаживаемый вариант программы.

В качестве примера выполним документирование программы решения системы трех линейных алгебраических уравнений, приведенной в п.7.I.

Программная документация - текст оптимизированного варианта программы с комментариями приведен в п.7.I.

Внешняя документация.

I. Описание программы.

Программа выполняет решение системы трех линейных алгебраических уравнений:

$$a_{11}x_1 + a_{12}x_2 + a_{13}x_3 = b_1 \quad , \quad a_{11} \neq 0$$
$$a_{21}x_1 + a_{22}x_2 + a_{23}x_3 = b_2$$
$$a_{31}x_1 + a_{32}x_2 + a_{33}x_3 = b_3$$

методом Гаусса.

Здесь $a_{11}, \ldots, a_{33}$ - вещественные коэффициенты системы, b_1, b_2, b_3 - вещественные числа (правая часть системы), x_1, x_2, x_3 - неизвестные вещественные величины системы.

Алгоритм решения системы определен в п.7.I.

Программа обеспечивает получение единственного решения системы. Если система не имеет решения или имеет бесчисленное множество решений, то выдается сообщение об ошибке. Тип ошибки - переполнение при делении (ошибка I25).

Исходные данные a_{ij}, b_i вводятся построчно, начиная с первой

строки. Указанием ввода является сообщение "Ввод данных" на дисплее. Дополнительное сообщение "Строка I", "Строка 2", "Строка 3" регулирует порядок ввода.

Данные каждой строки вводятся по одному в следующем порядке:

a_{i_1}

a_{i_2}

a_{i_3}

b_i , где i - номер строки

Результаты вычислений отображаются на экране в виде:

X1 = A

X2 = B

X3 = C где A, B, C - конкретные числовые значения результатов.

2. Инструкция к выполнению программы:

1) ввести программу

2) включить счет, выполнив команду RUN

3) ввести исходные данные

4) считать результаты

5) для нового счета перейти к п.2

3. Тестовый пример: $X_1 + 2X_2 + 3X_3 = 14$

$4X_1 + 7X_2 + 3X_3 = 21$

$3X_1 + X_2 - X_3 = 2$

Ввод исходных данных выполняется после сообщения "Ввод данных":

- ввести элементы первой строки:
  сообщение "Строка I"
  ответ на запрос - ?
  I, 2, 3, I4 [ВК]
- ввести элементы второй строки:
  сообщение "Строка 2"
  ответ на запрос - ?
  4, 7, 3, 2I [ВК]

- ввести элементы третьей строки:
  сообщение "Строка 3"
  ответ на запрос - ?
  3, I, -I, 2 [ВК]

Результаты вычислений отображаются на дисплее в виде:

X1 = 1

X2 = 2

X3 = 3 .

7.4. Прикладные программы

Массовое использование персональных ЭВМ для решения различных по типу задач предъявляет определенные требования к разработке математического их обеспечения. Это прежде всего прикладные программы, главным требованием в создании которых является ориентация на запросы широкого круга пользователей - непрофессионалов в области вычислительной техники.

В разработке программ необходим комплексный подход, предусматривающий объединение программ в специализированные пакеты, обеспечивающие эффективное решение инженерных задач, статистическую обработку экспериментальных данных и т.д.

Публикуемые сейчас прикладные программы составляются для ПЭВМ, использующих различные версии языка Бейсик: Электроника-ДЗ-28, Электроника-С60, Искра-226, ДВК-2 и др., а также зарубежные ПЭВМ.

Различие версий Бейсика затрудняет непосредственный ввод программ в ПЭВМ Электроника МК 90, требуется их предварительный перевод.

При переводе программ необходимо:

I. Уточнить особенности версии Бейсика, на которой написана программа, сходство этой версии с версией МК 90 .

2. Если версии Бейсика достаточно близки, можно использовать построчный или пооператорный перевод, учитывая синтаксические различия.

3. Отсутствующие в версии МК 90 операторы или математические функции программы-оригинала по возможности заменить схожими по смыслу последовательностями операторов, функций версии МК 90 .

4. Если версии Бейсика сильно различаются, целесообразно подготовить детальный алгоритм решения задачи и затем заново составить про-

грамму для МК 90.

При работе с прикладными программами следует обращать внимание на содержание программы и ее оформление. Некритический подход к чужим программам может приводить к неверным результатам, большим затратам времени и сил на их исправление.

В то же время оформление программы, ее документирование характеризуют уровень программиста-разработчика и степень доверия к программе.

В качестве примера рассмотрим программу, выполняющую арифметические операции с комплексными числами, приведенную в справочнике [3].

Описание программы:

Комплексные числа записываются в алгебраической форме

$$Z = A + jB = Re\ Z + j\ Im\ Z.$$

Сложение и вычитание комплексных чисел

$Z_1 = A_1 + jB_1$ и $Z_2 = A_2 + jB_2$ выполняется по формулам

$$Z_1 + Z_2 = (A_1 + A_2) + j(B_1 + B_2)$$

$$Z_1 - Z_2 = (A_1 - A_2) + j(B_1 - B_2)$$

Умножение и деление комплексных чисел

$Z_1 = A_1 + jB_1$ и $Z_2 = A_2 + jB_2$ выполняется по формулам

$$Z_1 \cdot Z_2 = (A_1A_2 - B_1B_2) + j(A_1B_2 + B_1A_2)$$

$$\frac{Z_1}{Z_2} = \frac{A_1A_2 + B_1B_2}{A_2^2 + B_2^2} + j\frac{B_1A_2 - A_1B_2}{A_2^2 + B_2^2}$$

Текст программы:

```
10 PRINT ' АРИФМЕТИЧЕСКИЕ ОПЕРАЦИИ С КОМПЛЕКСНЫМИ'
20 PRINT ' ЧИСЛАМИ В АЛГЕБРАИЧЕСКОЙ ФОРМЕ'
30 INPUT ' ВВЕДИТЕ RE Z1, IM Z1' A, B
40 INPUT ' ВВЕДИТЕ RE Z2, IM Z2' C, D
50 INPUT ' ВВЕДИТЕ КОД ОПЕРАЦИИ +1, -2, *3, /4' K
60 IF K=1 THEN 110
70 IF K=2 THEN 130
80 IF K=3 THEN 140
90 IF K=4 THEN 160
100 PRINT ' КОД УКАЗАН НЕВЕРНО ': GOTO 50
110 LET A=A+C : LET B=B+D
120 PRINT ' Z0 = 'A' + J*('B')': GOTO 40
130 LET A=A-C : LET B=B-D : GOTO 120
140 LET E=A*C - B*D : LET F=A*D + C*B
150 LET A=E : LET B=F : GOTO 120
160 LET L=C*C + D*D : LET E=A*C + B*D : LET F=B*C-A*D
170 LET A=E/L : LET B=F/L : GOTO 120 : END
```

Инструкция к выполнению программы отсутствует.

Контрольный пример:

$$\frac{(5-j3)(3+j2)}{(5+j3)(2-j4)} + (0.5 - j\cdot 1)$$

Результат $Re\, Z_0 = 1{,}158823529$ и $Im\, Z_0 = 1{,}464705882$,

т.е. $Z_0 = 1{,}158823529 + j\cdot 1{,}464705882$

Уточним характер доработок программы:

I. В описании программы не пояснены и не используются в расчетном алгоритме величины $Re\, Z$, $Im\, Z$. Дважды определен вид комплексных чисел Z_1 и Z_2. Не указано, что при делении Z_1 / Z_2 и $Z_2 = 0$ решения нет, а сообщение об ошибке не комментируется.

Нет описания порядка ввода данных, при выполнении "цепочных" расчетов предусмотрен ввод только второго числа (Z_2).

2. Необходима инструкция к выполнению программы, поясняющая особенности ее работы.

3. Текст программы составлен на версии Бейсика, применяемой в массовых системах подготовки программ на базе микро-ЭВМ "Электроника - ДЗ - 28".

При переводе программы на версию Бейсика МК 90 необходимо заменить комментированные операторы INPUT (строки 30, 40, 50), что удлиняет программу.

Например, строка 30 переводится так:

```
30 PRINT " ВВЕДИТЕ  RE Z1, IM Z1" : INPUT A, B
```

С другой стороны, функция INC версии МК 90 позволяет отказаться от цифрового кодирования арифметических операций (строки 50÷90), заменив его привычным символьным кодированием:

\+ - сложение
\- - вычитание
* - умножение
/ - деление

Подробнее об этом сказано ниже.

4. Контрольный пример "закрепляет" недостаток программы - цепочные расчеты предполагают для выполнения следующей (за текущей) операции ввод только второго числа.

Небольшое изменение примера:

$$(0{,}5 + j1) - \frac{(5-j3)(3+j2)}{(5+j3)(2-j4)}$$

может при его выполнении поставить пользователя в тупик.

Результаты контрольного примера не согласованы по точности с исходными данными. Они служат хорошей иллюстрацией к высказыванию К.Ф.Гаусса:"Недостатки математического образования с наибольшей отчетливостью проявляются в чрезмерной точности численных результатов".

Выполним документирование и перевод данной программы с учетом программы "комплексная арифметика" (п.6.2).

Описание программы.

Программа "Комплексная арифметика" выполняет арифметические операции с комплексными числами в алгебраической форме $Z = A + jB$, где

A - действительная часть комплексного числа;

jB - мнимая часть комплексного числа;

B - коэффициент при мнимой части.

Пусть даны два комплексных числа:

$$Z_1 = A_1 + jB_1 \quad , \quad Z_2 = A_2 + jB_2$$

Тогда результаты операций $Z_3 = A_3 + jB_3$ вычисляются по формулам:

сложение: $Z_3 = Z_1 + Z_2$. $A_3 = A_1 + A_2$, $B_3 = B_1 + B_2$

вычитание: $Z_3 = Z_1 - Z_2$. $A_3 = A_1 - A_2$, $B_3 = B_1 - B_2$

умножение: $Z_3 = Z_1 \cdot Z_2$. $A_3 = A_1 \cdot A_2 - B_1 B_2$, $B_3 = A_2 \cdot B_1 + A_1 \cdot B_2$

деление: $Z_3 = Z_1 / Z_2$. $R = A_2^2 + B_2^2$

$$A_3 = (A_1 \cdot A_2 + B_1 \cdot B_2) / R$$

$$B_3 = (A_2 \cdot B_1 - A_1 \cdot B_2) / R$$

При выполнении цепочных вычислений в случае $Z_2 = 0$ выдается сообщение "Ошибка I25" (переполнение при делении).

Исходные данные A_1, B_1 и A_2, B_2 вводятся последовательно при выполнении программы.

Указанием ввода является сообщение на экране "Ввод данных". Дополнительные сообщения "Введите A_1, B_1", "Введите A_2, B_2" уточняют порядок ввода.

Результаты вычислений отображаются на экране в виде:

<Операция>

A3 = < M >

B3 = < M >

где M, N - конкретные числовые значения результатов.

< Операция > - сообщение о типе выполненной операции.

При выполнении цепочных вычислений результаты предыдущей опера-

ции (A_3, B_3) , являясь "первым" числом (A_1, B_1) для следующей операции, обуславливают ввод только второго (A_2, B_2) числа.

Инструкция к выполнению программы:

1. Ввести программу
2. Включить счет: RUN
3. Ввести исходные данные: а) A_1, B_1 б) A_2, B_2
4. Для выполнения операции нажать клавиши:
   - сложение [+]
   - вычитание [−]
   - умножение [*]
   - деление [/]

   Клавиши нажимаются в положении переключателя В/Н - ВР
5. Считать результаты: A_3 , B_3
6. При продолжении расчета:

а) цепочные вычисления, ввод второго числа - перейти к п.3б , нажав клавиши:

[F] [G] [3] [0] [ВК]

б) операция с двумя числами -перейти к п.2.

Тестовый пример:

$$\frac{(3+j4)}{(7+j2)}+(7.4-j\cdot5.6)\cdot(6.4+j8)-(2.6-j1.9)=89.8+j\cdot25.9.$$

Порядок действий после ввода программы:

1. Включить счет, нажав клавиши [F] [R] [ВК]
2. Выполнить деление:
   - ввести данные: A1=3 , B1=4
     A2=7 , B2=-2
   - указать операцию, нажав клавишу [/]
   - записать результаты A3=0.245 , B3=0.641
3. Выполнить умножение:
   -включить счет, нажав клавиши [F] [R] [ВК]

- ввести данные: A1 = 7.4 , B1 = -5.6

A2 = 6.4 , B2 = 8

- указать операцию, нажав клавишу [*]

4. Выполнить сложение, используя предыдущий результат:
   - включить счет, нажав клавиши [F] [G] [3] [0] [ВК]
   - ввести второе данное: A2 = 0.245 , B2 = 0.641
   - указать операцию, нажав клавишу [+]
5. Выполнить вычитание, используя предыдущий результат:
   - включить счет, нажав клавиши [F] [G] [3] [0] [ВК]
   - ввести второе данное: A2 = 2.6 , B2 = -1.9
   - указать операцию, нажав клавишу [-]
6. Считать результат примера:

   A3 = 89.8

   B3 = 25.9 , с округлением.

Текст программы приведен ниже.

Рассмотрим подробнее фрагмент программы с 55 по 80 строки.

Оператор LET K = INC (строка 55) присваивает переменной K значение функции INC - числовой ход нажатой клавиши.

Если в момент выполнения этого оператора клавиша не нажата, значение функции равно 0. Условие K = 0 (строка 60) выполняется и происходит возврат на строку 55, т.е. программа "ждет" нажатия клавиши арифметической операции. При нажатии клавиши операции функция INC выдает ее числовой код:

Нажата клавиша (В/Н - ВР)	Значение INC
[+] - сложение	59
[-] - вычитание	45
[*] - умножение	58
[/] - деление	47

Соответственно переменная K принимает одно из указанных значений INC .

Далее в программе (строки 65÷80) проверяется значение K и осу-

Программа : Комплексная арифметика Лист : 1

Листов : 2

Строка	Оператор	Выражение
1,5	CLS:PR	INT
1,0	PRINT"	Вводи данных"
1,5	PRINT	
2,0	PRINT	Введите A1, B1"
2,5	INPUT	A, B
3,0	CLS:PR	INT
3,5	PRINT"	Введите A2, B2"
4,0	INPUT	C, D
4,5	CLS:PR	INT
5,0	PRINT"	Укажите операцию"
5,5	LET K=	INC
6,0	IF K=0	THEN 5,5
6,5	IF K=5	G THEN 100
7,0	IF K=4	5 THEN 130
7,5	IF K=5	8 THEN 145
8,0	IF K=4	7 THEN 160
8,5	PRINT:	PRINT" Операция"
9,0	PRINT"	чказана неверно"
9,3	FOR I=	1 TO 8:NEXT I
9,5	GOTO 4	5
10,0	CLS:PR	INT: PRINT" Сложение"
10,5	LET E=	A+C: LET F=B+D

Программа : Комплексная арифметика

Лист : 2

Листов : 2

Строка	Оператор	Выражение
1,1,0	LET A=	E:LET B=F
1,1,5	PRINT"	A3=";E
1,2,0	PRINT"	B3=";F
1,2,5	DIS:END	
1,3,0	CLS:PRI	NT:PRINT" Вычитание"
1,3,5	LET E=	A-C:LET F=B-D
1,4,0	GOTO 1	1,0
1,4,5	CLS:PRI	NT:PRINT" Умножение"
1,5,0	LET E=	A*C-B*D:LET F=C*B+A*D
1,5,5	GOTO 1	1,0
1,6,0	CLS:PR	INT:PRINT" Деление"
1,6,5	LET H=	C*C+D*D
1,7,0	LET F=	(C*B-A*D)/H
1,7,5	LET E=	(A*C+B*D)/H
1,8,0	GOTO 1	1,0

ществляется переход к блоку выполнения арифметической операции.

Ошибка при нажатии клавиши операции исправляется (строки 85÷95) повторным запросом "Укажите операцию".

О других операторах программы.

Оператор CLS очищает экран и устанавливает курсор в левом верхнем углу экрана. С экрана убирается вся информация, в том числе и служебная строка.

Оператор PRINT (строка I5) без указания выражения ("пустой" PRINT) осуществляет пропуск строки, улучшая наглядность отображаемой информации.

Строка 93 FORI=1 TO 80 : NEXTI - "пустой" цикл, обеспечивающий временную задержку при отображении сообщения:

Операция
указана неверно,

чтобы пользователь мог его прочитать.

Оператор DIS (строка I25) восстанавливает служебную строку на экране, "стертую" оператором CLS . Пользователь после выполнения программы может вновь контролировать положение переключателей символов.

ПРОГРАММА "КОМПЬЮТЕР МК 90"

"Компьютер МК 90" - информационная программа, демонстрирующая возможности портативной микро-ЭВМ "Электроника МК 90".

Информация в процессе выполнения программы отображается на экране дисплея в виде компактных текстовых блоков и графических изображений.

В начальной части программы поясняются режимы отображения текстовой информации с использованием русского и латинского алфавитов. Затем перечисляются вычислительные операции ПЭВМ, виды решаемых на ней задач.

Четыре графических изображения (рисунка) иллюстрируют графические возможности ПЭВМ.

В конце программы ПЭВМ "исполняет" музыкальный фрагмент И.С.Баха.

В тексте программы, приведенном ниже, использованы практически все операторы Бейсика, реализованные в версии языка ПЭВМ.

Программа занимает 4639 байт памяти ОЗУ или 6 Кбайт памяти сменного модуля ОЗУ.

Время выполнения программы составляет ~ 4 минуты.

Инструкция к программе:

1. Имя программы " РК ".
2. Ввод программы: набрать на клавиатуре текст программы, размещая его в ОЗУ.

Записать программу в сменный модуль СМП под указанным именем.

3. Выполнение программы:
- если программа находится в ОЗУ, выполнить команду RUN
- если программа находится в рабочем модуле СМП, выполнить команду вызова ее в ОЗУ:

```
LOAD " РК", R
```

Время вызова ~ 1,5 минуты.

Программа : Компьютер МК 90

Лист : 1
Листов : 10

Строка	Оператор	Выражение
2	CLS:DRAW	020,20:PRINT<S2,2>"
3	4,PABCT	":DRAW020,41
4	PRINT<S	2,2>"ВЫЙТЕ !":PLAY19
,4	0:PLA	Y24,48:PLAY28,54
6	PLAY31,	260:CLS:DRAW021,21:
P	RINT"Я	-персональный"
8	DRAW021	,35:PRINT"компьютер"
:D	RAW07	,49
10	PRINT"Э	лектроника МК90":
F	OR I=1T	03:PLAY28,48
12	FORJ=1T	05:NEXTJ:NEXTI:PLAY
2	8,54:F	ORI=1T013:NEXTI
14	PLAY26,	48:PLAY29,130:FORI=1
T	03:CL3	:DRAW020,32
16	PRINT<S	1,1>"МК 90":FORJ=1T0
9	9:NEXT	J:NEXTI:CLS
18	DRAW07,	35:PRINT"или короче-
П	еко":F	ORI=1T02000:NEXTI:CLS
20	PRINT:P	RINT" я создан "::
P	RINT"	обзвединенииу"
22	PRINT"	Интеграл,г Минск":
F	ORI=1T	03200:NEXTI:CLS:PRINT

Программа : Компьютер МК 90

Строка	Оператор	Выражение
24	PRINT"	Сделаю, как вы видите":
	PRINT"	я хорошо:"
26	PRINT"	-компактный": PRINT"
	-красивый"	
28	PRINT"	-и, заметьте, умный":
	FOR I=1 T	0500: NEXT I: CLS
30	DRAW01,0	,32: PRINT" НРАВЛЮСЬ Я
	ВАМ?":	FOR I=0 TO 6: PLAY I, 20
32	NEXT I: C	LS: DRAW01,0,32: PRINT
	"ПОНЯЛ,	ПРОДОЛЖАЮ"
34	FOR I=6 T	00 STEP -1: PLAY I, 10:
	NEXT I: C	LS: PRINT
36	PRINT"	Относительно умна"
	PRINT"	добавлю: чем я его
38	PRINT"	целых 48 БИС": PRINT:
	PRINT"	БИС - большая"
40	PRINT"	интегральная схема":
	FOR I=1 T	0500: NEXT I: CLS
42	PRINT: PR	INT" Сама БИС малень
	кая": PR	INT" - вот такая:"
46	DRAW A5,0	,50,,70,,34: FOR I=0 TO 6:
	PLAY I,20	: NEXT I: CLS: PRINT

Программа : Компьютер МК 90 Лист : 3

Листов : 10

Строка	Оператор	Выражение
4,8	P,R,I,N,T,"	Н,о, ,ч,т,о, ,б,ы, ,м,ы, ,(,л,ю,д,и,"
	P,R,I,N,T,"	и, ,к,о,м,п,ь,ю,т,е,р,ы,), ,б,е,з,"
5,0	P,R,I,N,T,"	н,е,е, ,д,е,л,а,л,и,?," :,P,R,I,N,T,
	P,R,I,N,T,"	Я, ,е,й, ,д,а,ж,е, ,с,о,н,е,т,"
5,2	P,R,I,N,T,"	п,о,с,б,я,т,и,л,:," :,F,O,R,I,=,1,T,O,
	5,0,0, :,N,E,X	T,I, :,C,L,S, :,P,R,I,N,T, :,P,R,I,N,T
5,4	P,R,I,N,T,"	З,н,а,м,е,н,ь,е, ,в,е,к,а," :,P,R,I,N,T
	", ,м,а,л,ю,т	к,а, ,Б,И,С,"
5,6	P,R,I,N,T,"	к,о,м,п,ь,ю,т,е,р, ,П,е,к,о," :,
	P,R,I,N,T,"	ч,с,ч,л,и,т, ,м,ы,с,л,ь,"
5,8	F,O,R,I,=,1,T	O,4,5,0, :,N,E,X,T,I, :,C,L,S, :,P,R,I,N,T
	:,P,R,I,N,T,"	З,н,а,к,о,м, ,в,а,м, ,Б,е,й,с,и,к,?,"
6,0	P,R,I,N,T,"	о,д,н,о, ,и,з, ,д,в,у,х, :,:,:," :,
	P,R,I,N,T,"	к,о,м,п,ь,ю,т,е,р, ,П,е,к,о,"
6,2	P,R,I,N,T,"	в,а,ш, ,л,у,ч,ш,и,й, ,д,р,у,г,!," :,
	F,O,R,I,=,0,T	O,4,0, :,P,L,A,Y,I, ,1,0, :,N,E,X,T,I,
6,4	C,L,S, :,P,R,I	N,T, :,P,R,I,N,T," ,К,с,т,а,т,и, ,о,
	Б,е,й,с,и,к,е	", :,P,R,I,N,T," ,Э,т,о, ,м,о,й,
6,6	P,R,I,N,T,"	р,о,д,н,о,й, ,я,з,ы,к," :,F,O,R,I,=,1,
	T,O,3,0,0, :,N	E,X,T,I, :,C,L,S, :,P,R,I,N,T,
6,8	P,R,I,N,T,"	В,ы, ,н,а,п,р,и,м,е,р," :,P,R,I,N,T,"
	п,и,ш,е,т,е,	т,а,к, :"

Программа : Компьютер МК 90

Лист : 4
Листов : 10

Строка	Оператор	Выражение
70	PRINT"	2 x 2 = 4" : PRINT :
	PRINT"	Ая мак :"
72	PRINT"	PRINT 2 * 2" : FOR I = 1 TO
	500 : NEX	T I
74	CLS : PRI	NT : PRINT" Или мак :" :
	PRINT : P	RINT" 10 LET A = 2 * 2"
76	PRINT"	20 PRINT A" : PRINT" 3
	0 END" :	PRINT" RUN"
78	FOR I = 1 T	O 500 : NEXT I : CLS : PRINT
	: PRINT"	Знаю цифры :"
80	PRINT"	-русский" : PRINT" -бе
	лорусский	" : PRINT" -english"
82	PRINT"	-deutsch, ua." :
	FOR I = 1 T	O 500 : NEXT I
84	PRINT"	японского не знаю" :
	FOR I = 1 T	O 2300 : NEXT I : CLS : PRINT
86	PRINT"	Теперь о том, что" :
	PRINT"	я умею :" : LET A = 1
88	FOR I = 1 T	O 3000 : NEXT I : CLS : PRINT
	" Выполня	ю операции :"
90	PRINT"	-сложение" : PRINT" -в
	ычитание"	

Программа : Компьютер МК 90 Лист : 5

Листов : 10

Строка	Оператор	Выражение
90 2	PRINT"	-умножение":PRINT"
	деление"	
90 4	PRINT"	-возв.в степень"::
	PRINT	-извлечение корня"
90 6	FOR I=1T	OAX:NEXT I:CLS:PRINT:
	PRINT"	Вычисляю функции:"
90 8	PRINT"	-сини":PRINT" -коси
	нус":PR	INT" -арктангенс"
100	PRINT"	-экспонента":FOR I=1
	TOAX:NEX	T I:CLS:PRINT
102	PRINT"	Вычисляю функции:"::
	PRINT"	-логарифм"
104	PRINT"	-модуль числа":PRINT
	" -знак	числа"
106	PRINT"	-целая часть числа":
	FOR I=1T	OA:NEXT I:CLS
108	DRAW10,10	,22:1FA<>1THEN24
110	PRINT" СПЕВАЕТ ЗА МНОЙ?"::	
	FOR I=0T	O7::PLAY I,20:NEXT I
112	DRAW05,0	,32:PRINT"ШУТКА!"::
	FOR I=7T	O0STEP-1::PLAY I,10
114	NEXT I:C	LS:PRINT:LET A=380

Программа : Компьютер МК 90

Строка	Оператор	Выражение
1 1 6	P R I N T "	У ч и т и т е , я ш у с т р ы й ! " :
	P R I N T : F	O R I = 1 T O 1 6 0 : N E X T I
1 1 8	P R I N T "	5 0 т ы с я ч " : P R I N T " о п е
	р а ц и й " :	P R I N T " в с е к у н д у "
1 2 0	P R I N T "	и э т о - Н Е Ш У Т К А ! " :
	F O R I = 1 T	O A : N E X T I : C L S : P R I N T
1 2 2	P R I N T "	И т а к , ч т о ж е я " : P R I N T
	" У М Е Ю :	" : G O T O 8 8
1 2 4	C L S : P R I	N T : P R I N T " Г е н е р и р у ю "
	: P R I N T "	с л у ч а й н ы е ч и с л а "
1 2 6	P R I N T "	В ы п о л н я ю " : P R I N T " с т а
	т и с т и ч е	с к у ю "
1 2 8	P R I N T "	о б р а б о т к у д а н н ы х " :
	F O R I = 1 T	O A : N E X T I : C L S : P R I N T
1 3 0	P R I N T "	Р е ш а ю у р а в н е н и я : " :
	P R I N T "	ч е с и с т е м ы : "
1 3 2	P R I N T "	- л и н е й н ы е а л г е б р . " :
	P R I N T "	- н е л и н е й н ы е "
1 3 4	P R I N T "	- д и ф ф е р е н ц и а л ь н ы е " :
	P R I N T "	- и н т е г р а л ь н ы е "
1 3 6	F O R I = 1 T	O A : N E X T I : C L S : P R I N T :
	P R I N T "	В ы ч и с л я ю и н т е г р а л ы "

Программа : Компьютер МК 90

Лист : 7

Листов : 10

Строка	Оператор	Выражение
138	PRINT"	спец. функции":PRINT"
	числа	с заданными"
140	PRINT"	свойствами и их":
	PRINT"	последовательности"
142	FORI=1T	OA:NEXTI:CLS:PRINT"
	Рисую:"	:PRINT" -фигуры"
144	DRAWD20	,35,35,55,5,55,20,35
	:DRAWC6	0,45,10
146	DRAWA85	,55,105,35:FORI=1TO
	300:NEX	TI:CLS
148	PRINT"	-графики":FORI=0TO2
	STEP2:D	RAWO20,32
150	DRAWXI,	5,4:NEXTI:DRAWO20,32
	:DRAWX1	,10,8:LETB=2/9
152	LETC=PI	/180:FORI=0TO360STEP
	4.5:LET	X=20+I*B
154	LETY=32	-20*SIN(C*I):DRAWHX,
	Y:NEXTI	:FORK=1TO99:NEXTK
156	CLS:PRI	NT" -гистограммы":
	FORI=0T	O2:LOCATE12,10+I*16
158	PRINT86	+I:FORJ=1TO2:READB:
	LETY=I*	16+(J-1)*8+10

Программа : Компьютер МК 90

Строка	Оператор	Выражение
160	DRAW A,30	,Y+4,,30+B,Y:
	IF J=1 THE	N 164
162	DRAW 0,30	,Y:DRAW G1,,B,4
164	NEXT J:N	EXT I:FOR K=1 TO 2600:
	NEXT K:C	LS:LET B=100:LET C=55
166	PRINT"	пикомы":DRAW D25,15,
	21,,20,,2	1,,48,,19,,50,,19,,C,,23,,C
168	DRAW D23	,C,,25,,53,,27,,C,,31,,C,,3
	1,,50,,29	,48,,29,,20,,25,,15
170	DRAW D90	,,5,,86,,10,,86,,25,,84,,27
	,,84,,47,	80,,50,,80,,C,,85,,C
172	DRAW D85	,C,,90,,52,,95,,C,,B,,C,,B,
	50,,96,,4	7,,96,,27,,94,,25
174	DRAW D94	,,25,,94,,10,,90,,5:DRAW D
	23,,25:P	RINT"р":DRAW D23,,32
176	PRINT"м	":DRAW D23,,39:PRINT"о
	":FOR I=	1 TO 8:READ B,C,D,E
178	DRAW DB,	C,D,E:NEXT I:DRAW D88,
	25:PRINT	"р":DRAW D88,,32
180	PRINT"с	":DRAW D88,,39:PRINT"о
	":DRAW D	43,,25:PRINT"МИРУ"
182	DRAW D43	,37:PRINT"МИР!"

Программа : Компьютер МК 90 Лист : 9

Листов : 10

Строка	Оператор	Выражение
180	FOR K=1 T	0:A:: NEXT K:: CLS:: PRINT::
	PRINT"	И,,н,о,к,о,н,е,ц,,о,х,о,т,н,о,"
186	PRINT"	м,у,з,ы,к,а,л,ь,н,ы,е," :: FOR I=1 T 0
	260: NEX	T I:: CLS:: DRAW 05,0,,22
188	PRINT" И	.,С,.,Б,а,х," :: DRAW 03,0,,32::
	PRINT" И	н,в,е,н,ц,и,я," :: DRAW 05,,42
190	PRINT" M	o,d,e,r,a,t,o" :: FOR I=1 T 0 22::
	READ N,L	:PLAY N,,L*40:: NEXT I
192	FOR K=1 T	0 13:: NEXT K:: FOR I=1 T 0 11
	;READ N:	PLAY N,,80:: NEXT I
194	FOR K=1 T	0 13:: NEXT K:: FOR I=1 T 0 7::
	READ N:: P	LAY N,,40:: NEXT I
196	FOR I=1 T	0 4:: READ N:: PLAY N,,80::
	NEXT I:: F	OR J=1 T 0 3:: FOR K=1 T 0 13
198	NEXT K:: F	OR I=1 T 0 5:: READ N,,L
200	PLAY N,,L	*40:: NEXT I:: NEXT J:: PLAY
	31,,80:: F	OR I=1 T 0 51:: READ N
202	PLAY N,,40	:: NEXT I:: PLAY 16,,160
204	DATA 48,	39,,57,,31,,67,,27,,10,,20
	,21,,31,	29,,39,,40,,50,,10,,50,,21
206	DATA 39,	29,,31,,40,,20,,70,,20,,83
	,30,,96,	40,,11,,50,,70,,50,,83

Программа : Компьютер МК 90 Лист : 10

Листов : 10

Строка	Оператор	Выражение
210	DATA 46,	96,30,11,0,20,4,1,23,
	1,28,1,	31,1,30,1,23,1,30,1
210	DATA 33,	1,31,2,35,2,27,2,35,
	2,28,1,	23,1,28,1,31,1,30,1
212	DATA 23,	1,30,1,33,1,31,2,28,
	2,35,31	,35,28,31,23,26,24,
214	DATA 28,	33,36,33,30,33,26,30
	,21,24,	23,26,31,35,31,1,28
216	DATA 1,36	1,1,24,2,33,2,30,1,2
	,1,30,	1,23,2,31,2,28,1,24
218	DATA 1,28	8,1,21,2,30,2,23,28,
	31,30,2	3,30,33,31,28,31,35
220	DATA 33,	30,33,36,35,31,35,38
	,36,35,	33,31,30,31,33,35,36
222	DATA 33,	39,33,30,33,31,40,36
	,33,30,	33,27,30,31,28,23,28
224	DATA 30,	27,28,23,19,23
226	FOR I=1T	08:NEXT I:CLS:DRAW0
	20,20	
228	PRINT<S	2,2>"ДОСВИДАНИЯ!":DRAW0,20
	,41:PRI	NT<S2,2>"ДАНИЯ!"
230	FOR I=1T	0A:NEXT I:CLS:DIS:END

Содержание