

ГОСУДАРСТВЕННЫЙ  
КОМИТЕТ СССР  
ПО ВЫЧИСЛИТЕЛЬНОЙ  
ТЕХНИКЕ И  
ИНФОРМАТИКЕ

ВСЕСОЮЗНЫЙ МЕЖОТРАСЛЕВОЙ  
НАУЧНО-УЧЕБНЫЙ ЦЕНТР  
ПО ВЫЧИСЛИТЕЛЬНОЙ  
ТЕХНИКЕ И  
ИНФОРМАТИКЕ

**АЛГОРИТМИЧЕСКИЙ  
ЯЗЫК БЕЙСИК  
ДЛЯ ПРОГРАМ-  
МИРОВАНИЯ  
НА ПЭВМ «АГАТ»**

**(ДЛЯ УЧАЩИХСЯ 9-10 КЛАССОВ)**

**1989**

**СПРАВОЧНИК**



ГОСУДАРСТВЕННЫЙ КОМИТЕТ СССР  
ПО ВЫЧИСЛИТЕЛЬНОЙ ТЕХНИКЕ  
И ИНФОРМАТИКЕ

389-5  

---

1281-7

ВСЕСОЮЗНЫЙ МЕЖОТРАСЛЕВОЙ  
НАУЧНО-УЧЕБНЫЙ ЦЕНТР  
ПО ВЫЧИСЛИТЕЛЬНОЙ ТЕХНИКЕ  
И ИНФОРМАТИКЕ

Э.И.Бокина, Н.М.Тарасова

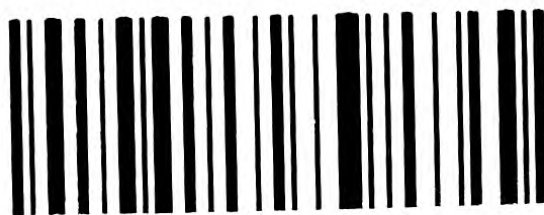
АЛГОРИТМИЧЕСКИЙ ЯЗЫК БЕЙСИК  
ДЛЯ ПРОГРАММИРОВАНИЯ  
НА ПЭВМ "АГАТ"

(для учащихся 9-10 классов)

ГОСУДАРСТВЕННАЯ  
БИБЛИОТЕКА  
СССР  
ИМ. В. И. ЛЕНИНА  
1989 г.

©

ВМНУЦ ВТИ, 1989



2015073986

## ВВЕДЕНИЕ

Язык программирования БЕЙСИК создан в 1965 г. в Англии и в настоящее время значительное число программ для микроЭВМ написано на нем. Язык БЕЙСИК — один из наиболее простых языков программирования. Правда, некоторые затруднения может вызвать то, что слова в нем английские. Впрочем, английский язык положен разработчиками в основу большинства алгоритмических языков. Если же Вы не изучали английский язык, то не огорчайтесь — чтобы объясниться на БЕЙСИКе Вам придется запомнить не более двух десятков английских слов. Изучив алгоритмический язык, Вы узнаете, как в нем называются те или иные действия и как оформляются алгоритмические конструкции. Алгоритм, записанный на языке программирования, называется программой. Язык БЕЙСИК сочетает в себе простоту и легкость для изучения и понимания. Может быть, этот справочник окажется полезным при изучении основ программирования на языке БЕЙСИК персональных ЭВМ (ПЭВМ) "Агат".

Язык БЕЙСИК предназначен для использования профессиональными программистами при разработке широкого класса программ на ПЭВМ "Агат" и неподготовленными пользователями при решении простых вычислительных задач на уровне калькулятора, а также при эксплуатации ими готовых программных изделий.

### 1. ПРОГРАММНОЕ ОБЕСПЕЧЕНИЕ ПЭВМ "АГАТ"

#### 1.1. Программа "Системный монитор"

Программа "Системный монитор" представляет собой минимум программного обеспечения, который необходим для работы пользователя с ПЭВМ "Агат", объем требуемой оперативной памяти (ОП) — 4 Кбайт. Программа выполняет три основные функции:















1. При включении ПЭВМ "Агат" ("холодный" старт) осуществляет осмотр конфигурации и запускает выполнение элементов программного обеспечения.

2. Обеспечивает обмен данными с базовыми устройствами ввода-вывода (видеоконтрольным устройством (ВКУ), клавиатурой, магнитофоном).

3. Представляет пользователю диалоговые возможности для осмотра, сравнения, изменения содержимого ОП и регистров.

При включении питания ПЭВМ "Агат" запуск программы осуществляется автоматически.

Программа "Системный монитор" воспринимает информацию трех типов: адреса, данные, команды. В диалоге этой программы можно использовать две одновременно нажимаемые клавиши:

	и		- очистка всего экрана;
	и		- очистка до конца текущей строки;
	и		- очистка до конца экрана;
	и		- звуковой сигнал;
	и		- отмена набранной строки;
			- чтение указанного курсором символа в буфер ввода;
			- удаление последнего введенного символа из буфера;
			- перевод курсора экрана ВКУ на одну строку вверх;
			- перевод курсора экрана ВКУ на одну строку вниз.

#### Функциональные клавиши ПЭВМ "Агат"

	- сброс		- управление
	- перевод строки	   	- управление движением курсора вверх, вниз, вправо, влево
	- повторение		
	- регистр		
	- редактирование		

Строка на экране дисплея содержит 32 символа, допустимая строка программы - 255 символов.

Для работы в режиме микрокалькулятора перед арифметическим выражением необходимо поставить PRINT или знак вопроса (?).

## 1.2. Дискровая операционная система

Дискровая операционная система (ДОС) ПЭВМ "Агат" предназначена для создания, сопровождения и уничтожения наборов данных (файлов) пользователя на персональном или гибком магнитном диске (ГМД) ПЭВМ.

ДОС позволяет работать с файлами трех типов:

А - БИСИК-программа;

В - двоичный;

Т - текстовой.

Объем ОП, необходимой для работы ДОС, составляет от 10 до 21 Кбайт.

Необходимым для работы ДОС является наличие в постоянном запоминающем устройстве (ПЗУ) программы "Системный монитор".

ДОС обеспечивает обмен с ГМД со скоростью 2,5 Кбайт/с. При начальной загрузке ДОС программа переносится с гибкого магнитного диска (ГМД) в оперативное запоминающее устройство (ОЗУ). Процесс загрузки занимает не более одной минуты, при этом настройка и загрузка программ происходят автоматически и не требуют вмешательства оператора. После загрузки в стандартном варианте ДОС подключается диалог БИСИК. При начальной загрузке автоматически устанавливается номер разъема (a), в котором установлен контроллер (2-6) и номер привода (d), под которым подключен ГМД (1-2). Для ПЭВМ с одним ГМД a всегда равно единице.

## I.2.I. Общие команды ДОО

Формат команды	Выполняемое действие	Примечание
1. CATALOG {, Ss} {, Dd} {, Vv}	Чтение каталога	В каталоге отображается: тип файла (A, B, T), размер файла в секторах (1 сектор - 256 байт), имя файла
2. INIT имя файла {, Ss} {, Dd} {, Vv}	Инициализация ГМД	Время инициализации ГМД - 1 мин, при успешной работе сообщений не выдается и в каталоге ГМД можно увидеть единственный файл с указанным именем
3. RENAME файл 1, файл 2 {, Ss} {, Dd} {, Vv}	Переименование файлов	Файл 1 - старое имя файла, файл 2 - новое имя файла
4. DELETE имя файла {, Ss} {, Dd} {, Vv}	Уничтожение файлов	
5. LOCK имя файла {, Ss} {, Dd} {, Vv}	Защита файла на запись	Файл, защищенный от записи, в каталоге помечается символом *
6. UNLOCK имя файла {, Ss} {, Dd} {, Vv}	Отмена защиты от записи	

Обозначения, используемые в форматах команд:

s - номер разъема;

d - номер привода;

v - номер тома;

{ } - фигурные скобки можно опустить, при этом используются значения, установленные предыдущим приказом.

s, d, v - принимаются по умолчанию при наличии одного НГМД.

Имя файла должно начинаться с буквы, может содержать до 30 любых символов, включая знаки, и специальные символы, кроме символа " , " .



### 1.2.2. Работа с файлами типа А

Файлы типа А на ГМД содержат программы, составленные на языке программирования БЕИСИК.

Формат команды	Выполняемое действие	Примечание
1. LOAD имя файла {,Ss} {,Dd} {,Vv}	Загрузка программы с ГМД	Программа и данные, находящиеся в ОП, теряются
2. SAVE имя файла {,Ss} {,Dd} {,Vv}	Запись программы из памяти на ГМД	Если на ГМД есть файл с таким же именем, то он затирается новой информацией
3. RUN имя файла {,Ss} {,Dd} {,Vv}	Загрузка с ГМД и запуск программы	Программа и данные, находящиеся в ОП, теряются
4. CHAIN имя файла {,Ss} {,Dd} {,Vv}	Загрузка с ГМД и запуск программы	Программа в ОП теряется, данные передаются запускаемой программе

## 1.2.3. Работа с файлами типа В

Файлы типа В содержат программы, составленные на машинном языке ассемблера.

Формат команды	Выполняемое действие
1. BLOAD имя файла {,Aa} {,Ss} {,Dd} {,Vv}	Загружает файл с адреса. Если адрес не указан, то загрузка выполняется с адреса, указанного при записи на ГМД
2. BSAVE имя файла ,Aa, L1 {,Ss} {,Dd} {,Vv}	Записывает указанную зону памяти на ГМД
3. BRUN имя файла {,Aa} {,Ss} {,Dd} {,Vv}	Загружает файл с адреса А аналогично BLOAD и передает управление на начальный адрес файла

Обозначения, используемые в форматах команд:

A — начальный адрес файла;

L — длина двоичного файла.

Необходимо указывать десятичное значение (L4906), либо шестнадцатеричное с признаком "H" (AH1000).

### Г.2.4. Работа с файлами типа T

Предусмотрено два типа текстовых файлов:

- файлы с последовательным доступом (сплошная последовательность символов);
- файлы с прямым доступом (с записями фиксированной длины).

Формат оператора	Выполняемое действие	Примечание
<p>1. OPEN имя файла {,Ss} {,Dd} {,Vv}</p> <p>OPEN имя файла, Lj {,Ss} {,Dd} {,Vv}</p>	<p>Открытие файла для последовательного доступа</p> <p>Открытие файла для прямого доступа</p>	<p>Открытие отсутствующего файла приводит к созданию пустого файла (<math>&lt;1 \leq j \leq 32767</math>)</p> <p>Открытие файла с именем, имеющимся на ГМД, позиционирует файл на начало</p>
<p>2. CLOSE имя файла {,Ss} {,Dd} {,Vv}</p> <p>CLOSE имя файла {,Ss} {,Dd} {,Vv}</p>	<p>Заккрытие файла для последовательного доступа</p> <p>Заккрытие файла для прямого доступа</p>	<p>Прерывает логическую связь с открытым файлом. Последующие операторы PRINT или INPUT считывают или выводят данные на экран</p>
<p>3. WRITE имя файла {,Bb}</p> <p>WRITE имя файла {,Rr} {,Bb}</p>	<p>Запись в файл для последовательного доступа</p> <p>Запись в файл для прямого доступа</p>	<p>Устанавливает связь с открытым файлом, инициирует вывод данных в файл</p> <p>Последующие операторы PRINT выводят данные в открытый файл</p>

Формат оператора	Выполняемое действие	Примечание
4. READ имя файла {,Bb}  READ имя файла {,Rr} {,Bb}	Чтение файла для последовательного доступа  Чтение файла для прямого доступа	После выполнения операторов OPEN и READ весь ввод с помощью программы "Системный монитор" или INPUT и GET БЕЙСИКА выполняется из открытого текстового файла
5. EXEC имя файла {,Rr} {,Ss} {,Dd} {,Vv}	Исполнение из текстового файла, если файл содержит программу	После EXEC весь ввод с помощью программы "Системный монитор" выполняется из текстового файла до тех пор, пока в нем не встретится оператор CLOSE без предшествующего ему OPEN.
6. POSITION имя файла {,Rr}	Позиционирование файла	Пропуск двух записей в файле. Выключает режим чтения и записи
7. APPEND имя файла {,Ss} {,Dd} {,Vv}	Нарастивание файла с последовательным доступом	Открывает файл и позиционирует его на последний символ. Последующая команда WRITE будет нарашивать файл

Обозначения, используемые в форматах операторов:

b - номер байта в файле;

r - номер записи в файле.

### 1.2.5. Программные команды ДОС

Программы на языке БЕЙСИК передают команды ДОС оператором PRINT, при этом работает подпрограмма "Системного монитора", перед которой выводится символ с кодом  $\alpha 84$  (CHR  $\alpha$ (4) в языке БЕЙСИК. Управляющий код необходимо вводить с первой позиции строки, указав оператор PRINT в следующем формате:

```
110 PRINT:PRINT _CHR $\alpha$ (4); "CATALOG"
120 PRINT:PRINT _CHR $\alpha$ (4); "RUN SAVOT"
```

### 1.3. Интерпретатор языка БЕЙСИК

Интерпретатор языка БЕЙСИК занимает 14 Кбайт, размещается в дополнительной ОП, служит для выполнения программ и директив языка БЕЙСИК. При использовании ГМД необходимо иметь в ОП ПЭВМ "Агат" ДОС.

## 2. ОСНОВЫ ПРОГРАММИРОВАНИЯ НА ЯЗЫКЕ БЕЙСИК

### 2.1. Представление данных

#### Типы переменных и констант

Тип данных	Диапазон	Примеры записи	Комментарий
Вещественные (AB)	$10^{-37} <  x  < 10^{37}$	50.8 -128.14 4251 0,56 9.11E2 1.6E-19	Набор из одной или нескольких десятичных цифр, положительных, отрицательных, целых или дробных
Целые (AB%)	+/-32767	28% -985% 0%	Набор десятичных цифр, оканчивающихся знаком процент (%)
Строковые (литерал или текстовые) (AB $\alpha$ )	0-255 символов	ПРИВЕТ A=СУММЕ C+D SIGMA R20 A#BC N--(V)	Набор буквенно-цифровых или специальных символов. Имя строковой константы или переменной должно оканчиваться знаком " $\alpha$ "

Данные могут быть представлены простыми переменными, массивами и константами.

Примеры использования переменных и констант в программе:

1) 10 A% = 10 - переменной A% будет присвоена целая часть значения 10.

2) 20 A = 10.2 - переменной A будет присвоено дробное значение 10.2.

3) 30 A\$ = "АГАТ-БЕИСИК" - переменной A\$ будет присвоено символьное значение, указанное в кавычках "АГАТ-БЕИСИК".

## 2.2. Представление операций

При составлении математических или логических выражений в программе используются следующие операции.

### Арифметические операции

- = присваивание (переменная = выражение)
- взятие с обратным знаком, вычитание
- + сложение
- \* умножение
- / деление
- ^ возведение в степень

### Операции сравнения и логические

- = равно
- < > не равно
- < меньше
- > больше
- <= меньше или равно
- >= больше или равно
- NOT логическое НЕ
- AND логическое И
- OR логическое ИЛИ

Логическая истина тождественна арифметической единице, ложь - нулю.

Для строковых переменных применимы отношения "=" и "< >".

Запись арифметических и логических операций  
на языке программирования Бейсик и в математике

Математическое выражение	Клавиши ПЭВМ "Агат"	Предложения в программе Бейсик	Выполняемое действие
$a^b$	<input type="text" value="^"/>	$A^B$	Возведение в степень
$a \cdot b$	<input type="text" value="*"/>	$A*B$	Умножение
$a : b$	<input type="text" value="/"/>	$A/B$	Деление
$a + b$	<input type="text" value="+"/>	$A+B$	Сложение
$a - b$	<input type="text" value="-"/>	$A-B$	Вычитание
$a > b$		$A > B$	Больше
$a \geq b$		$A \geq B$	Больше или равно
$a < b$		$A < B$	Меньше
$a \leq b$		$A \leq B$	Меньше или равно
$a = b$		$A = B$	Равно
$a \neq b$		$A <> B$	Не равно

2.3. Элементарные математические функции,  
изучаемые в школе

Функция	Обозначение	Выполняемое действие
$\sin x$	SIN(X)	Вычисляет синус аргумента X
$\cos x$	COS(X)	Вычисляет косинус аргумента X
$\operatorname{tg} x$	TAN(X)	Вычисляет тангенс аргумента X
$\operatorname{arctg} x$	ATN(X)	Вычисляет арктангенс аргумента X
$\sqrt{x}$	SQR(X)	Положительный квадратный корень аргумента X
$e^x$	EXP(X)	Вычисляет показательную функцию $e^x$ (где $e = 2,71228$ , $X \leq 37$ )
$ x $	ABS(X)	Вычисляет модуль X (X - любое число)
	INT(X)	Вычисляет целую часть аргумента X
$\ln(x)$	LOG(X)	Натуральный логарифм X (X > 0) принимает следующие значения: +1 при X=0 -1 при X=0 0 при X=0
	SGN(X)	

Примечание. X выражается в радианах.

## 2.4. Составление математических выражений.

### Порядок выполнения операций

Выражение – это формула для вычисления значений. Выражения могут быть переменными, функциями, постоянными значениями и комбинациями этих элементов, связанных знаками операций. При записи для указания порядка выполнения операций могут использоваться круглые скобки. В языке БЕИСИК допустимы выражения двух типов: числовые и строковые.

Числовым выражением называется произвольная конечная последовательность числовых операций. Результатом числовой операции является число. Числовые операции подразделяются на следующие группы:

- арифметические операции;
- операции сравнения;
- логические операции;
- вызов функции.

Строковые или текстовые выражения состоят из текстовой константы или текстовой переменной.

### Приоритет выполнения действий в выражениях, написанных на языке БЕИСИК

Приоритет	Обозначение	Выполняемое действие
1	( )	Действия в скобках
2	SQR SIN TAN	Вычисление стандартных встроенных функций
3	^	Возведение в степень
4	-	Минус, изменяющий знак стоящей за ним величины
5	* /	Умножение, деление
6	+ -	Сложение, вычитание

Операции, имеющие равный приоритет, выполняются слева направо.



## 2.5. Работа в режиме калькулятора

При необходимости выполнения команды сразу, ее пишут без номера с оператором PRINT. Вместо слова PRINT можно набрать знак вопроса без кавычек. Операторы PRINT и присваивание воспринимаются интерпретатором и обеспечивают режим калькулятора, т.е. режим непосредственного выполнения команд.

```
] Aα = "ОПТОВАЯ ЦЕНА"  
] Bα = "РОЗНИЧНАЯ ЦЕНА"  
] Cα = "НАЦЕНКА"  
] K = 10  
] V = K * 0,02  
] ? Bα ; "=" ; Aα ; "+" ; Cα  
  РОЗНИЧНАЯ ЦЕНА = ОПТОВАЯ ЦЕНА + НАЦЕНКА  
] ? Bα ; "=" ; B+K ; "РУБ"  
  РОЗНИЧНАЯ ЦЕНА = 10,2 РУБ
```

} Видим на экране ВКУ

## 3. ПРЕДСТАВЛЕНИЕ КОМАНД В ЯЗЫКЕ БИГИК

Команды в языке БИГИК подразделяются на четыре типа:

- команды, начинающиеся с ключевого слова (системные команды, команды редактирования, команды для описания массивов и строк, команды ввода-вывода данных, графические команды;
- команды присваивания, начинающиеся с имени переменной;
- команды ассемблера, начинающиеся с символа "!";
- команды отладочного набора, начинающиеся с символа "\*".

В данном справочнике рассматриваются команды только первых двух типов.

В командах первых двух типов различные директивы отделяются друг от друга двоеточием (:), внутри команды слова отделяются друг от друга пробелами.

Строки программы нумеруются, максимальный номер - не более 65535.

### 3.1. Системные команды

Обозначение команды	Выполняемое действие
1. LOAD	Загрузка программы с МЛ
2. SAVE	Запись программы на МЛ
3. NEW	Установка начального состояния ОП, стирание программы и данных
4. RUN	Запуск программы со строки с наименьшим номером
RUN 200	Запуск с конкретной строки
5. STOP	Останов программы с выдачей текста строки, в которой произошел останов
6. CONT	Продолжение выполнения программы после STOP
7. END	Конец программы
8. CALL X	Вызов подпрограммы по адресу X
9. URS (X)	Вызов подпрограммы с передачей значений. Значение X помещается в ячейках (157-163). Адрес X подпрограммы должен быть помещен в ячейки (11-12)
10. TRACE	Включение режима вывода номера строки при выполнении каждого оператора
11. NOTRACE	Отменена TRACE

### 3.2. Команды редактирования текстов.

#### Оформление программ

#### 3.2.1. Команда LIST - вывод текста программы на экран ВКУ

#### Варианты записи команды LIST

Вариант	Форма записи	Выполняемое действие
1	LIST	Вывод на экран ВКУ всего текста программы
2	LIST N1 LIST 80	Вывод на экран ВКУ строки программы с номером I
3	LIST N1,N2 LIST 20,200	Вывод на экран ВКУ текста программы со строки с номером I до строки с номером 2
4	LIST N1, LIST 40	Вывод на экран ВКУ текста программы со строки с номером I до конца текста
5	LIST N2 LIST 60	Вывод на экран ВКУ текста программы с начала до строки с номером 2

### 3.2.2. Команда DEL - стирание строки программы из ОП

Команда DEL всегда записывается с двумя параметрами, указанными через запятую.

Возможные варианты команды DEL

Вариант	Формат команды	Выполняемое действие
1	DEL N1,N2	Удаление из текста программы группы строк с номером 1 до номера 2
2	DEL N1,N1	Удаление из текста программы одной строки с номером 1

Чтобы удалить строку без использования DEL необходимо после обратной квадратной скобки указать номер удаляемой строки и нажать клавишу  I .

### 3.2.3. Команда REM - введение комментариев в программу

REM - используется для введения комментариев в программу. Комментарий до конца строки при выполнении программы игнорируется.

REM <текст> - текст поясняет суть программы.

Пример включения комментария в программу:

```
50 REM ПРОГРАММА ВЫЧИСЛЕНИЯ КОРНЕЙ КВАДРАТНОГО УРАВНЕНИЯ
```

### 3.2.4. Работа с курсором, редактирование строки

Для обеспечения наглядности изображения, улучшения восприятия текста используют команды:

HOME - очистка экрана, курсор помещается в левом верхнем углу экрана ;

VTAB Y - устанавливает курсор на строку экрана с номером (0-31);

HTAB X - передвигает курсор на X-ю позицию текущей экранной строки. Следующий оператор PRINT будет выводить информацию с этой позиции. Нумерация строк и позиций на экране сверху вниз и слева направо ;

TAB(X) - элемент оператора PRINT, аналогичен HTAB ;

SPC(X) - элемент оператора PRINT, вывод X пробелов.

На ПЭВМ "Агат" возможно высвечивание информации на экран ВКУ в трех режимах: нормальном, инверсном и мерцающем.

NORMAL - включает прямой (светлый по черному) режим вывода текста  
INVERSE - включает инверсный (черный по светлому фону) режим вывода текста  
FLASH - включает мерцающий режим вывода текста.

Примечание. По умолчанию принимается прямой режим.


SPEED=X - задает скорость вывода текста (0-255)  
RIBBON= <номер цвета> - задает цветное изображение текста на экране.  
Номера цветов приведены в таблице на с.28. Каждый новый оператор RIBBON задает новый цвет. Для вывода текста на цветной экран можно использовать следующую последовательность операторов:

10 HOME:INVERSE:RIBBON=N:HOME

где N - номер цвета. Экран окрасится в указанный цвет.

#### 4. ОПЕРАТОРЫ ВВОДА-ВЫВОДА ИНФОРМАЦИИ НА ЭКРАН ВКУ

Операторы ввода-вывода информации на экран ВКУ позволяют организовать передачу данных с внешнего устройства во внутреннюю память и обратно.

Формат оператора	Выполняемое действие	Комментарий
<p>INPUT "запрос"; &lt;список ввода &gt;                      INPUT &lt;список ввода &gt;</p> <p>INPUT "НАЧИНАЙТЕ ВВОД"; A                      INPUT 1%,X,A α</p>	<p>Ввод целого, вещественного и строкового значений с клавиатуры или заменяющего ее устройства</p>	<p>Запрос, состоящий из текстовых данных и пробелов, необходимо заключать в кавычки. Он дает пояснение к вводимой информации.</p> <p>Список ввода - последовательность имен переменных и/или элементов массивов, разделенных запятыми.</p> <p>Ввод значений с клавиатуры осуществляется после появления на экране ВКУ знака вопроса (?). Окончание ввода - нажатие клавиши </p>
<p>GET &lt;имя литерной переменной &gt;                      GET A α</p>	<p>Ввод одного символа с клавиатуры</p>	<p>Выполнение программы приостанавливается и ПЭВМ ждет сигнала с клавиатуры. В ответ можно нажать любую информационную клавишу. Введенный символ на экране не изображается; после его ввода выполнение программы продолжается</p>
<p>READ X</p>	<p>Присваивает X значение очередного элемента списка данных, определенного оператором DATA</p>	<p>Элементы списка отделяются друг от друга запятой</p>

Формат оператора	Выполняемое действие	Комментарий
DATA <СПИСОК ДАННЫХ > DATA 1,5E-7, СТРОКА, "ТЕКСТ, ТЕКСТ"		Если запятая (,) есть внутри элемента, то он заключается в кавычки
RESTORE	Устанавливает в начальное положение указатель списка данных	Следующим будет прочитан первый элемент списка
PRINT <СПИСОК ВВОДА >  PRINT (печатать)  PRINT "ВВЕДИТЕ НОМЕР"  PRINT A%, I%, X  PRINT X, SQR(X)  PRINT "значение X=", X  PRINT	Вывод на экран ЭВМ значений элементов списка  Пустой оператор вызывает переход на следующую печатную строку.  Печать сообщений  Печать вычислений значения X и X  Печать сообщений и результатов вместе  Пропуск строки	Элементы списка разделяются запятой (,) или точкой с запятой (;)  Запятая (,) задает печать в очередную зону экрана (размер зоны кратен восьми разрядам)  Точка с запятой (;) задает вывод информации подряд без пробела

Примечание. Операторы чтения (READ) и определения данных (DATA) всегда используются парно, т.е. идет присвоение значений переменным, перечисленным в READ, значения переменных берутся из DATA. В оператор данных помещается только число, а не выражение. Если данных не хватает, то выдается сообщение об ошибке.

Пример программы:

```
10 NORMAL : HOME : RIBBON=1 :
20 VTAB 2 : HTAB 7;
30 PRINT "ВОПРОС 12"; GQ ;
40 NORMAL : RIBBON=2:
50 VTAB 4 : HTAB 1:
60 PRINT "ЗАПРОСИТЕ ИНФОРМАЦИЮ О ТЕКУЩЕМ"
70 VTAB 6 : HTAB 1:
80 PRINT "СОСТОЯНИИ УСТРОЙСТВ"
90 VTAB 8 : RIBBON=4
100 FOR I=1 TO 32
110 PRINT "*"; GQ :
120 NEXT
130 RIBBON=3
140 INPUT "ВВЕДИТЕ ОТВЕТ"; AQ :
150 IF A Q = "D U" THEN 240
160 RIBBON=6
170 PRINT "ОТВЕТ НЕВЕРНЫЙ"
180 RIBBON=7
190 PRINT "ПРАВИЛЬНЫЙ ОТВЕТ"
200 PRINT "D U"
210 FOR I=1 TO 3000
220 NEXT
230 GOTO 270
240 INVERSE : RIBBON=5:
250 PRINT "ОТВЕТ ВЕРНЫЙ"
260 FOR I=1 TO 3000
270 .....
```

Примеры ввода в программу совокупности данных (DATA и READ) и восстановление блока данных (RESTORE):

```
a) 10 DATA 5,6,7,8,9,10
    20 READ A,B,C,E,K,M
b) 10 DATA 5,6,7,8,9,10
    20 READ A
c) 10 DATA 8,4,6,7,11
    20 LET S=0: LET K=1
    30 IF K > 5 THEN 80
    40 READ Y
```

```
50 LET S=S+Y
```

```
60 LET K=K+1
```

```
70 GOTO 30
```

```
80 PRINT S
```

```
90 END
```

```
d) 10 DATA 1,2,3,4
```

```
20 READ A,B,C,P
```

```
30 PRINT A+B+C+P
```

```
40 RESTORE
```

```
50 READ K,M,Y,X
```

```
60 PRINT K*M*Y*X
```

```
70 END
```

Фрагменты программ с оператором PRINT

```
a) 10 INPUT X
```

```
20 IF X=0 THEN 60
```

```
30 LET Y=X ^ 2+5*X
```

```
40 PRINT "X="; X, "Y="; Y
```

```
50 GOTO 10
```

```
60 PRINT "ПРОГРАММА ЗАКОНЧЕНА"
```

```
70 END
```

```
b) 10 PRINT "A=";
```

```
20 PRINT A
```

```
30 PRINT A+B
```

```
40 PRINT A,B,C
```

```
50 PRINT "РЕЗУЛЬТАТ"
```

```
c) 10 DATA 5,10,15,20
```

```
20 READ A,B,C,D
```

```
30 LET P=(A+B+C)*D
```

```
40 PRINT A,B,C,D
```

```
50 PRINT "РЕЗУЛЬТАТ : P="; P
```

```
60 END
```



## 5. ОПИСАНИЕ МАССИВОВ. ОПЕРАТОРЫ ЯЗЫКА РАБОТЫ СО СТРОКОВЫМИ ДАННЫМИ

Массивы используются в программе для временного хранения данных в ОП. Операторы работы со строковыми данными позволяют формировать строки, сортировать строки по алфавиту, "вырезать" из строки последовательность символов и т.д.

Формат оператора	Выполняемое действие	Пример	Комментарий
1. DIM  DIM(X,Y)  DIM(X,Y,Z)	<p>Описывает массивы, резервирует места в ОП для числовых и строковых данных</p> <p>Двухмерный массив А с диапазоном индексов <math>\phi-X, \phi-Y</math></p> <p>X - количество элементов в строке Y - количество элементов в столбце</p> <p>Трехмерный массив А с диапазоном индексов O-X, O-Y, O-Z</p>	<p>DIM A(10),A(10,10) DIM A%(10),A2%(5,5) DIM A%(10),A2%(6,6)</p> <p>DIM A(12),C%(4), B(2,3) DIM B(3,3,3)</p>	<p>DIM содержит список имен массивов, разделенных запятыми, с максимальными границами значений индексов, заключенных в скобки. Определяется в программе до обращения к данным</p>
2. LEN (A%)  STR % (X)  VAL (A%)	<p>Значение, равное числу символов в строке аргумента</p> <p>Значение, строка, содержащая текстовое представление целого или вещественного аргумента</p> <p>Значение, вещественное, текстовое представление которого расположено в начале аргумента (до первого нечислового символа)</p>	<p>X=LEN (A%)</p> <p>A%=STR%(%26) PRINT A%</p> <p>A%= "150 РУБ." C=VAL (A%)</p>	<p>В переменной X количество символов в строке А</p> <p>В переменной А% - 38</p> <p>В переменной с - вещественное представление числа 150</p>

Формат оператора	Выполняемое действие	Пример	Комментарий
CHR α(X)	Значение символа КОИ-8, код которого равен X	Aα=CHRα(40)	В переменной A символ, совпадающий с кодом 40
ASC (A α)	Код КОИ-8 первого символа строки аргумента	Aα="Б"; Bα="Г" IF(ASC(Aα))=(ASC(Bα))THEN 50	Сравниваются внутренние коды символов "Б" и "Г"
LEFT α(Aα,X)	Строка из первых X символов аргумента	Aα="КАЛИНИНГРАД" Bα=LEFTα(Aα,7)	Переменная Bα="КАЛИНИН"
RIGHTα (Aα,X)	Строка из последних X символов аргумента	Aα="КАЛИНИНГРАД" Bα=RIGHTα(Aα,4)	Переменная Bα="ГРАД"
MIDα(Aα,X,Y)	Строка из Y символов Aα, начиная с X-го	Aα="КАЛИНИНГРАД" Bα=MIDα(Aα,3,5)	Переменная Bα="ЛИНИН"

Примечание. Для сцепления нескольких строковых данных (переменных) используется знак плюс (+). Например:

```
10 Aα="АГАТ"
20 Bα="БЕЙСИК-"
30 PRINT Bα+Aα
```

на экране будет БЕЙСИК-АГАТ

При работе с массивами необходимо помнить о том, что не объявленные в операторе DIM массивы автоматически (по умолчанию) получают размерность всех индексов от 0 до 10.

До использования массива в программе, его необходимо объявить в операторе DIM. Помнить, что переменные A, A%, A#, A(1), A%(1), A#(1) для интерпретатора различны, а появление в программе массива A(1,2) и массива A(3,4,5) - ошибка.

Размерность массивов можно определять через переменные, вводимые до определения DIM в следующем виде:

```
10 INPUT "ВВЕДИТЕ РАЗМЕРНОСТЬ МАССИВА"; N
```

```
20 DIM(N)
```

N - определит размерность массива.

Пример употребления оператора DIM. Найти максимальный элемент в одномерном массиве.

```
10 PRINT "НАХОЖДЕНИЕ МАКСИМУМА"
```

```
20 REM ВВОД ДАННЫХ (ЭЛЕМЕНТОВ МАССИВА)
```

```
30 DIM A (10)
```

```
40 DATA 1,2,3,4,5,6,100,50,25,10
```

```
50 FOR I=1 TO 10
```

```
60 READ A(I)
```

```
70 NEXT I
```

```
80 PRINT "СОРТИРОВКА"
```

```
90 LET I=1
```

```
100 LET M=A(I)
```

```
110 LET I=I+1
```

```
120 IF I > 10 THEN 150
```

```
130 IF A(I) > M THEN 100
```

```
140 GOTO 110
```

```
150 PRINT "MAX="; M
```

```
160 END
```

Примеры работы со строками

Рассмотрим программу, с помощью которой можно сортировать массив слов по первому символу латинского алфавита.

```
10 DIM M(10)
```

```
20 INPUT "ВВЕДИТЕ КОЛИЧЕСТВО СЛОВ", KOL
```

```
30 PRINT "ВВОДИТЕ ДАННЫЕ"
```

```
40 FOR I=0 TO KOL
```

```
50 INPUT A(I)
```

```

60 NEXT
70 FOR I=2 TO KOL
80 FOR Y=I-1 TO 1 STEP-1
90 IF ASC(LEFT$(M$(I),1)) < ASC(LEFT$(M$(J),1)) THEN A$=M$(I):
    M$(I)=M$(J): M$(J)=A$
100 NEXT J
110 NEXT I
120 FOR I=1 TO KOL
130 PRINT M$(I)
140 NEXT

```

Примеры использования операторов работы со строками

```

10 A$="КАЛИНИНГРАД"
20 B$=LEFT$(A$,7)
30 PRINT B$
40 C$=RIGHT$(A$,4)
50 PRINT C$
60 D$=MID$(A$,3,5)
70 PRINT D$
80 E$=B$+C$
90 PRINT E$

```

30-я строка высветит на экране КАЛИНИН,  
 50-я строка высветит на экране ГРАД,  
 70-я строка высветит на экране ЛИНИН,  
 90-я строка высветит на экране КАЛИНИНГРАД

## 6. ОПЕРАТОРЫ ГРАФИКИ

### 6.1. Обозначение графических режимов

Обозначение	Назначение графического режима	Диапазон допустимых значений	Диапазон номеров страниц
Ц в е т н ы е	GR	Включение графики низкого назначения (64x64)	N = 2 N >= 31
	MGR	Включение графики среднего разрешения (128x128)	N = 7 N > 1

Обозначение	Назначение графического режима	Диапазон допустимых значений	Диапазон номеров страниц
HGR (черно-белый)	Включение графики высокого разрешения (256x256)	0-255	N=1-7

Примечание.

GR = < N >

MGR = < N >

HGR = < N >

где N - номер страницы графического режима размером 2 Кбайт в режиме GR и 8 Кбайт в остальных режимах.

## 6.2. Операторы графического режима

Формат оператора	Выполняемое действие	Пример	Комментарий																		
1. COLOR= <номер цвета>	Устанавливает цвет для последующих операторов	Color=3	<table border="1"> <thead> <tr> <th>Номер цвета</th> <th>Цвет</th> </tr> </thead> <tbody> <tr><td>1</td><td>красный</td></tr> <tr><td>2</td><td>зеленый</td></tr> <tr><td>3</td><td>желтый</td></tr> <tr><td>4</td><td>синий</td></tr> <tr><td>5</td><td>фиолетовый</td></tr> <tr><td>6</td><td>голубой</td></tr> <tr><td>7</td><td>белый</td></tr> <tr><td>8</td><td>черный</td></tr> </tbody> </table>	Номер цвета	Цвет	1	красный	2	зеленый	3	желтый	4	синий	5	фиолетовый	6	голубой	7	белый	8	черный
Номер цвета	Цвет																				
1	красный																				
2	зеленый																				
3	желтый																				
4	синий																				
5	фиолетовый																				
6	голубой																				
7	белый																				
8	черный																				
2. PLOT X,Y  PLOT X1,Y1 TO X2,Y2	<p>Помещает точку текущего цвета в X-ю позицию Y-й строки экрана</p> <p>Линия текущего цвета из точки с координатами (X1,Y1) в точку с координатами (X2,Y2)</p>	<p>5 PLOT 15,15</p> <p>100 PLOT 20,20 TO 100,70</p> <p>120 PLOT 0,0 TO 127,127: TO 0,127: TO 0,0 - ломаная линия</p>	<p>Координаты начальной точки можно не указывать, тогда в качестве начальной используется конечная точка последнего по выполнению оператора PLOT</p> <p>В строке максимум до 256 символов</p>																		
3. SCRN (X,Y)	Высвечивает точку на экране с координатами (X,Y)	15 SCRN(17,20)																			
4. TEXT=N	Устанавливает текстовый режим	TEXT =2	<p>N - номер страницы, переключает программу из графического режима в текстовый.</p> <p>Обращать внимание на номера страниц, для различных исполнений ПЭВМ их значения различны</p>																		

Формат оператора	Выполняемое действие	Пример	Комментарий
5. PDL(X)	Значение, установленное ручкой X-го аналого-цифрового пульта		

- Примечания: 1. Соединение текстовой и графической страниц (рисунок и текст) недопустимо.  
2. Следует помнить, что оператор COLOR обязателен в графическом режиме в отличие от RIBBON в символьном.  
3. В ГВР оператор COLOR определяет не цвет, а частоту точек на экране.  
4. Выход из графического режима в программе осуществляет оператор TEXT=N.

Пример программы построения на ВКУ трех квадратов, вложенных друг в друга, в цветном графическом режиме (128x128 точек).

```
10 MGR=3
20 REM ВКЛЮЧЕНА 3-Я ГРАФИЧЕСКАЯ СТРАНИЦА 128x128
30 COLOR=1:REM УСТАНОВЛЕН КРАСНЫЙ ЦВЕТ
40 PLOT 0,0 TO 0,128 TO 128,128 TO 128,0 TO 0,0
50 REM НАРИСОВАН КВАДРАТ 128x128
60 COLOR=2:REM ЗЕЛЕНый ЦВЕТ
70 PLOT 5,5 TO 5,123 TO 123,123 TO 123,5 TO 5,5
80 REM НАРИСОВАН КВАДРАТ 123x123
90 COLOR=4:REM ЖЕЛТЫЙ
100 PLOT 10,10 TO 10,118 TO 118,118 TO 118,10 TO 10,10
110 REM КВАДРАТ 118x118, ЦВЕТ - ЖЕЛТЫЙ
120 GET A$:REM ПОКА НЕ НАЖАТА КЛАВИША
130 TEXT=15:HOME
140 INPUT "ПОВТОРИТЬ (ДА/НЕТ)?," A$
150 A$="Д" OR A$="D" THEN 10
160 PRINT "ПРИВЕТ!"
```



## 7. ОПЕРАТОРЫ ПЕРЕДАЧИ УПРАВЛЕНИЯ

Формат оператора	Транскрипция и перевод	Выполняемое действие	Пример	Комментарий
1. GOTO <N>	[ <i>gou tu</i> ], идти к	Безусловный переход на оператор с указанным номером	10 GOTO 50	N - номер строки перехода
2. ON(X) GOTO N <sub>1</sub> , N <sub>2</sub> , ..., N <sub>n</sub>		Переключатель-переход на строку с номером из списка в зависимости от значения X	ON(X)GOTO 100, 200	Если X=1, то переход на оператор с номером 100; если X=2, то переход на оператор с номером 200; если X=0, то выполняется следующий оператор программы
Оператор перехода к подпрограмме				
3. GOSUB N	[ <i>GO to subroutine</i> ], Идти к подпрограмме	Оператор перехода к подпрограмме	20 GOSUB 1100	N - номер строки перехода
4. ON X GOSUB N <sub>1</sub> , N <sub>2</sub> , ..., N <sub>n</sub>		Переключатель-вызов, вызывает переход на X-ю из перечисленных подпрограмм		

Формат оператора	Транскрипция и перевод	Выполняемое действие	Пример	Комментарий
5. RETURN	[return], возвратиться	Возврат к оператору, следующему за последним GOSUB	10 GOSUB 50 20 A=B+C 50 B=120+KOL 60 RETURN	Это логический конец подпрограммы, передача управления на оператор, следующий за GOSUB т.е. на оператор 20.
6. Оператор перехода по условию IF...THEN IF < переменная > знак логического отношения < число > THEN < номер строки перехода >	[if...then], если...то	Указание на- правления пе- рехода при со- блюдении усло- вия	5 IF I>20 THEN 20 10 Y=X*6 20 Y=X*5+5*X	

Примечание. При использовании операторов goto, gosub не допускать перехода на несуществующую строку (ошибка и останов интерпретации). В операторах-переключателях, если значение переключателя больше числа номеров строк в списке или равно нулю, то переход осуществится на следующий оператор.

Пример обращения к подпрограмме.

а) 60 GOSUB 140

⋮

140 X=Y\*Y

150 P=X+5

160 RETURN

б) ВЫЧИСЛИТЬ  $A=B+C$ , если  $\begin{cases} B=1; C=2 \\ B=3; C=4 \end{cases}$

10 LET B=1: LET C=2

20 GOSUB 100

30 LET B=3: LET C=4

40 GOSUB 100

50 END

100 LET A=B+C

110 PRINT "A="; A, "B="; B, "C="; C

120 RETURN

## 8. ОРГАНИЗАЦИЯ ЦИКЛОВ

При составлении алгоритмов решения практических задач нередко возникают случаи, когда приходится неоднократно повторять одни и те же предписания. Многократно повторяемые участки вычислений называются циклами. В БЕЙСИКЕ для реализации циклических процессов имеются специальные операторы цикла.

Формат оператора цикла	Пример	Комментарий
1. FOR < переменная > = < начальное значение > TO < конечное значение > STEP < шаг > ... NEXT < переменная >	<pre>10 FOR X=1 TO 10 STEP 3 . . . . . 50 NEXT X</pre>	Повторить выполнение программы между операторами FOR и NEXT
<pre>FOR f : , для TO tu : , до STEP step , шаг NEXT [nekst] , переменная</pre>	<pre>10 FOR I=3 TO 10 STEP 4 . . . . . 50 NEXT I 40 FOR I=999 TO 1 . . . STEP -3 60 NEXT</pre>	Значение шага может быть положительным (без знака) и отрицательным со знаком (-), если шаг не указан, то автоматически он равен единице
2. IF < переменная > = < число > THEN < номер строки перехода > ... GOTO < номер строки перехода > IF [if] , если THEN [den] , тогда GOTO [sou tu:] , ИДИ К	<pre>10 IF I=20 THEN 50 . . . 40 GOTO 10</pre>	Передать управление согласно значению выражения

Формат оператора цикла	Пример	Комментарий
<p>3. IF (логическое выражение) THEN K</p>	<pre>IF X=1. THEN PRINT Y IF X=0 THEN 100</pre>	<p>Последовательность операторов после THEN и до конца строки, выполняется только при истинности логического выражения IF. Иначе выполнение продолжается со следующей строки</p>
<p>4. IF NOT (логическое выражение) THEN K.</p> <p>K - номер строки, до которой находятся все операторы, которые нужно выполнить в случае истинности логического выражения</p>	<pre>10 FOR I=0 TO 10 20 IF A(I)=5 THEN I=10: NEXT:GOTO 40 30 NEXT:PRINT "HI"; 40 PRINT "HAIEN!" 50 END</pre>	<p>Если последовательность операторов, которые необходимо выполнить при истинности логического выражения не умещается в одной строке, можно использовать этот прием</p>

Примечание. Операторы FOR и NEXT употребляются всегда парой, причем FOR указывает начало цикла, а NEXT - конец циклического участка, если шаг изменения параметра цикла равен единице, то оператор STEP <число> можно опустить. Не разрешается передавать управление внутрь цикла

Примеры программ с использованием циклов.

а) напечатать первые сто положительных чисел натурального ряда вместе с их квадратными корнями

```
10 LET X=1 - подготовка цикла
20 PRINT X, SQR(X) - тело цикла
30 LET X=X+1 -- модификация X
40 IF X=100 THEN 20 - проверка выхода на конец программы
50 END
```

б) сложить первые N целых чисел

```
10 READ N
20 LET S=0
30 FOR K=1 TO N
40 LET S=S+K
50 NEXT K
60 PRINT S
70 GOTO 10
80 DATA 3, 10, 0
90 END
```

## 9. ВВОД-ВЫВОД ДАННЫХ НА ГМД

Команды ввода-вывода данных на ГМД выдаются в программе в формате программного обращения через оператор PRINT

```
10 PRINT CHR$(4); "команда"
```

Предусмотрено два типа текстовых файлов (в каталоге помечается тип T): файлы с последовательным доступом и файлы с прямым доступом.

В последовательном файле хранится сплошная последовательность символов с записями переменной длины.

Прямой файл содержит записи фиксированной длины.

Для создания текстового файла и чтения данных из текстового файла используются следующие операторы:

OPEN - открыть файл с указанным именем;

READ/WRITE - читать или писать из файла (инициируют ввод-вывод данных с ГМД);

INPUT/PRINT - считать данные с ГМД;

CLOSE - закрыть файл после обработки данных.

## 9.1. Создание файла

1. OPEN открывает файл типа T на ГМД с указанным именем.  
Формат оператора:

```
10 PRINT CHR$(4); "OPEN < имя файла >"
```

Пример.

```
10 PRINT CHR$(4); "OPEN IMF"
```

Можно имя файла вводить с клавиатуры:

```
10 INPUT "ВВЕДИТЕ ИМЯ ФАЙЛА"; F$
```

```
20 PRINT CHR$(4); "OPEN"; F$
```

Будет открыт файл с именем, введенным в F\$.

2. WRITE включает запись данных на ГМД в файл, указанный в операторе OPEN. После оператора WRITE все операторы PRINT в программе будут осуществлять запись данных в файл.

Формат оператора:

```
10 PRINT CHR$(4); "WRITE < имя файла >"
```

Пример. В F\$ имя файла введено с клавиатуры.

```
30 PRINT CHR$(4); "WRITE"; F$
```

```
40 PRINT "ПРОБУЕМ СОЗДАТЬ ФАЙЛ"
```

```
50 PRINT "ИВАНОВ"
```

```
60 PRINT "ПЕТРОВ"
```

} Запись  
данных  
в файл

3. CLOSE закрывает созданный файл с указанным именем и устанавливает признак конца данных (файла).

CLOSE используется по окончании создания файла. Последующие операторы PRINT после CLOSE в программе выводят информацию на экран.

Формат оператора:

```
10 PRINT CHR$(4); "CLOSE < имя файла >"
```

Пример.

```
70 PRINT CHR$(4); "CLOSE"; F$
```

## 9.2. Чтение данных из файла

Пример.

```
10 INPUT "ВВЕДИТЕ ИМЯ ФАЙЛА, ИЗ КОТОРОГО ХОТИТЕ ПРОЧИТАТЬ ДАННЫЕ";
```

```
IM $
```

```
20 PRINT CHR$(4); "OPEN"; IM $
```

```
30 PRINT CHR$(4); "READ"; IM $
```

```
40 INPUT A $ } Чтение
```

```
50 INPUT B $ } данных
```

```
60 INPUT C $ } из файла
```

```

70 PRINT CHR$(4); "CLOSE"; IM
80 PRINT A
90 PRINT B
100 PRINT C
110 END

```

### 9.3. Дозапись данных в созданный файл

Если необходимо дозаписать данные в конец уже созданного файла, то вместо оператора OPEN используется оператор APPEND

Формат оператора:

```
10 PRINT CHR$(4); "APPEND - имя файла."
```

#### Пример.

```

10 INPUT "ВВЕДИТЕ ИМЯ ФАЙЛА"; IM
20 PRINT CHR$(4); "APPEND"; IM
30 PRINT CHR$(4); "WRITE"; IM
40 INPUT "ВВЕДИТЕ НОВЫЕ ДАННЫЕ ИЛИ END"; A
50 IF A="END" THEN 80
60 PRINT A
70 GOTO 40
80 PRINT CHR$(4); "CLOSE"; IM

```

### 9.4. Общие операторы, используемые при вводе-выводе данных на ГМД

При использовании операторов MON, C, I, O и NOMON C, I, O можно отслеживать выполнение операторов ввода-вывода на ГМД в процессе выполнения программы.

Формат операторов:

$$\left\{ \begin{array}{l} \text{MON} \\ \text{NOMON} \end{array} \right\} \text{ C, I, O}$$

MON - включает процесс отслеживания выполнения операторов;

NOMON - выключает процесс отслеживания выполнения операторов;

C - операторы ввода-вывода данных;

I - операторы ввода данных;

O - операторы вывода данных.

Указание хотя бы одного операнда C, I, O обязательно, например

10 MON I включает вывод операторов ввода на экран.



Пример создания файла.

Первой записью в файле должно быть количество записей в файле, последующие записи содержат фамилии учеников. Данные вводить с клавиатуры в DIM. Введенные данные записать на ГМД.


```
5 DIM Aα(30)
10 HOME
20 INPUT "ВВЕДИТЕ ИМЯ ФАЙЛА"; IMF α
30 INPUT "ВВЕДИТЕ КОЛИЧЕСТВО ЗАПИСЕЙ В ФАЙЛЕ"; KOL
40 PRINT "ВВЕДИТЕ ДАННЫЕ"
50 FOR I=0 TO KOL-1
60 INPUT Aα(I)
70 NEXT I
80 PRINT CHRα(4); "OPEN"; IMF α
90 PRINT CHRα(4); "WRITE"; IMF α
100 PRINT KOL - первая запись в файле
110 FOR I=0 TO KOL-1
120 PRINT Aα(I)
130 NEXT I
140 PRINT CHRα(4); "CLOSE"; IMF
```

Пример чтения данных из файла, созданного в предыдущем примере.

```
10. DIM Aα(30)
20 INPUT "ВВЕДИТЕ ИМЯ ФАЙЛА"; IMF α
30 PRINT CHRα(4); "OPEN"; IMF α
40 PRINT CHR (4); "READ"; IMF α
50 INPUT KOL - количество записей в файле в KOL
60 FOR I=0 TO KOL-1
70 INPUT Aα(I)
80 NEXT I
90 PRINT CHRα(4); "CLOSE"; IMF α
100 FOR I=0 TO KOL-1
110 PRINT Aα(I)
120 NEXT I
```

## 10. ПРАВИЛА ЗАПИСИ ПРОГРАММЫ НА ЯЗЫКЕ БЕЛСИК

1. В начале каждой строки программы ставится номер

10	LIST	X=5	
номер строки	оператор		возврат клетки (BK)

2. В памяти машины строки программы всегда располагаются в порядке возрастания номеров. Удобно пронумеровать строки с интервалом 10, чтобы была возможность вставить необходимые пропущенные строки в свободные места.

3. В каждой строке можно записать один или несколько операторов. Операторы отделяются друг от друга двоеточием.

40 LET X=50: LET C=20

4. При наборе программы после записи строки обязательно нажимать клавишу ВК.

5. Переменные в БИМСИКЕ можно обозначать не только буквой, но и буквой с цифрами или двумя буквами:

A1, C5, X0, AY

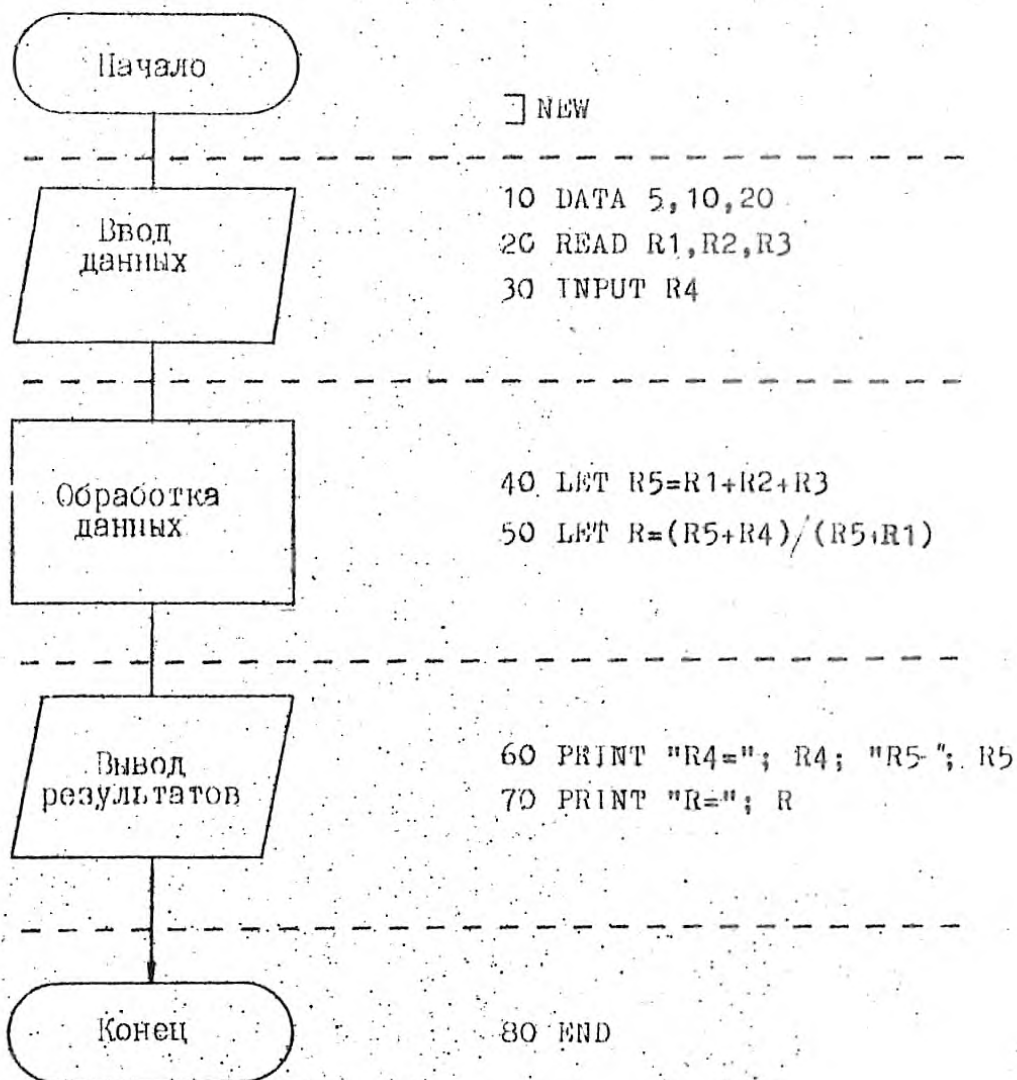
В программе используются только заглавные буквы. Номера строк служат для идентификации строк. Программист может вводить строки программы в любом порядке. Перед началом выполнения они сортируются и редактируются. Каждый оператор начинается со служебного слова, которое определяет тип оператора. Еще раз запомните основные операторы:

LET - пусть,  
IF - если;  
THEN - то;  
GOTO - перейти на;  
READ - прочитать;  
DATA - данные;  
PRINT - напечатать;  
END - конец;  
FOR - для;  
STEP - шаг;  
TO - до.

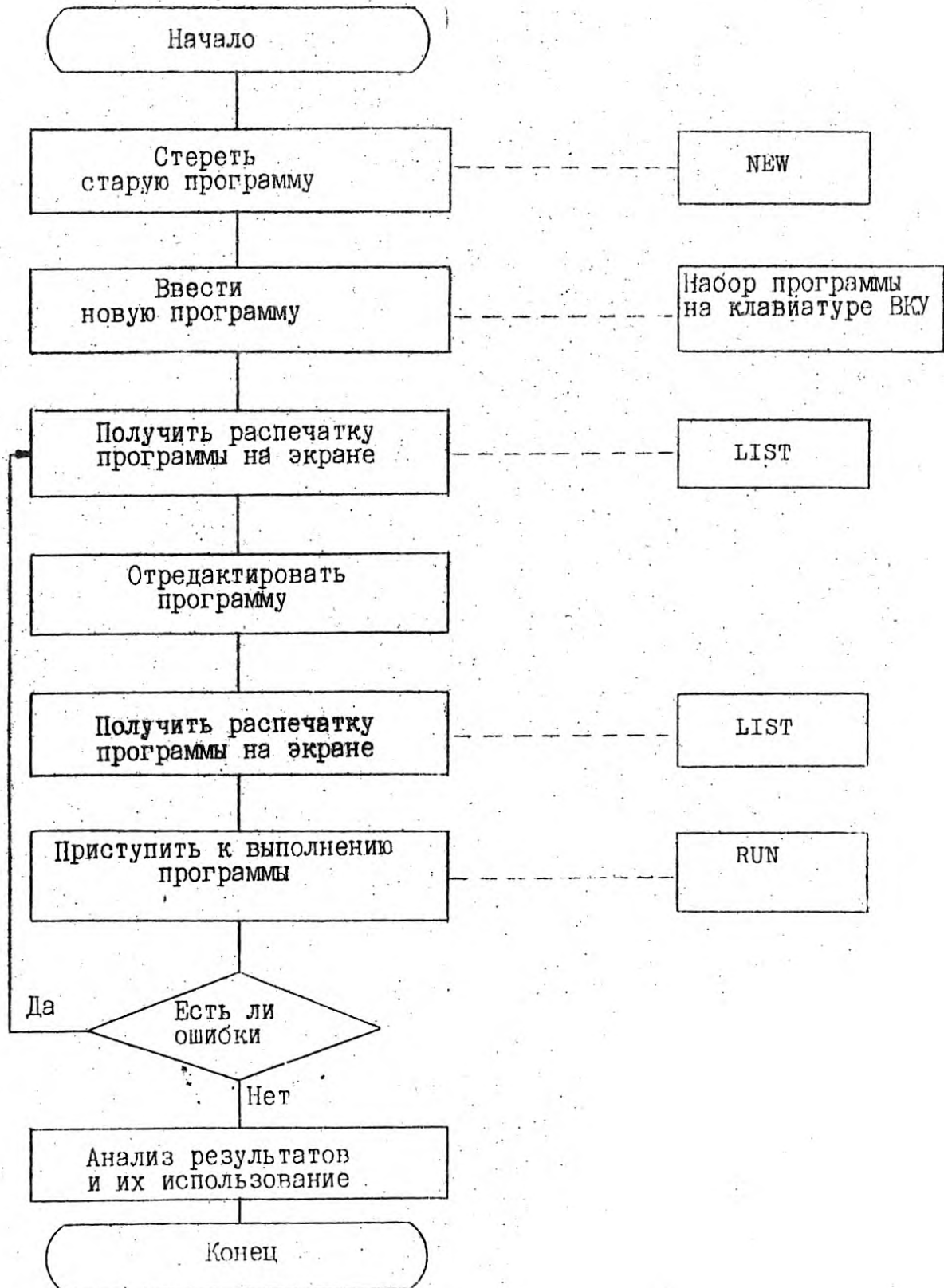
11. СХЕМА АЛГОРИТМА И СТРУКТУРА ПРОГРАММЫ  
НА ЯЗЫКЕ БЕЙСИК

Схема алгоритма

Структура программы



## 12. АЛГОРИТМ РАБОТЫ ПОЛЬЗОВАТЕЛЯ С ПРОГРАММОЙ НА ЭВМ



## Ввод программы с ГМД

1. Включить ПЭВМ.
2. Загрузить с системного ГМД ДОС и интерпретатор языка БЕЙСИК.
3. После высвечивания на экране пригласительного знака " ] " и мигающего курсора необходимо:
  - заменить системный ГМД на тот, с которого необходимо загрузить нужную программу;
  - набрать на клавиатуре следующую директиву:

```
] CATALOG ]
```

После нажатия клавиши  загорается индикатор на передней панели НГМД и на экране появляется перечень всех программ, которые хранятся на данном ГМД.

4. Выбрать нужную из перечисленных программ, набрать на клавиатуре следующую директиву:

```
] RUN имя программы ]
```

Снова загорается индикатор на НГМД, программа загружается в ОП. После загрузки ПЭВМ начинает выполнять программу.

## 13. СООБЩЕНИЯ ОБ ОШИБКАХ

NO FOR ERROR - NEXT без FOR  
SYNTAX ERROR - синтаксическая ошибка  
NO GOSUB ERROR - RETURN без GOSUB  
NO DATA ERROR - мало данных  
ILLEGAL VALUE ERROR - ошибочное значение  
OVERFLOW ERROR - переполнение  
OUT OF MEMORY ERROR - мало памяти  
UNDEF STATEMENT ERROR - нет номера  
SUBSCRIPT ERROR - ошибка индекса  
TYPE ERROR - тип ошибки  
LONG STRING ERROR - строка длинна  
UNDEF NAME ERROR - нет метки  
BYTE UNCOMPL ERROR - байт неполный  
LABEL ERROR - ошибочная метка  
OPCODE ERROR - ошибочный код  
DOUBLE DEF NAME ERROR - уже есть файл с этим именем

FILE NOT FOUND - файл не найден

DISK FULL - ГМД полный

#### 14. СЛОВАРЬ ТЕРМИНОВ

Ссылка означает, что действия, определенные в данной строке станут доступными при выполнении оператора, содержащего эту ссылку.

Вещественные данные связываются с именами, определяющими числовые константы, переменные или функции.

Имя - с его помощью всегда именуют константы, переменные, массивы или элементы массивов. Имена переменной и массива состоят из букв латинского алфавита, за которыми может следовать цифра.

Константа всегда определена в процессе выполнения программы.

Вещественные константы представляются в естественном формате, например:

123.456

.53

Вещественные числа могут быть присвоены вещественным и целым переменным.

Целые константы представляются в десятичном виде (диапазон + 32767) или в шестнадцатеричной форме (от 0 до FFFF) присваиваются целым и вещественным переменным.

Текстовая константа (строковая) - это непустая последовательность символов, заключенных в кавычки, ограничена длиной строки. Символ пробела является значащим, символ кавычки записывается в кавычках, имеет длину не более 256 символов.

Переменная - есть данное, идентифицируемое символическим именем. На него можно сослаться и присвоить ему значение.

Массив - это упорядоченный набор данных, имеющий одно или два измерения. Допускаются только вещественные, поименованные массивы.

Элемент массива - одна из компонент набора данных образующих массив.

Индекс - представляет собой заключенный в круглые скобки список индексных выражений.

Подпрограмма - последовательность предложений, заканчивающаяся специальным оператором возврата из подпрограммы.

Арифметическое выражение формируется из знаков арифметических операций и арифметических операндов (могут быть переменными, функциями или числовыми константами). В выражении могут использоваться круглые скобки.

## ЛИТЕРАТУРА

### Обязательная

1. Машина вычислительная электронная персональная "Агат". Техническое описание. Фг 3.032.002. ТО1.
2. Изучение языка программирования БЕИСИК в средних профтехучилищах. Методические рекомендации. Л., 1987. Госкомитет СССР по профессионально-техническому образованию ВНИИ профтехобразования.
3. Разработка педагогических программных средств вычислительной техники в средних профтехучилищах. Методические рекомендации. Л., 1987. Госкомитет СССР по профессионально-техническому образованию ВНИИ профтехобразования.
4. Отраслевой стандарт. Микропроцессорные средства вычислительной техники. Программное обеспечение. Язык программирования БЕИСИК. ОСТ 11 305.911-82.

### Рекомендуемая

1. Алгоритмические языки: Методические рекомендации для учителей средней школы. - М., 1984.
2. Кетков Ю.Л., Куракина И.М. Программирование на алгоритмических языках БЕИСИК и ФОРТРАН. Учебное пособие. - Горький, 1983.
3. Кетков Ю.Л. Программирование на БЕИСИКЕ. - М., 1978.
4. Куликов В.Д. Курс программирования: Учебное пособие. - Л., 1982.
5. Минимальный БЕИСИК / Сост. И.Н. Брусенцов, Т.Н. Брусенцова. - М., 1981.
6. Программирование на упрощенном БЕИСИКЕ: Методические рекомендации учителю / Сост. И.Б. Антипов. - М., 1984.
7. Уерт Т. Программирование на языке БЕИСИК. - М., 1981.
8. Элементы алгоритмического языка БЕИСИК: Методические материалы. - Омск, 1983.
9. Язык программирования БЕИСИК: простейшие программы. Методические рекомендации учителю. - М., 1984.

## ОГЛАВЛЕНИЕ

Введение .....	3
1. Программное обеспечение ПЭВМ "Агат" .....	3
1.1. Программа "Системный монитор".....	3
1.2. Дисковая операционная система .....	5
1.2.1. Общие команды ДОС .....	6
1.2.2. Работа с файлами типа А .....	7
1.2.3. Работа с файлами типа В .....	8
1.2.4. Работа с файлами типа Т .....	9
1.2.5. Программные команды ДОС .....	11
1.3. Интерпретатор языка БЕЙСИК .....	11
2. Основы программирования на языке БЕЙСИК .....	11
2.1. Представление данных. Типы переменных и констант .....	11
2.2. Представление операций .....	12
2.3. Элементарные математические функции изучаемые в школе .....	13
2.4. Составление математических выражений. Порядок выполнения операций .....	14
2.5. Работа в режиме калькулятора .....	15
3. Представление команд в языке БЕЙСИК .....	15
3.1. Системные команды .....	16
3.2. Команды редактирования текстов. Оформление программ .....	16
3.2.1. Команда LIST - выводение текста программы на экран ВКУ .....	16
3.2.2. Команда DEL - стирание строки программы из ОП .....	17
3.2.3. Команда REM - введение комментариев в программу .....	17
3.2.4. Работа с курсором, редактирование строки .....	17
4. Операторы ввода-вывода информации на экран ВКУ .....	19
5. Описание массивов. Операторы языка работы со строковыми данными .....	23
6. Операторы графики .....	26
6.1. Обозначение графических режимов .....	26
6.2. Операторы графического режима .....	28



7. Операторы передачи управления .....	31
8. Организация циклов .....	34
9. Ввод-вывод данных на ГМД .....	36
9.1. Создание файла .....	37
9.2. Чтение данных из файла .....	37
9.3. Дозапись данных в созданный файл .....	38
9.4. Общие операторы, используемые при вводе-выводе данных на ГМД .....	38
10. Правила записи программы на языке БЕЙСИК .....	39
11. Схема алгоритма и структура программы на языке БЕЙСИК	41
12. Алгоритм работы пользователя с программой на ЭВМ .....	42
13. Сообщения об ошибках .....	43
14. Словарь терминов .....	44
Литература .....	45