Министерство образования Российской Федерации

Южно-Российский государственный технический университет
(Новочеркасский политехнический институт)

А.Ю. Прокопов, И.А. Мартыненко, С.Г. Страданченко,
Н.В. Титов, Е.М. Красунцев, Н.К. Вершинин

ШАХТНОЕ И ПОДЗЕМНОЕ СТРОИТЕЛЬСТВО

Решение практических задач на ЭВМ

Учебное пособие

Допущено Министерством образования Российской Федерации в качестве учебного пособия для студентов высших учебных заведений, обучающихся по специальности 090400 "Шахтное и подземное строительство"

Новочеркасск 2000

ББК 32.973-018
075
УДК 681.3.06

Рецензенты: доктор техн. наук, проф. Н.С. Булычев
доктор техн. наук, проф. М.Н. Шуплик

Авторы: Прокопов А.Ю. (гл. 1, 2, 3, 5 – 7);
Мартыненко И.А. (гл. 2);
Страданченко С.Г., Титов Н.В. (гл. 3, 4);
Красунцев Е.М., Вершинин Н.К. (гл. 8).

П78 Шахтное и подземное строительство. Решение практических задач на ЭВМ: Учеб. пособие/ Юж.-Рос. гос. техн. ун-т. Новочеркасск: ЮРГТУ, 2000. 172 с.
ISBN 5-88998-099-8

Содержит теоретические основы программирования на языке QBasic применительно к инженерным задачам шахтного и подземного строительства и описание прикладных программ для решения таких задач. Включает варианты лабораторных работ по курсу "Программирование и расчеты на ЭВМ".

Предназначено для студентов специальности 0904 "Шахтное и подземное строительство".

П $\frac{2404010000-156}{98П(03)-99}$ Без объявл. УДК 681.3.06

ISBN 5-88998-099-8

ПРЕДИСЛОВИЕ

В процессе обучения студенты технических вузов постоянно сталкиваются с необходимостью использовать персональный компьютер для выполнения различных учебных заданий. Для эффективной работы на персональных компьютерах студенты должны уметь не только использовать готовые прикладные программы, но и владеть основами алгоритмизации и навыками составления собственных программ. С этой целью введена учебная дисциплина "Программирование и расчеты на ЭВМ", которая является завершающей в общем плане освоения основ программирования и работы на персональном компьютере. Данная дисциплина имеет прикладной характер и направлена на подготовку инженеров, способных решать различные задачи по своей специальности с использованием ЭВМ.

Настоящее учебное пособие содержит теоретические основы программирования на языке QBasic и описание прикладных программ, разработанных сотрудниками кафедры “Строительство подземных сооружений и шахт” Шахтинского института ЮРГТУ. Для улучшения восприятия теоретический материал сопровождается подробными пояснениями и конкретными примерами. Кроме того, в пособие включены темы и варианты лабораторных работ, призванных закрепить на практике теоретические знания, полученные студентами в области программирования.

При написании данного пособия не ставилась цель охватить сразу все возможности языка QBasic, поскольку, как показывает практика, излишняя загроможденность языка не позволяет студенту эффективно использовать основные средства языка для решения прикладных задач. В то же время объем настоящего пособия, на наш взгляд, достаточен не только для выполнения предлагаемых в нем лабораторных работ, но и для написания программ, которые могут встретиться инженеру-практику, использующему ЭВМ на его основной работе.

Для облегчения изучения основ программирования на QBasic пособие разбито на несколько разделов, каждый из которых образует единый блок теоретического материала. Последовательное изучение предлагаемого материала является подготовительной ступенью к выполнению лабораторных работ, которые приведены в конце каждого раздела.

Предлагаемый в настоящем пособии курс предусматривает проведение лабораторных работ, образующих два основных блока. Первый блок (работы №1 – 5) включает задания, требующие от студента знаний и навыков в области программирования на языке QBasic. Второй блок (работы №6 – 9) содержит задания, выполнение которых требует применения прикладных

программ, разработанных сотрудниками кафедры строительства подземных сооружений и шахт и используемых студентами старших курсов при курсовом и дипломном проектировании.

Разработанный курс лабораторных работ охватывает основные принципы программирования алгоритмов различной структуры, при этом задания по программированию основаны на решении задач по специальности и тесно связаны с такими курсами, как “Геомеханика”, “Технология и безопасность буровзрывных работ”, “Горные и строительные машины”, “Технология шахтного и подземного строительства” и др.

С целью облегчения понимания и запоминания операторов языка в пособие включено прил. 1, в котором в алфавитном порядке перечислены все служебные слова языка QBasic с их расшифровкой и примерным переводом на русский язык.

Для удобства отладки программ при выполнении лабораторных работ в пособие включено прил. 2, в котором перечислены основные ошибки, обнаруживаемые компилятором, и описаны возможные причины ошибок.

Для удобства работы с пособием в него включен алфавитный указатель, в котором приведены все операторы и служебные слова языка, рассматриваемые в настоящем пособии.

Основными задачами изучения дисциплины авторы считают:

– изучение и закрепление основ алгоритмизации и программирования;
– привитие навыков использования основных приемов программирования в решении задач шахтного строительства;
– ознакомление с прикладными программами, использующимися для проектирования строительства горных предприятий.

В результате изучения дисциплины студенты должны:

– знать основные средства языка программирования QBasic;
– уметь составлять алгоритмы расчетов, строить блок-схемы алгоритмов и программировать на языке QBasic;
– уметь использовать прикладные программы шахтного строительства.

1. ОСНОВЫ АЛГОРИТМИЗАЦИИ

Решение задачи на ЭВМ включает следующие основные этапы:

1. Постановка задачи и разработка математической модели.
2. Выбор метода численного решения.
3. Разработка алгоритма и структуры данных.
4. Реализация алгоритма на языке программирования (разработка программы).
5. Ввод и отладка программы на ЭВМ.
6. Решение задачи на ЭВМ, обработка и оформление результатов расчета.

При этом первые два этапа осуществляются без участия ЭВМ и включают выбор и обоснование физических законов или вывод зависимостей для корректного решения поставленной задачи, а также выбор математического метода решения задачи. Последующие этапы непосредственно связаны с переложением разработанной математической модели на язык, понятный для ЭВМ. Для этого необходимо, в первую очередь, определить порядок действий, т.е. разработать алгоритм.

Алгоритм – это точно определенная последовательность действий или операций, необходимых для решения конкретной задачи.

При составлении алгоритма все действия, выполняемые ЭВМ, можно разделить на следующие блоки:

1. *Блок ввода исходной информации*, необходимой для выполнения программы.

2. Основной блок действий, включающий определение всех необходимых параметров, предусмотренных постановкой задачи, или *блок преобразования исходной информации*.

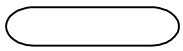
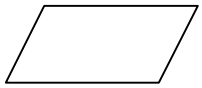
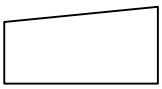
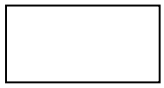
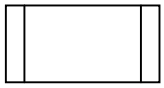
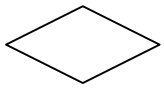
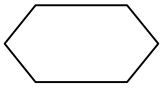
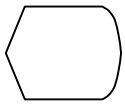
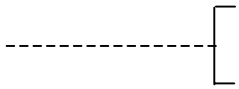
3. *Блок вывода результатов* выполнения программы.

Любой алгоритм может быть представлен в графической форме. Данный способ наглядно изображает этапы вычислений при решении конкретной задачи. Сущность графического способа заключается в том, что каждой операции ставится в соответствие определенный графический символ. Форма символов установлена ГОСТ 19.002-80, а правила составления схем алгоритмов – ГОСТ 19.002-80.

Представление алгоритмов, выполненное графическим способом, называется *блок-схемой*. Наиболее часто употребляемые символы блок-схем представлены в табл.1.

Таблица 1

Символы блок-схем

Название символа	Обозначение	Пояснение
Пуск-останов		Используется для указания начала, конца алгоритма, а также временных остановок
Ввод (данные)		Используется для обозначения автоматического ввода информации
Ручной ввод		Обозначение ручного ввода информации
Процесс		Вычислительное действие или последовательность вычислительных действий
Предопределенный процесс		Вычисления по подпрограмме пользователя или стандартной подпрограмме
Решение		Проверка условий в случае ветвления алгоритма по условию
Модификация		Начало цикла в случае циклического алгоритма с циклическим параметром
Документ		Используется для обозначения вывода информации на печатающее устройство
Дисплей		Используется для обозначения вывода информации на дисплей
Комментарий		Позволяет включать в схемы алгоритмов пояснения к функциональным блокам
Линия потока		Используется для обозначения порядка выполнения действий
Соединитель (внутристраничный)-узел		Для указания связи между последовательно выполняемыми функциональными блоками
Соединитель (межстраничный)		Для указания связи между функциональными блоками при разрыве блок-схемы

Сделаем более подробное пояснение некоторых символов из табл. 1.

Процесс – применяется для обозначения одного или последовательности действий, изменяющих значение, форму представления или размещения данных. Например, для обозначения вычислений можно использовать математические выражения, для пересылки данных – стрелки, для других действий – пояснения на естественном языке.

Решение – используется для обозначения переходов управления в алгоритме по условию. Внутри блока должно быть обязательно указано условие, в зависимости от выполнения которого решение задачи пойдет по направлению, указанному стрелкой ДА, в случае невыполнения – по направлению НЕТ.

Модификация – используется для организации циклических конструкций. Внутри блока обязательно должен быть записан параметр цикла, для которого указываются его начальное и конечное значение, а также правило изменения параметра цикла для каждого повторения.

Линия потока – используется для обозначения порядка выполнения действий. Для улучшения наглядности следует придерживаться стандартных правил изображения линий передачи управления – сверху вниз и слева направо. Если необходимо показать передачу управления снизу вверх или справа налево, то направление следует отметить стрелкой.

Смысл других символов достаточно ясен и отдельного пояснения не требует. Различные способы построения блок-схем будут рассмотрены ниже при решении конкретных примеров.

2. ОСНОВНЫЕ СВЕДЕНИЯ О ЯЗЫКЕ QBASIC

2.1. Алфавит языка QBasic

Программа на языке QBasic есть запись последовательности сгруппированных в строки команд (инструкций), под действием которых ПЭВМ выполняет необходимые для решения заданной задачи операции. В начале любой строки программы может стоять номер строки или метка. В качестве метки может использоваться либо число, либо сочетание букв и цифр. В последнем случае после метки необходимо поставить двоеточие.

Примеры строк программы на QBasic:

```
10 A=12
20 S=A+B
met1: I=1
```

Здесь в качестве меток выступают 10, 20 и met1. Однако нумерация (применение меток) строки в QBasic оправдана лишь в том случае, когда в тексте программы на данную строку есть ссылка (передача управления), для остальных же строк метки необязательны и нецелесообразны.

Программа на QBasic состоит из команд, которые представляют собой совокупность различных символов – цифр, букв и специальных знаков, перечисленных ниже. В одной строке программы может содержаться одна или несколько команд (в последнем случае команды отделяются друг от друга двоеточием, например: A = 1: B = 12: K = 5).

Алфавит QBasic включает:

– 26 латинских прописных *букв*:
A, B, C, D, E, F, G, H, I, J, K, L, M, N, O, P, Q, R, S, T, U, V, W, X, Y, Z ;

– 10 арабских *цифр*:
Ø, 1, 2, 3, 4, 5, 6, 7, 8, 9
(нуль в программах на QBasic обычно перечеркивается наклонной чертой, чтобы нельзя было спутать его с буквой O);

– *знаки арифметических операций:*

–	вычитание или присвоение знака минус,
+	сложение,
*	умножение,
/	деление,
^	возведение в степень.

В QBasic широко используются следующие *специальные знаки*:

( )	–	скобки	!	–	восклицательный знак
:	–	двоеточие	=	–	равенство
;	–	точка с запятой	>	–	меньше
.	–	точка	<	–	больше
,	–	запятая	\	–	наклонная черта
" "	–	кавычки	#	–	«решетка»
‘	–	апостроф	$	–	знак денежной единицы
?	–	вопросительный знак	%	–	знак процентов
&	–	амперсант			

Назначение указанных символов будет рассмотрено нами в дальнейшем в сочетании с операторами языка.

2.2. Форма записи чисел

Основной вид информации, которой оперирует ПЭВМ, – *числа*.

Наиболее общей формой представления чисел на Бейсике является десятичная форма. Ей соответствуют десятичные числа, записываемые в виде

$$\pm M \cdot 10^{\pm E}$$

где M – мантисса, E – порядок.

Форма записи чисел на QBasic близка к естественной. Они представляются в виде последовательности соответствующих цифр с особыми знаками: знаками мантиссы и порядка + или – , символом ввода порядка E (от слова *Extent* – степень) и знаком разделения целой и дробной части чисел – точкой вместо разделительной запятой.

Общий вид числа на QBasic можно представить следующей схемой:

±	…	.	…	E	±	…
знак мантиссы	целая часть числа	десятичная точка	дробная часть числа	символ ввода порядка	знак порядка	величина порядка

В таком виде более целесообразно представлять очень большие или очень малые числа, при этом для положительных значений мантиссы и порядка знак "+" может быть опущен, например:

2.141E05 1.23E4 2.345E-09 -1.22E-05 8.8E-2

(214100) (12300) ($2{,}345 \cdot 10^{-9}$) ($-1.22 \cdot 10^{-5}$) (0,088)

Числа могут быть целыми, т.е. не содержать дробной части:

0 18 -123 7630

Дробные числа с небольшим абсолютным значением величины порядка (до 3) могут быть представлены в виде

3.1416 7.05 -12.55 2.7182 0.023 -0.189

Однако, если первая цифра мантиссы – число 0, ее можно не вводить. Так, правильные дроби записываются в виде

.012 .3 -.991 -.0011

2.3. Переменные в QBasic

2.3.1. Переменная. Имя переменной.

Переменные – величины, значение которых может изменяться в ходе выполнения программы. Для обозначения переменных в программе на QBasic используют *имена* переменных (*идентификаторы*), которые могут содержать до 40 знаков (без пробелов) – букв и цифр, например:

C, A3, ARMIR, PROV12, AMM6SV, DV24M и т.д.

При выборе имени переменной **необходимо помнить следующее**:

– имя переменной должно начинаться с буквы, в противном случае цифра, стоящая в начале имени, будет восприниматься машиной как метка (номер строки);

– имена переменных, состоящие из одинаковых прописных и строчных букв, воспринимаются как одно и то же имя, например, s и S, stv и STV и т.п. (здесь стоит заметить, что в большинстве версий QBasic при вводе имен переменных предусмотрена автоматическая замена строчных букв на прописные или наоборот в случае задания пользователем идентичных имен, отличающихся лишь регистром букв);

– недопустимо включать в имя переменной буквы кириллицы, а также специальные символы (за исключением %, &, !, #, $ в конце имени, называемые суффиксами и используемые для описания типа данных);

– недопустимо использовать в качестве имен переменных зарезервированные слова (прил. 1), использующиеся в QBasic с целью реализации некоторых заранее определенных синтаксических потребностей, но идентификаторы могут содержать их внутри себя. Например, END является недопустимым именем для переменной, поскольку имеется зарезервированное слово END. Однако, имена ENDHERE и FRIEND допустимы. Кроме того, идентификатор не должен начинаться с сочетания букв FN, такое сочетание обозначает функцию, определенную пользователем (подробнее об этом см. п.6.1). Попытка использования зарезервированных слов в качестве идентификаторов вызовет фиксацию ошибки во время компиляции;

– для удобства отладки программы имена переменных целесообразно назначать в соответствии с обозначениями переменных, входящих в расчетную формулу, при этом переменные, обозначаемые греческими буквами, именовать полным названием буквы: ALFA, BETA, PI, FI и т.п.

2.3.2. Типы переменных и их описание

Переменные в QBasic могут быть *числовыми* (переменной присваивается числовое значение) или *символьными* (переменной присваивается любой символ или последовательность символов, включая пробелы).

Рассмотрим более подробно каждый из типов переменных.

На QBasic существует несколько типов числовых переменных.

Для операций с небольшими по модулю целыми числами используются *целочисленные переменные одинарной (обычной) точности* (INTEGER). Арифметические операции с такими переменными ПЭВМ выполняет точно. Переменные объявляются целочисленными обычной точности путем ввода после них знака **%**, например:

A5%, DSV%, STVOL%,

а их значение может изменяться в пределах -32768 до +32767.

Кроме знака % переменные могут быть описаны как целочисленные одинарной точности с помощью оператора **DEFINT** (DEFINED INTEGER), например:

```
DEFINT A-D
```

После такой строки в программе все переменные, имя которых начинается с A, B, C и D, объявляются целочисленными, и знак % после имени переменной становится необязательным.

Для описания некоторой конкретной переменной как целочисленной необходимо использовать оператор описания переменной

DIM <имя переменной> **AS** <тип переменной>

Например, при необходимости описания переменной T5 как целочисленной одинарной точности необходимо ввести строку

```
DIM T5 AS INTEGER
```

Целочисленные переменные двойной точности (длинные целые, LONG) обозначают суффиксом **&**, а их значения лежат в диапазоне от –2147483648 до +2147483647.

Примеры таких переменных:

LVV&, PAJ&, NGTU&

Для описания целочисленных переменных двойной точности может быть использован оператор **DEFLNG** (DEFINED LONG) или оператора описания переменной.

Например, строка

```
DEFLNG D-F,K
```

объявляет целочисленными двойной точности все переменные, начинающиеся с букв D, E, F или K, а команда

```
DIM K9B AS LONG
```

одну конкретную переменную К9В.

Рекомендуется широко использовать целые числа в программах на QB, поскольку они позволяют получать точные результаты, экономят память компьютера и обеспечивают высокую скорость вычислений.

Однако при решении инженерных задач чаще всего приходится иметь дело с действительными (вещественными) числами и переменными. Поэтому по умолчанию (при отсутствии специального указания типа переменных) все переменные и константы задаются как *вещественные одинарной точности*. (SINGLE). На то же может указывать знак **!**, стоящий в конце имени переменной, например, WWW!, RAS! и т.п. Таким образом, числовая переменная, у которой указан суффикс "!", и та, у которой этот суффикс отсутствует, считаются одинаковыми и не различаются языком QBasic.

Диапазон изменения вещественных чисел одинарной точности для положительных чисел от 1,4E-45 до 3,4E+38, а для отрицательных чисел от –3,4E+38 до –1,4E-45 (значения мантисс указано приближенно).

Для описания этого типа переменных используют оператор **DEFSNG** (DEFINED SINGLE) или оператор описания переменных. Смысл и синтаксис команд аналогичен описанию целочисленных переменных, например:

```
DEFSNG A
DIM B AS SINGLE
```

Вещественные числа двойной точности (DOUBLE) обычно применяются для точных математических вычислений, не допускающих потерю значности. Их обозначают знаком #. При выводе этих чисел, для их отличия от вещественных чисел одинарной точности, буква E, отделяющая мантиссу от порядка, заменяется буквой D. Диапазон изменения вещественных чисел двойной точности для положительных чисел от 4.9D-324 до 1.8D+308, для отрицательных – от –1.8D+308 до –4.9D-324 (значения мантисс указаны приближенно).

Для описания этого типа переменных используют оператор **DEFDBL** (DEFINED DOUBLE) или оператор описания переменных. Например:

```
DEFDBL A-Z
```

Данная команда объявляет все переменные в программе как вещественные двойной точности.

```
DIM A,B,C AS DOUBLE
```

При выполнении данной строки переменные A,B и C будут объявлены как вещественные двойной точности.

В отличие от числовых переменных, *символьным* (*строковым*, STRING) переменным могут присваиваться любые символы или последовательность символов. Символьная переменная обозначается знаком **$**. Напри-

мер,

PROFIL$, RASSTREL$, TIP$, VV$ и т.д.

Строки широко используются для представления данных и организации диалога с компьютером. Символьным переменным можно присваивать значения, содержащие русские буквы. Для присваивания значения символьной переменной, значение берется в кавычки, например:

```
PROFIL$ = "двутавр"
KR$ = "СВП-27"
VV$ = "аммонит 6ЖВ" и т.д.
```

Так же, как и числовые, символьные переменные могут быть объявлены в начале программы. Для этого используют оператор **DEFSTR** (DEFINE STRING) или оператор описания переменных, например строка

```
DEFSTR V
```

объявляет символьными все переменные, начинающиеся с буквы V, а строка

```
DIM VV AS STRING
```

объявляет символьной переменную VV.

Ко всему сказанному о присваивании типов данных следует добавить, что явное указание типа знаками %, &, !, #, $ имеет приоритет (преимущество) перед объявлением типа с помощью DEF.

2.3.3. Преобразование типов данных

Определенный программой тип данных может изменяться в ходе ее выполнения. Для преобразования типов данных используются следующие функции:

CINT – функция, преобразующая числовое выражение в целое путем округления дробной части выражения. Если дробная часть выражения превышает 0,5, округление производится в большую сторону; в противном случае – в меньшую.

Например, в результате выполнения следующего фрагмента программы:

```
B = 34.50009
C = 12.23
E = CINT(B)
F = CINT(C)
```

переменная E примет значение 35, а F – значение 12.

CLNG – функция преобразования, приводящая числовое выражение к длинному целому путем округления целой части. Действие данной функции аналогично функции CINT, но распространяется на значения, не вмещающиеся в формат целочисленных переменных одинарной точности.

CSNG – функция, преобразующая числовое выражение в значение

обычной точности.

Например, в результате выполнения программы:

```
a# = -34.5657899#
b# = 12564.235599#
c = CSNG(a#)
d = CSNG(b#)
PRINT a#, b#
PRINT c, d
```

на печать выйдут следующие значения:

```
-34.5657899      12564.235599
-34.56579        12564.24
```

CDBL – функция, преобразующая числовое выражение в число двойной точности.

Например:

```
x = 7 / 9
PRINT x
PRINT CDBL(x)
```

Результат выполнения программы:

```
.7777778
.7777777910232544
```

2.4. Арифметические операции

Для выполнения элементарных арифметических действий в QBasic используют знаки, приведенные в табл.2.

Таблица 2

Знаки арифметических операций в QBasic

Действие	Условное обозначение	Пример
Сложение	+	C = A + B
Вычитание	–	C = A – B
Умножение	*	C = A * B
Деление	/	C = A / B
Целочисленное деление	\	C = A% \ B%
Остаток после целочисленного деления	MOD	C = A% MOD B%
Возведение в степень	^	B = A^3

Обычные арифметические действия (+, –, *, /) над вещественными числами выполняются с определенной степенью точности.

Действия с целочисленными переменными выполняются точно.

Поясним на примерах некоторые операции.

В результате целочисленного деления (знак \) отбрасывается дробная

часть числа, например, после выполнения части программы

```
A% = 100
B% = 33
C = A%\B%
D = A% MOD B%
```

переменной C будет присвоено значение 3.

При нахождении остатка от деления (MOD) в данном случае получаем значение переменной D, равное 1 (поскольку $\frac{100}{33} = 3\frac{1}{3}$). Если же целочисленное деление применить для вещественных переменных, то их значения вначале округляются до целых, а затем происходит деление как для целочисленных переменных.

2.5. Стандартные функции в QBasic

Вычисление стандартных математических функций уже запрограммировано в самом языке QBasic. Все эти функции вычисляются специальными численными методами, реализованными микропрограммно. Поэтому для их использования в программе достаточно написать имя соответствующей функции на языке QBasic. Наиболее часто используемые стандартные функции приведены в табл. 3.

Таблица 3

Стандартные функции языка QBasic

Наименование функции	Математическое обозначение	Запись на QBasic	Примечание
Квадратный корень	$\sqrt{x}$	SQR(X)	x ≥ 0
Показательная	e^x	EXP(X)	x ≤ 88,7
Логарифм натуральный	*ln x*	LOG(X)	x > 0
Синус	*sin x*	SIN(X)	x – в радианах
Косинус	*cos x*	COS(X)	то же
Тангенс	*tg x*	TAN(X)	то же
Арктангенс	*arctg x*	ATN(X)	Вычисляемое значение угла от -π/2 до π/2 рад.
Абсолютное значение	$\|x\|$	ABS(X)	
Присвоение знака	*знак x*	SGN(X)	+1 при x > 0 0 при x = 0 -1 при x < 0
Случайная	–	RND	Выдает случайное число в интервале от 0 до 1

Как видно из приведенных примеров, значение аргумента обязательно заключается в круглые скобки.

Следует обратить также особое внимание на то, что аргумент, стоя-

щий под знаком тригонометрических функций, должен быть выражен в **радианах**, в то время как большинство угловых величин в расчетах инженера-шахтостроителя (угол наклона выработки, угол падения пласта, угол внутреннего трения пород и т.д.) удобнее задавать в градусах.

Для перевода значения угла из радиан в градусы в программе следует использовать соотношение

$$\alpha^{рад} = \frac{\alpha^{\circ} \cdot \pi}{180^{\circ}}$$

Значение функции ATN(X) также выдается в радианах, для перевода его в градусы необходимо использовать соотношение

$$\alpha^{\circ} = \frac{\alpha^{рад} \cdot 180^{\circ}}{\pi}$$

2.6. Организация обратных тригонометрических и гиперболических функций

Так как из обратных тригонометрических функций в число стандартных входит только ATN(X) [arctg(x)], то другие функции можно получить из соотношений:

$$\arcsin(x) = \operatorname{arctg}\frac{x}{\sqrt{1-x^2}};$$

$$\arccos(x) = \frac{\pi}{2} - \operatorname{arctg}\frac{x}{\sqrt{1-x^2}};$$

$$\operatorname{arcctg}(x) = \frac{\pi}{2} - \operatorname{arctg}(x).$$

Гиперболические и обратные гиперболические функции легко выражаются через экспоненциальные и логарифмические функции. Так, гиперболический синус *sh x*, косинус *ch x* и тангенс *th x* связаны с функцией e^x соотношениями:

$$\operatorname{sh}(x) = \frac{e^x - e^{-x}}{2};$$

$$\operatorname{ch}(x) = \frac{e^x + e^{-x}}{2};$$

$$\operatorname{th}(x) = \frac{\operatorname{sh}(x)}{\operatorname{ch}(x)} = \frac{e^x - e^{-x}}{e^x + e^{-x}}.$$

Для обратных гиперболических функций справедливы соотношения:

$$\operatorname{arsh}(x) = \ln(x + \sqrt{1+x^2});$$

$$\operatorname{arch}(x) = \ln(x + \sqrt{x^2 - 1});$$

$$\operatorname{arth}(x) = \ln\sqrt{\frac{x+1}{1-x}}.$$

2.7. Расчет логарифмических функций

В стандартные функции QBasic входит только натуральный логарифм (LOG(X)). Для расчета логарифма по любому основанию необходимо использовать формулу перехода к новому основанию:

$$\log_a b = \frac{\log_c b}{\log_c a}.$$

В частности, для перехода от десятичного логарифма к натуральному:

$$\lg b = \frac{\ln b}{\ln 10} = 0{,}4342945 \cdot \ln b.$$

Заметим, что лучше использовать не более короткое выражение lnb/ln10, а последнее – с заранее вычисленным значением 1/ln10. Это почти вдвое сокращает время вычисления функции LOG(X). Аналогично вычисляются логарифмы *b* при любом другом основании:

$$\log_a b = \frac{\ln x}{\ln a}.$$

2.8. Функции округления

Часто при решении прикладных задач возникает необходимость округления результата до ближайшего большего или меньшего целого.

С этой целью в QBasic используют следующие функции:

INT – математическая функция, определяющая ближайшее меньшее целое значение числа;

FIX – математическая функция, определяющая целую часть числа.

FIX(*x*) эквивалентно выражению SGN(*x*)*INT(ABS(*x*)). Различие между FIX и INT состоит в том, что при *x*<0 FIX(*x*) выдает ближайшее целое отрицательное число, большее *x*, тогда как INT(*x*) выдает ближайшее целое отрицательное число, меньшее *x*. При положительных же значениях x действие функций FIX(x) и INT(x) одинаково.

Данные функции используются для нахождения целой части числа, что не всегда соответствует округлению в математическом смысле. Для корректного математического округления целесообразно использовать рассмотренную нами ранее (п. 2.3.3) функцию преобразования типов данных CINT(*x*).

Различия в результатах выполнения трех названных функций наглядно демонстрируют следующие значения функций:

x	INT(x)	CINT(x)	FIX(x)
99.3	99	99	99
99.7	99	100	99
-99.3	-100	-99	-99
-99.7	-100	-100	-99

2.9. Выражения в QBasic

Арифметические выражения состоят из констант, переменных и функций, соединенных знаками арифметических операций (+, –, *, /, ^).

Выражения пишутся в одну строку. Для указания порядка выполнения операций используются круглые скобки. Внутри скобок операции выполняются слева направо с общепринятыми приоритетами:

– вычисление значений функций;
– возведение в степень;
– умножение и деление;
– сложение и вычитание.

Иногда возникают сомнения в выполнении одинаковых по приоритету операций, например A*B/C/D. В таких случаях полезно помнить, что равноценные по приоритету операции выполняются слева направо. Таким образом, вначале вычисляется A*B, затем полученное число делится на C и результат (A*B/C) еще раз делится на D.

В сомнительных случаях необходимо использовать одно из Золотых правил программиста: "**Лишние скобки обходятся дешевле, чем ошибки**". Например, выражение ab/cd, вполне понятное при обычной математической форме записи, нельзя записать в виде A*B/C*D, так как это даст результат abd/c, а не ab/cd. Введя скобки, получим правильную запись A*B/(C*D). Правомерна также и такая запись (A*B)/(C*D) или A*B/C/D. Число вводимых скобок неограниченно.

Конкретные примеры арифметических выражений в QBasic приведены в табл.4.

Таблица 4

Примеры арифметических выражений в QBasic

Формула	Выражение в QBasic
$T_{бур} = \frac{N \cdot l}{n_1 \cdot k_1 \cdot v}$	Tbur=N*L/(n1*k1*nu)
$P = \frac{\gamma \cdot l^3}{6 \cdot h \cdot \mathrm{tg}^2(\frac{\pi}{4} - \frac{\rho}{2})}$	P=gamma*L^3/(6*H*(TAN(pi/4–ro/2))^2)
$d_0 = \frac{4{,}4 \cdot l_0}{\sigma \cdot \sqrt{\mathrm{tg}\,\rho}} \cdot \sqrt[3]{\frac{l_0}{h_0}}$	D0=4.4*L0/(sigma*SQR(TAN(ro)))*(L0/H0)^(1/3)

Два знака операции подряд недопустимы. Например, не допускается запись 2A+B вместо 2*A+B или A*–BC вместо A*(-B)*C.

При необходимости расчета по очень сложным выражениям целесообразнее разбивать формулу на несколько характерных частей, например, числитель и знаменатель, а затем дополнительной операцией рассчитывать необходимое значение.

2.10. Комментарии

Для удобства отладки программы в нее целесообразно включать комментарии. Для этого в QBasic используется оператор **REM**, вслед за которым можно набрать любой текст. При этом комментарий может содержать буквы как латинского, так и русского алфавита. Оператор REM должен быть последним или единственным в строке, т.к. все стоящие после него символы не воспринимаются как команды.

Оператор REM может быть заменен апострофом. При этом, если комментарий, начинаемый апострофом, стоит в конце строки после исполняемых операторов, то перед апострофом двоеточие не ставится, а перед оператором REM ставится, например:

```
REM Расчет ленточного фундамента
SIG = 3E07 ' Предел прочности бетона на сжатие
G = 4500 : REM Грузоподъемность автобетоновоза, т
```

3. ПРОГРАММИРОВАНИЕ АЛГОРИТМОВ ЛИНЕЙНОЙ СТРУКТУРЫ

Алгоритмом линейной структуры будем считать алгоритм с однократным последовательным безусловным выполнением всех действий. Такой алгоритм состоит из блока ввода исходных данных, одной последовательности вычислительных действий и вывода результатов.

3.1. Организация ввода исходных данных

Перед началом выполнения вычислительных действий необходимо задать переменным начальные значения, т.е. ввести исходные данные для расчета. Рассмотрим способы ввода исходных данных.

3.1.1. Оператор присвоения

Самым простым способом задания переменной некоторого значения является оператор присвоения **LET**. Например,

```
LET A5=12
```

Здесь переменной A5 присваивается значение 12. В QBasic ключевое слово LET можно не писать (что обычно и делается в программах), тогда данная строка будет иметь вид

```
A5=12
```

Примеры других присвоений даны ниже:

```
K = 0.025
DV% = 12500
X# = 12.3443566511
NAME$ = "ШИ ЮРГТУ"
I = (H-H1)/H
```

В последнем примере к моменту присвоения значения переменной I должны быть присвоены значения переменным H и H1, в противном случае данным переменным по умолчанию будет присвоено значение 0, что в итоге приведет к ошибке "Division by zero" (деление на нуль).

В случае, если начальное значение некоторой переменной A равно 0, оператор присвоения

$$A = 0$$

является необязательным, поскольку при расчете выражения, в котором не определены некоторые переменные, им автоматически присваивается значение 0.

3.1.2. Блок данных. Автоматический ввод исходных данных

Второй прием, с помощью которого переменным могут быть присвоены значения, основан на использовании оператора блока данных **DATA** и оператора считывания данных из блока данных **READ**.

Эти два оператора всегда используются совместно, например:

```
READ f, mu, L, S, P
DATA 8, 1.05, 2.2, 16.3, 45
```

Здесь оператор READ содержит список имен переменных, а оператор DATA список числовых значений, которые должны быть присвоены соответствующим переменным в списке READ. Так, в результате выполнения данных строк переменные получат следующие значения:

f = 8, mu = 1.05, L = 2.2, S = 16.3, P = 45

Данные операторы могут быть использованы для любых типов переменных, в том числе и символьных, например:

```
READ VV$, TIP$, VRUB$
DATA аммонит Т-19, контурное, клиновой
```

Выполнение данных операторов равносильно использованию следующих операторов присвоения:

```
VV$ = "аммонит Т-19"
TIP$ = "контурное"
VRUB$ = "клиновой"
```

Число операторов DATA и READ в программе может быть различным. Все операторы READ образуют единый блок данных, в котором находится внутренний указатель. При присвоении переменной значения из блока данных, внутренний указатель автоматически переходит на следующий элемент блока, так что программа в любой момент знает, какие элементы блока данных уже присвоены, а какие еще нет. Например, в программе

```
DATA 1
⋮
DATA 8, 5
⋮
DATA 2,7,22
READ B2,B4
⋮
READ A,C1,C5
```

переменной B2 будет присвоено значение 1, B4=8, A=5, C1=2, C5=7, причем, внутренний указатель будет стоять на числе 22 в последнем операторе DATA. Следовательно, если еще где-либо в программе встретится оператор READ, то присвоение переменным значений из блока данных начнется со значения 22.

Для повторного использования данных, стоящих в списке DATA, служит оператор **RESTORE**, который возвращает внутренний указатель на заданную метку и позволяет оператору READ повторно считывать данные, например:

```
   DATA 1,2,3
   READ A,B,C
10  DATA 7,8,9
   READ D,E,F
   RESTORE 10
   READ G,H
```

В результате выполнения данных строк переменным будут присвоены следующие значения: A=1, B=2, C=3, D=7, E=8, F=9, затем оператор RESTORE 10 возвращает внутренний указатель на метку 10 и переменная G получает значение 7, а H=8.

В том случае, если метка после RESTORE отсутствует, то следующий оператор READ будет считывать данные из первого в программе оператора DATA, например:

```
DATA 1,2,3
READ A,B,C
DATA 7,8,9
READ D,E,F
RESTORE
READ G,H
```

В этом случае G=1, H=2.

Если же в блоке данных некоторый элемент (число) необходимо пропустить, то в операторе READ соответствующая переменная повторяется нужное число раз, например:

```
DATA 4,7,9
READ A,A,A
```

В результате переменной A сначала будет присвоено значение 4, затем 7, а затем 9. Следовательно, окончательное значение у переменной A будет 9 и тем самым, первых два значения из блока данных будут пропущены.

При использовании операторов DATA – READ **необходимо также помнить следующее**:

– имена переменных в списке READ и числовые или символьные значения в списке DATA всегда разделяются запятыми;

– оператор READ должен находиться в том месте программы, где происходит присвоение данных переменным, а оператор DATA – в любом месте программы;

– для наглядности программы и простоты ее отладки операторы DATA и READ целесообразно помещать в соседних строках и следить за тем, чтобы число имен переменных в операторе READ соответствовало количеству числовых или символьных значений в операторе DATA. В том случае, когда список переменных больше, чем список данных, выдается сообщение об ошибке "Out of DATA" (конец данных в операторе DATA). Если же список переменных короче, чем список данных, то лишние (не прочитанные никаким READ) данные игнорируются;

– типы данных и переменных должны соответствовать: чтение числового значения в символьную переменную не приведет к ошибке, напротив, попытка считать символьную строку в числовую переменную вызовет ошибку "Type mismatch" (ошибка в типе данных);

– чтение числового значения, слишком большого для переменной заданного типа, приведет к ошибке "Overflow" (переполнение).

3.1.3. Ручной ввод исходных данных

Вышерассмотренные способы ввода исходной информации рассчитаны на вычисление некоторых параметров с конкретными исходными данными и не позволяют использовать программу многократно без ее корректировки. При необходимости многократного выполнения программы с различными значениями исходных данных целесообразнее организовать их ручной ввод. Для этого в QBasic существует оператор **INPUT**, который позволяет запросить у пользователя значение необходимой переменной во время работы программы.

Так, при выполнении строки

```
INPUT F
```

происходит останов программы, а на монитор выводится знак "?". После ввода пользователем некоторого значения (например, 8), переменной F присваивается данное значение, а затем выполнение программы продолжается.

Для удобства использования программы ее организуют в форме диалога с пользователем. Для этого каждый оператор INPUT сопровождается комментарием, который ставится перед именем переменной и заключается в кавычки, например:

```
INPUT "Введите площадь поперечного сечения выработки, м кв."; S
INPUT "Введите крепость пород"; f
INPUT "Введите мощность угольного пласта, м"; m
```

В результате на экран будут поочередно выдаваться указанные в качестве комментария запросы. После ввода пользователем последнего значения идет выполнение последующих строк программы. При этом переменные S, f и m становятся определенными, т.е. приобретают числовые значения, введенные с клавиатуры.

Один оператор INPUT позволяет ввести одновременно значения нескольких переменных, при этом вводимые значения должны отделяться запятыми, например:

```
INPUT "Задайте диаметр, мм, и плотность, кг/м куб, заряда"; D, RO
```

Концом ввода считается нажатие клавиши Enter.

Оператор INPUT может запрашивать не только числовые, но и любые символьные значения, например:

```
INPUT "Введите профиль и типоразмер проводника"; PR$, TIP$
```

После ввода с клавиатуры через запятую

РЕЛЬС, Р43

произойдет присвоение следующих значений символьным переменным: PR$="РЕЛЬС", TIP$="Р43".

При использовании оператора INPUT **следует помнить**, что:

– для исключения некорректности численных значений при выводе на экран комментария необходимо запрашивать не только наименование параметра, но и его единицу измерения, используемую в дальнейших расчетах;

– при запросе в одном операторе нескольких значений необходимо следить за строгим соответствием числа запрашиваемых и числа вводимых значений. Ввод большего или меньшего количества значений приведет к ошибке, о чем QBasic выдаст сообщение "Redo from start" (начните сначала);

– тип вводимых значений также должен соответствовать типам переменных, перечисленных в операторе INPUT. В случае попытки присвоить символьную константу числовой переменной возникнет ошибка и произойдет повторный запрос исходных данных;

– вводимые символьные значения не должны содержать в себе запятой, поскольку данный символ воспринимается как разделитель. При необходимости использования одной фразы, включающей запятую, в качестве символьной переменной, можно прибегнуть к оператору LINE INPUT.

3.1.4. Ввод исходных данных из файла

При использовании большого объема исходных данных ввод их вручную с клавиатуры во время выполнения программы бывает трудоемким и длительным. В данном случае можно использовать загрузку исходной информации из файла, в который предварительно должны быть записаны все необходимые значения в заданной последовательности.

Для чтения исходных данных из файла сначала необходимо открыть требуемый файл. Это позволяет сделать оператор **OPEN**, который указывает путь к файлу и его имя, тип доступа к файлу (в данном случае тип **INPUT**, определяющий последовательный ввод), а также присваивает файлу определенный номер.

Например, команда

```
OPEN "a:bwr.dat" FOR INPUT AS # 1
```

означает: открыть файл под номером "1" с именем bwr.dat, находящийся на диске "a", для ввода исходных данных.

Чтение данных из файла осуществляется оператором **INPUT #**. Так, например, строкой

```
INPUT #1, S, f, e, q1, a, W
```

организуется присвоение переменным S, f, e, q1, a и W числовых значений, перечисленных через запятую в файле с номером "1", в данном случае в файле bwr.dat.

Перед завершением программы (или сразу после ввода исходных данных) необходимо закрыть используемый файл с помощью оператора **CLOSE**.

Для этого после ключевого слова CLOSE указывается знак # и номер закрываемого файла, присвоенный при его открытии:

```
CLOSE # 1
```

При необходимости ввода исходных данных из разных файлов требуется открывать несколько файлов, каждому из которых присваивать свой номер, указываемый затем в операторе INPUT, например:

```
OPEN "C:\ LINA\ kod1.dat" FOR INPUT AS #1
OPEN "D:\ ALBERT\ kod2.dat" FOR INPUT AS #2
INPUT #1, ALFA, BETA, XI, FI, DELTA, OMEGA
INPUT #2, W, a, Dsv, Dpr, f, q0, e
CLOSE #1
CLOSE #2
```

В данном случае значения переменных ALFA, BETA, XI, FI, DELTA и OMEGA будут считываться из файла kod1.dat, расположенного в каталоге LINA на диске "C", а переменных W, a, Dsv, Dpr, f, q0, e – из файла kod2.dat, находящегося в каталоге ALBERT на диске "D".

При организации ввода исходных данных из файлов **необходимо обратить особое внимание на следующее**:

– до запуска программы необходимо создать все файлы с исходной информацией и разместить их в каталогах, указанных в программе. В случае отсутствия заданного файла или указания неверного каталога или подкаталога (пути к файлу) возникает аварийная ситуация, о чем QBasic выдает сообщение "File not found" (файл не найден);

– при указании неверного (несуществующего) диска или наличии синтаксических ошибок в команде (отсутствие знаков ":" или "\") выдается сообщение "Path not found" (строка не найдена);

– при попытке считывания данных с дискеты при ее отсутствии в дисководе выдается сообщение "Disk not ready" (диск не готов);

– количество числовых или символьных значений в файле должно быть не меньше количества переменных в строке INPUT, в противном случае возникает останов программы и выдается сообщение "Input past end of file" (ввод после конца файла);

– недопустимо смешивать типы переменных с типами считываемых из файла значений. При попытке присвоить числовой переменной символьного значения останова программы не происходит, но этой переменной присваивается числовое значение, равное нулю;

– при открытии нескольких файлов им необходимо присваивать различные номера. При попытке открытия файла с номером ранее открытого выдается сообщение "File already open" (файл уже открыт). Однако после закрытия предыдущего файла, следующему открываемому файлу в этой программе можно присваивать ранее существовавший номер;

– для закрытия сразу всех открытых файлов можно использовать оператор CLOSE без указания номера файла.

3.2. Организация вывода результатов

3.2.1. Вывод результатов выполнения программы на экран и принтер

Для вывода данных на экран дисплея в процессе выполнения программы используется оператор **PRINT**. Аналогично оператору INPUT, он может включать не только список переменных, но и сопровождающий данный список комментарий, заключаемый в кавычки, например:

```
PRINT V, V1
PRINT Kut; Qp; Qsp
PRINT "Крепость пород – "; f
PRINT "Тип бурильной установки -"; TBU$
```

Как видно из примеров, между элементами списка вывода может стоять или точка с запятой, или запятая. Если между элементами стоит точка с запятой, то второй элемент выводится на экран через позицию вслед за первым. Если после первого элемента стоит запятая, то второй элемент выводится на экран в следующей зоне (вся строка QBasic разделяется на 5 зон по 14 позиций в каждой зоне).

Один оператор PRINT может содержать несколько чередующихся текстовых элементов и имен переменных, отделяемых запятой или точкой с запятой.

Например, в результате выполнения части программы

```
L = 500: S = 12: m = 1
f = 8: a = 0.5: W = 0.6
Qsp = 5
PRINT "L = "; L; "м  "; " S = "; S; "м кв. "; " m = "; m; "м"
PRINT "f = ";f, "a = ";a; "м", "W = "; W; "м"
PRINT "Количество воздуха для проветривания -"; Qsp; "м куб./с"
```

на экран выводится следующая информация:

L = 500 м S = 12 м кв. m = 1 м

f = 8 a = 0.5 м W = 0.6 м

Количество воздуха для проветривания – 5 м куб./с

После выполнения какого-либо оператора PRINT происходит перевод курсора на экране дисплея на новую строку, таким образом, следующий оператор PRINT выводит значения с новой строки. Однако, если после окончания

списка ввода поставить запятую или точку с запятой, то следующий оператор PRINT будет выводить значения в той же строке, например:

```
V0 = 1.5: K = 4: N = 80: t = 40
PRINT "V0 = "; V0; "м/мин "; " K = "; K;
PRINT " N = "; N; " t = "; t; "мин"
```

На экране при этом появятся значения:

V0 = 1.5 м/мин K = 4 N = 80 t = 40 мин

Пустой оператор PRINT (без списка переменных и комментариев) вызывает перевод строки – печать пустой строки. Например, строка

```
PRINT: PRINT: PRINT
```

обеспечивает печать трех пустых строк, т.е. создает пробел из трех строк.

Для вывода информации на печатающее устройство используется оператор **LPRINT**, формат которого аналогичен оператору PRINT.

При организации вывода информации с помощью операторов PRINT или LPRINT **необходимо помнить следующее**:

– максимальное количество символов, размещающееся в одной строке на экране (для большинства современных мониторов) – 80. Если оператор PRINT предусматривает вывод большего количества символов, то после каждых 80 знаков будет происходить автоматический перенос на следующую строку, таким образом, слишком длинная строка будет напечатана в виде нескольких строк;

– при выводе подряд идущих символьных переменных, имена которых разделены точкой с запятой, они печатаются слитно, поэтому в начало или конец таких переменных целесообразно включать пробелы;

– для быстроты и удобства ввода программы в компьютер вместо ключевого слова PRINT можно вводить знак "?", который во время компиляции QBasic автоматически заменяет словом PRINT;

– выводимые имена переменных после слова PRINT можно разделять пробелами. Это равносильно разделению переменных точкой с запятой. Такое разделение QBasic осуществляет автоматически во время компиляции;

– при использовании оператора LPRINT в случае неисправности или неготовности печатающего устройства выполнение программы прерывается, и выдается сообщение "Device fault" (отказ устройства).

3.2.2. Использование оператора PRINT для выполнения вычислительных операций, работа в режиме калькулятора

Оператор PRINT может быть использован и для печати результата вычислений по формуле, стоящей после ключевого слова PRINT, например:

```
Ks = 0.5: d = 0.036: ro = 1000: W = 0.6: TETA = 0.95
PRINT (Ks*d^2*ro)/(W^2*TETA)
```

В результате будет произведен расчет по заданной формуле и выдано полученное значение:

1.894737

Эту же строку можно сопровождать комментариями, например:

```
PRINT "Удельный расход ВВ равен"; (Ks*d^2*ro)/(W^2*TETA);"кг/м куб."
```

В этом случае на экран выйдет сообщение:

Удельный расход ВВ равен 1.894737 кг/м куб.

Как видно, применение такого способа расчета несколько сокращает текст программы и является предпочтительным при несложных вычислениях.

Еще более простым способом расчета является *работа в режиме калькулятора*, позволяющая выполнять вычислительные действия с использованием всех стандартных функций QBasic фактически без набора программы. Для этого в специальном окне QBasic (Immediate) необходимо ввести соответствующую строку с заданными действиями. В нашем случае строка будет иметь вид

? (0.5*0.036^2*1000)/(0.6^2*0.95)

После нажатия клавиши Enter получаем ответ.

Обычно режим калькулятора используется для однократных вычислений малой сложности, а также может быть очень полезным в ходе редактирования и отладки программы для проверки корректности ее работы.

3.2.3. Вывод результатов выполнения программы в файл

При большом объеме результатов выполнения программы их вывод на экран бывает не всегда удобным, поскольку количество строк, одновременно помещающихся на экране, обычно не превышает 23-27 (в зависимости от режима работы монитора). Поэтому при выводе большого объема информации на экран приходится включать в программу промежуточные остановы и выводить результаты по частям. Кроме того, выводимая на экран информация нигде не сохраняется после запуска следующей части программы. Вывод же всех результатов расчета на принтер часто бывает ненужным или экономически нецелесообразным.

Для избежания названных неудобств можно организовать вывод результатов в файл. Аналогично чтению данных из файлов (п. 3.1.4), для печати результатов в файл предварительно открывается или создается этот файл. Но здесь в качестве типа доступа к файлу вместо INPUT указывается тип **OUTPUT**, определяющий последовательный вывод. Например, строка

```
OPEN "C:\LINA\INFORM\lab4.txt" FOR OUTPUT AS #1
```

означает открыть файл lab4.txt под номером 1, находящийся на диске C, в каталоге LINA, в подкаталоге INFORM для вывода результатов.

Печать результатов в файл осуществляется оператором **PRINT #** с указанием номера, присвоенного файлу при открытии. Например, строка

```
PRINT #1, Ssv, B, H,  Vmax
```

производит запись в файл под номером 1 (в данном случае это файл lab4.txt) значения переменных Ssv, B, H и Vmax.

Перед завершением программы (или после окончания записи информации в файл) он должен быть закрыт оператором CLOSE # (п.3.1.4).

При записи информации в файл **полезно помнить следующее**:

– в случае указания в операторе OPEN имени файла, несуществующего к моменту выполнения программы, этот файл будет автоматически создан в заданном каталоге и в него будет записана вся информация, указанная оператором PRINT # ;

– если же в операторе OPEN указан несуществующий диск или каталог, QBasic выдаст сообщение об ошибке "Path not found"(строка не найдена);

– при указании в операторе OPEN имени уже существующего файла во время записи информации в файл все ранее существовавшие в нем данные будут уничтожены;

– все возможности оператора PRINT (пп. 3.2.1 и 3.2.2) распространяются и на оператор PRINT # ;

– при использовании нескольких операторов PRINT # с одним и тем же номером информация записывается в файл построчно за исключением случая, когда список вывода оканчивается точкой или точкой с запятой (при этом вся информация записывается в файл в одну строку);

– при выводе длинных строк (более 80 знаков) все символы печатаются в одну строку, причем длина строки практически не ограничена;

– при попытке записи на переполненный диск QBasic выдает сообщение об ошибке "Disk full" (диск заполнен).

3.2.4. Организация форматированного вывода

При работе с вещественными переменными оператор PRINT выдает их значения с большой точностью, что не всегда нужно, а иногда и просто не позволяет организовать желаемый вид печати результатов (например, при выводе таблиц). Для вывода на экран строки символов по заданному формату используют оператор **PRINT USING**.

При выводе числовых значений переменных для указания формата используются знаки #, например, оператор

```
PRINT USING "###.##"; RO
```

выведет числовое значение переменной RO по указанному формату, т.е. с тремя знаками целой части и двумя знаками дробной.

Как и оператор PRINT, оператор PRINT USING позволяет включать комментарии, например,

```
PRINT USING "Диаметр ствола в проходке Dпр = #.## м"; DPR
```

а также выводить несколько переменных в одной строке

```
PRINT USING "ALFA = ## град.; P = ##.## МПа; B = ##.# м";ALFA;P;B
```

выводить результаты выполнения программы на принтер

```
LPRINT USING "Производительность бурения P = ##.## м/мин"; PB
```

или в файл

```
OPEN "E:\QB45\PROGR\lab4.txt" FOR OUTPUT AS #2
PRINT #2, USING "Скорость проходки Vпр = ###.# м/мес"; Vpr
```

При необходимости вывода символьной переменной по заданному формату используются знаки \ \, причем количество печатаемых символов равно числу пробелов между слэшами плюс два символа.

Например, при выполнении программы

```
VV$ = "Горный факультет"
PRINT USING "\    \"; VV$
```

на печать будет выдано 6 первых знаков (т.к. между слэшами стоит 4 пробела), т.е. одно слово "Горный".

Если слэши набраны без пробелов, то выводятся два начальных символа строки. Если поле вывода больше, чем длина строки, то строка выравнивается влево, а справа печатаются пробелы.

Для вывода только первого символа строки служит знак !, например, оператор

```
PRINT USING "!"; "Г-III-2а"
```

выдаст на печать только один символ "Г".

При организации форматированного вывода на печать **следует также помнить**, что:

– при попытке вывести число, целая часть которого занимает больше знаков, чем предусматривает оператор PRINT USING, QBasic игнорирует заданный формат и выдает число в стандартном виде, при этом перед числом появляется знак % (не соблюден заданный формат);

– при выводе числа с целой частью, занимающей меньшее количество знаков, чем предусматривает PRINT USING, "лишние" знаки # заменяются пробелами;

– дробная часть в любом случае округляется с заданной точностью, при этом, если первое число отсекаемой части – 5 и более, то последняя выводимая на печать цифра увеличивается на 1 (округляется в большую сторону), в обратном случае – остается неизменной. Таким образом, для вывода на печать округленного значения (при неиспользовании его в дальнейших расчетах) необязательно использовать функции округления (п. 2.8), а достаточно указать необходимый формат выводимого значения оператором PRINT USING.

3.2.5. Вспомогательные операторы и функции, используемые при выводе

Для организации удобного и наглядного вывода информации на экран используют ряд вспомогательных операторов, управляющих положением курсора на экране. Рассмотрим некоторые из них.

Весь монитор разделен условно на знакоместа (чаще всего 80 – по горизонтали и 25 – по вертикали). Оператор **LOCATE** передвигает курсор в указанную позицию. Например, при выводе сообщения "Введите исходные данные" в центре экрана необходимо предварительно переместить курсор в заданную позицию:

```
LOCATE 12, 28
PRINT "Введите исходные данные"
```

При этом курсор передвигается в 12-ю строку (считая сверху) и 28-й столбец (слева), и с этого места печатается сообщение "Введите исходные данные". Следующий оператор PRINT (при отсутствии перед ним "своего" LOCATE) осуществляет печать в следующей (13-й) строке, начиная с 1-го столбца.

Функция **TAB** (n) сдвигает строку вывода на n позиций при использовании операторов PRINT или LPRINT. Например, результатом выполнения строк

```
H = 4.168
PRINT TAB(30) "Шаг армировки – ";H; "м"
```

будет

Шаг армировки – 4.168 м

т.е. печать строки начнется с 31-й колонки.

Функция **SPC** (n) осуществляет ввод n пробелов в текущей строке, например, строка программы

```
PRINT "Группа ВВ"; SPC(15); "Непредохранительные"
```

выведет на печать

Группа ВВ Непредохранительные

а в результате выполнения части программы

```
PM$ = "2ПНБ-2У"
OPEN "C:\LINA\E321.TXT" FOR OUTPUT AS #1
PRINT #1, SPC(10);"Тип породопогрузочной машины"; SPC(15); PM$
```

в файл E321.txt, находящийся в каталоге LINA на диске C, будет выведена строка

Тип породопогрузочной машины 2ПНБ-2У

При организации форматированного вывода вспомогательные функции (TAB, SPC) должны ставиться после кавычек, перед именем выводимой переменной.

Причем функция TAB может использоваться для указания количества позиций, отступаемых только от начала строки

Например, в результате выполнения строк

```
L=2.35
PRINT USING "Длина врубовых шпуров – ##.## м"; TAB(25); L
```

печать начнется с 26-й позиции:

Длина врубовых шпуров – 2.35 м

Для форматированного вывода на печать в одну строку нескольких переменных, разделенных заданным количеством пробелов, удобнее использовать функцию SPC:

```
A=2.25: B=11: C=0.392
PRINT USING "##.##  ##  #.##"; SPC(10);A;SPC(20);B;SPC(25);C
```

Результатом выполнения данного фрагмента будет строка

$\underbrace{\quad}_{\text{10 позиций}}$ 2.25 $\underbrace{\quad\quad\quad}_{\text{20 позиций}}$ 11 $\underbrace{\quad\quad\quad\quad}_{\text{25 позиций}}$ 0.39

3.3. Лабораторная работа №1

Тема. Программирование алгоритмов линейной структуры

Цель работы. Научиться составлять алгоритмы линейной структуры и программы расчета параметров по заданным формулам.

Задание. Согласно варианту задания составить блок-схему алгоритма и программу на языке QBasic по расчету указанных параметров. Организовать ручной ввод исходных данных и форматированный вывод на печать результатов расчета с указанием наименования рассчитанного параметра и единиц измерения. При необходимости предусмотреть перевод значений параметров к единой системе измерений.

3.3.1. Варианты лабораторной работы

Вариант 1

Рассчитать требуемую производительность подъемной установки $A_ч$, т/ч, по формуле

$$A_ч = \frac{k_p \cdot A}{N \cdot t},$$

где k_p – коэффициент резерва подъема, учитывающий возможность увеличения добычи по сравнению с проектной, а также неравномерность по-

ступления грузов к стволу (в угольной промышленности $k_р = 1{,}5$);

A – годовая проектная мощность шахты, т/год;

N – число рабочих дней в году по выдаче полезного ископаемого (принять $N = 300$);

t – число часов работы подъемной установки по выдаче полезного ископаемого в сутки, ч (исходя из трехсменной работы $t = 15$ ч).

Годовую проектную мощность шахты выбрать самостоятельно из типового ряда.

Вариант 2

Рассчитать удельный расход ВВ, кг/м^3, определяемый формулой

$$q = k_з \cdot d^2 \cdot \rho \cdot W^{-2} \cdot \eta^{-1},$$

где $k_з$ – коэффициент заполнения шпура, $k_з = 0{,}5 \div 0{,}6$;

d – диаметр патрона ВВ, м;

ρ – плотность патронированного ВВ, кг/м^3;

W – линия наименьшего сопротивления, м;

η – коэффициент использования шпура.

Необходимые исходные данные задать самостоятельно.

Вариант 3

Рассчитать продолжительность заряжания и взрывания шпуров при проходке вертикального ствола, мин, определяемую формулой

$$T_з = \frac{N}{\alpha_з \cdot N_з} \cdot \tau_з,$$

где N – число шпуров;

$\alpha_з$ – коэффициент средней численности проходчиков на заряжании шпуров (по данным наблюдений в среднем равен 0,8);

$N_з$ – число проходчиков, занятых при заряжании, определяемое из соотношения $N_з = \dfrac{S_{пр}}{5}$,

$S_{пр}$ – площадь поперечного сечения ствола в проходке, м^2;

Принятое число проходчиков округлить до ближайшего большего целого;

$\tau_з$ – время заряжания одного шпура в стволе с учетом забойки шпура гранулированным шлаком, монтажа электросети и ее проверки, мин, ориентировочно $\tau_з = 4 + 1{,}1 \cdot l_{шп}$.

Здесь $l_{шп}$ – длина шпура, м.

Предусмотреть ручной ввод следующих исходных данных: площади поперечного сечения ствола в проходке, длины и количества шпуров. На печать вывести продолжительность взрывных работ, выраженную в часах и минутах.

Вариант 4

Рассчитать необходимую ширину барабана подъемной машины, мм, по формуле

$$B = \left(\frac{H_{ст} + h_р + h_з}{\pi \cdot D_б} + n_{тр}\right) \cdot \left(\frac{d_к + \varepsilon}{z}\right),$$

где $H_{ст}$ – конечная глубина ствола, м;

$h_р$ – высота разгрузки, м;

$h_з$ – запас каната на испытание, м;

$D_б$ – диаметр барабана подъемной машины, м;

$n_{тр}$ – число витков трения;

$d_к$ – диаметр подъемного каната, мм;

ε – зазор между витками каната, мм;

z – число слоев навивки.

Расчет произвести для следующих данных: $h_р = 15$ м, $h_з = 40$ м, $n_{тр} = 3$, $\varepsilon = 2$ мм, $z = 2$. Предусмотреть ручной ввод параметров: $H_{ст}$, $D_б$, $d_к$, значения которых подобрать самостоятельно.

На печать вывести диаметр и ширину барабана подъемной машины.

Вариант 5

Рассчитать необходимое число автомашин для транспортирования породы в отвал по формуле

$$n = \frac{P_п}{Q},$$

где $P_п$ – производительность подъема породы в первой фазе погрузки, м^3/ч;

Q – производительность автосамосвала, м^3/ч, определяемая из выражения $Q = \dfrac{G}{k \cdot \gamma_п \cdot \dfrac{1{,}8 \cdot z}{v} + \Sigma t}$,

G – грузоподъемность самосвала, т

k – коэффициент неравномерности работы автомашины, $k = 1{,}2$;

$\gamma_п$ – плотность разрыхленной породы, т/м^3;

z – расстояние от ствола до породного отвала, км;

v – средняя скорость движения груженого и порожнего автосамосвала, км/ч;

Σt – продолжительность разгрузки и погрузки автомашины, ч.

Расчет произвести для следующих данных: $P_п = 60$ м^3/ч; $\gamma_п = 1{,}6$ т/м^3; $v = 25$ км/ч; $\Sigma t = 0{,}07$ ч. Организовать ручной ввод параметров G и z, значения которых задать самостоятельно. К расчетному числу автосамосвалов добавить 20 % резерва.

Вариант 6

Рассчитать дебит водопонизительных скважин Q_0, м^3/с, при искусственном понижении уровня подземных вод для условий безнапорного водоносного горизонта:

$$Q_0 = \frac{k \cdot \left(H_0^2 - z_0^2\right)}{0{,}73 \cdot \lg\left(\frac{R}{\rho}\right)},$$

где k – коэффициент фильтрации, м/с;

H_0 – мощность водоносного слоя в месте проходки ствола, м;

z_0 – максимальная высота участка над подстилающими породами, не поддающаяся обезвоживанию, м;

R – радиус влияния скважин, м, ориентировочно $R = 588 \cdot S \cdot \sqrt{H_0 \cdot k}$,

S – глубина откачки в центре водопонизительной установки, м;

ρ – радиус окружности расположения скважин, м, $\rho = \frac{D_{пр}}{2} + 4$,

$D_{пр}$ – диаметр ствола в проходке, м.

Расчет произвести для следующих исходных данных: $H_0 = 22$ м; $z_0 = 1$ м; $S = 22$ м; $k = 0{,}0005$ м/с. Предусмотреть ручной ввод величины $D_{пр}$.

Вариант 7

Рассчитать толщину ледопородной подпорной стены E_c, м, устраиваемой при проведении выработки открытым способом с применением замораживания пород, используя выражение

$$E_c = 0{,}7 \cdot h \cdot \sqrt{A},$$

где h – высота обнажения подпорной ледопородной стены, м;

A – коэффициент горизонтального распора, $A = tg^2\left(45 - \frac{\varphi}{2}\right)$,

φ – угол внутреннего трения породы, °.

Исходные данные задать самостоятельно.

Вариант 8

Рассчитать радиус сферической поверхности R, м, и угол наклона боковой поверхности к вертикали α,°, тампонажной подушки, сооружаемой при цементации пород из забоя ствола, если:

$$R = \frac{r^2 + \eta^2}{2 \cdot \eta}, \quad \alpha = \arcsin\frac{2 \cdot r \cdot \eta}{r^2 + \eta^2},$$

где r – радиус ствола в свету, м;

η – высота сферической поверхности тампонажной подушки, м, ориентировочно определяемая из соотношения $\eta = 0{,}15 \cdot D_0$, где D_0 – диа-

метр ствола в проходке, м

Исходные данные задать самостоятельно. На печать вывести значения параметров R и α.

Вариант 9

Рассчитать теоретическую производительность бурового комбайна $Q_{теор}$, м^3/мин, используя выражение

$$Q_{теор} = S \cdot v,$$

где S – площадь сечения выработки вчерне, м^2;

v – скорость подачи исполнительного органа комбайна, м/мин, $v = 0{,}06 \cdot n_{и.о} \cdot h_{max} \cdot m$,

$n_{и.о.}$ – частота вращения исполнительного органа, Гц;

h_{max} – максимальная толщина стружки, мм;

m – число резцов в линии разрушения.

Расчет произвести для следующих исходных данных: диаметр исполнительного органа комбайна D = 4,5 м; $n_{и.о.}$ = 0,105 Гц; h_{max} = 2,4 мм; m = 1.

Вариант 10

Рассчитать необходимый угол наклона оконтуривающего шпура α,°, используя выражение

$$\alpha = \arccos\frac{\lambda + a}{l_{шп}},$$

где λ – расстояние между оконтуривающим шпуром и проектным контуром выработки, м;

a – величина выхода забоя шпура за контур выработки, м;

$l_{шп}$ – длина шпура, м.

Исходные данные задать самостоятельно. На печать вывести значение угла наклона шпура, выраженное в градусах.

Вариант 11

Рассчитать толщину кольцевой изоляционной завесы C_m, м, образуемой при цементации пород через скважины из забоя ствола, используя формулу

$$C_m = \frac{D_1}{2} \cdot \left(\sqrt{\frac{m \cdot R_n}{m \cdot R_n - 2 \cdot \lambda \cdot P_г}} - 1 \right),$$

где D_1 – диаметр зоны растрескивания, м, $D_1 = (D_o + 2 \cdot C) \cdot \psi + 2 \cdot r_p$,

D_o – диаметр ствола в свету, м;

C – толщина крепи, м;

ψ – коэффициент перебора породы;

r_p – величина зоны растрескивания горных пород взрывными работами, м.

m – коэффициент условий работы материала завесы;

R_n – расчетное сопротивление затампонированных пород сжатию, МПа;

λ – коэффициент перегрузки;

$P_г$ – гидростатическое давление подземных вод, МПа.

Расчет произвести для следующих исходных данных: r_p = 1,2 м; m = 0,7; R_n = 18,6 МПа; λ = 1,25; $P_г$ = 2,2 МПа. Остальные параметры, необходимые для расчета, задать самостоятельно.

Вариант 12

Рассчитать среднее значение оптимального интервала замедления $\tau_{ср}$, мс, при короткозамедленном взрывании, используя эмпирическую зависимость

$$\tau_{ср} = \frac{31,5}{\sqrt[4]{1,3 \cdot f}} \cdot W - 6 \cdot \sqrt[4]{1,3 \cdot f} + 9,6 \ ,$$

где W – линия наименьшего сопротивления, м;

f – коэффициент крепости пород.

Исходные данные для расчета задать самостоятельно.

Вариант 13

Рассчитать оптимальную глубину шпура l, м, используя выражение:

$$l_{шп} = \sqrt[3]{\left(\frac{4 \cdot v_0 \cdot k}{N} \cdot \Sigma t_{подг.-закл} \right)^2} \ ,$$

где v_0 – скорость бурения шпура глубиной 1 м, м/мин;

k – число бурильных машин, одновременно работающих в забое;

N – число шпуров в комплекте;

$\Sigma t_{подг.-закл}$ – сумма затрат времени на выполнение подготовительно-заключительных операций проходческого цикла, мин.

Исходные данные задать самостоятельно.

Вариант 14

Рассчитать необходимое число погрузочных бункеров N, используя выражение

$$N = \frac{Q_б}{V_б},$$

где $V_б$ – вместимость одного бункера, м^3,

$Q_б$ – необходимая суммарная вместимость погрузочных бункеров,

$$Q_б = \frac{k_1 \cdot k_2 \cdot P}{n \cdot \gamma} - \frac{t \cdot p}{\gamma},$$

k_1 – коэффициент неравномерности подачи порожняка, k_1 = 1,25;

k_2 – коэффициент неравномерности работы шахты, k_2 = 1,2;

P – суточная производительность шахты, т/сут;

n – число составов порожняка, подаваемых на шахту в сутки;

γ – плотность угля, т/м^3;

t – продолжительность погрузки состава, ч;

p – часовая производительность шахты, т/ч.

Расчет произвести для V = 800 м^3. Исходные данные для расчета $Q_б$ задать самостоятельно.

Вариант 15

Рассчитать глубину шпуров $l_{шп}$, м, при проведении горизонтальной выработки, используя выражение

$$l_{шп} = \frac{T_ц - (N \cdot t_{зар} + t_{пр})}{\frac{N}{k \cdot v} + \frac{\eta \cdot S \cdot \cos\alpha}{P}},$$

где $T_ц$ – продолжительность проходческого цикла, ч;

N – число шпуров в комплекте;

$t_{зар}$ – продолжительность заряжания одного шпура, ч;

$t_{пр}$ – продолжительность проветривания забоя, ч;

k – число бурильных машин;

v – скорость бурения шпуров, м/ч;

η – коэффициент использования шпура;

S – площадь поперечного сечения выработки в проходке, м^2;

α – угол наклона шпуров,°;

P – производительность погрузки породы (в массиве), м^3/ч.

Исходные данные для расчета задать самостоятельно.

3.3.2. Пример выполнения лабораторной работы

Задание: Рассчитать техническую производительность подъемной установки $Р_{тех}$, м^3/ч, используя следующую формулу:

$$P_{тех} = \frac{3600 \cdot Q_б \cdot k_1}{T_ц \cdot k_2 \cdot k_3},$$

где $Q_б$ – вместимость бадьи, м^3;

k_1 – коэффициент заполнения бадей горной массой;

k_2 – коэффициент неравномерности работы подъемной машины в течение всего времени подъема горной массы;

k_3 – коэффициент разрыхления породы;

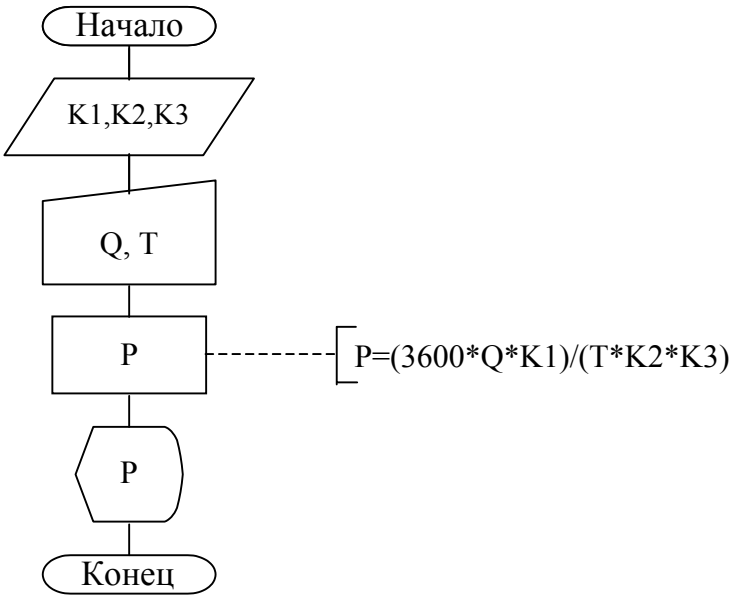
$T_ц$ – фактическая длительность цикла выдачи бадьи, с.

Расчет произвести для следующих исходных данных: k_1 = 0,85; k_2 = 1,15; k_3 = 2. Предусмотреть ручной ввод параметров $Q_б$ и $T_ц$, значение которых задать самостоятельно.

Присваиваем имена переменным и составляем таблицу идентификаторов:

Переменная	Идентификатор
$P_{тех}$	P
$Q_б$	Q
k_1	K1
k_2	K2
k_3	K3
$T_ц$	T

Составляем блок-схему алгоритма:



Составляем программу на языке QBasic:

```
CLS
DATA 0.85, 1.15, 2
READ K1, K2, K3
INPUT "Введите вместимость бадьи, м куб."; Q
INPUT "Введите продолжительность цикла подъема, с"; T
P = ( 3600 * Q * K1 ) / ( T * K2 * K3)
PRINT USING "Производительность подъема P = ###.## м куб./ч"; P
END
```

Результат выполнения программы (при Q = 3 м^3 и T = 120 с):
Производительность подъема P = 33.26 м куб./ч.

4. ПРОГРАММИРОВАНИЕ АЛГОРИТМОВ РАЗВЕТВЛЯЮЩЕЙСЯ СТРУКТУРЫ

Алгоритмом разветвляющейся структуры будем считать алгоритм, имеющий две или более ветви (последовательности действий), выполняемые в зависимости от соблюдения (несоблюдения) заданных условий.

4.1. Операции отношения в QBasic

Для проверки заданного условия, т.е. для сравнения арифметических выражений, значений числовых или символьных переменных в QBasic используют шесть операций отношения, перечисленных в табл. 5.

Таблица 5

Операторы отношения в QBasic

Оператор	Значение	Выражение в QBasic
=	Равенство	X = Y
< > или > <	Неравенство	X < > Y
<	Меньше	X < Y
>	Больше	X > Y
<= или =<	Меньше или равно	X <= Y
>= или =>	Больше или равно	X >= Y

Операторы отношения используют два значения. Результатом операции отношения является либо "TRUE" (ИСТИНА – ненулевое значение), либо "FALSE" (ЛОЖЬ – ноль).

При использовании операций отношения **необходимо помнить следующее**:

– если в одном выражении встречаются и арифметические операторы, и операторы отношения, то первыми выполняются арифметические операторы. Например, в выражении M1 + M2 <= P3*L3 + Q*L4^2 сначала вычисляется значение выражения M1 + M2, затем значение P3*L3 + Q*L4^2 и только потом выполняется операция отношения <=;

– при использовании операций = и < > результаты, отличающиеся даже на ничтожно малую величину, будут считаться неравными;

– по отношению к символьным переменным имеют смысл только операции = и < >, причем результатом операции "=" будет "ИСТИНА" только в случае полного совпадения всех символов (с учетом регистра букв);

– не следует путать знак "=", используемый в качестве оператора присвоения и в качестве символа отношения. В последнем случае данный знак

используется только совместно с условными операторами передачи управления, о которых речь пойдет ниже.

4.2. Организация условного и безусловного переходов при наложении одного условия

Для организации "ветвления" программы, т.е. передачи управления по условию на ту или иную ветку, в QB существует несколько структур.

Самой простой из них является **IF-THEN-ELSE** (ЕСЛИ-ТО-ИНАЧЕ). В общем виде она запишется так:

IF <условие> THEN <команда 1> ELSE <команда 2>

Данная условная структура осуществляет выполнение "команды 1" в случае выполнения "условия" или "команды 2" – в противном случае. Например, в программе

```
INPUT "Введите площадь поперечного сечения тоннеля"; S
INPUT "Введите крепость пород"; f
IF f => 9 THEN N = 37.6 + 1.36*S ELSE N = 30.9 + S
PRINT USING "Число шпуров в забое тоннеля N = ###"; N
```

число шпуров в забое тоннеля будет определяться по тому или иному выражению в зависимости от заданной пользователем крепости пород.

После ключевых слов THEN и ELSE может стоять не только оператор присвоения, но и любой другой оператор или функция, например:

```
⋮
Fd = FI*Rbt*Um*H0
IF F<= Fd THEN PRINT "Условие выполнено" ELSE PRINT "Увеличьте марку бетона"
```

Если условие, в зависимости от которого осуществляется "ветвление", задается в словесной форме (например, указывается тип ВВ, наименование или типоразмер профиля балки, тип машины и т.п.), можно поступать следующим образом:

```
INPUT "Введите площадь сечения выработки в проходке, м кв."; S
PRINT "Введите код, соответствующий типу вруба:"
INPUT " 1 – для отрывающих врубов, 2 – для дробящих"; TIP
IF TIP = 1 THEN A = 0.5 ELSE A = 0.75
L = A*SQR(S)
PRINT USING "Длина шпуров L = #.## м"; L
```

В данном случае типу вруба будет соответствовать определенный числовой код, по которому можно изменять порядок выполнения программы с помощью операторов условного перехода.

Условный оператор может быть записан в сокращенной форме, т.е. без использования ключевого слова ELSE, например:

```
IF SIG > SD THEN PRINT "Напряжения в конструкции выше допустимых !"
```

При несоблюдении условия в данном случае оператор PRINT не выполняется, а управление переходит к следующей строке программы.

Здесь стоит также упомянуть об операторе безусловного перехода GOTO, который остался в QBasic от первого поколения языка Basic и считается устаревшей конструкцией. Применение такого оператора является нежелательным, поскольку программу с обилием GOTO трудно отлаживать и модифицировать, так как она не имеет четкой структуры. Тем более QBasic обладает широкими возможностями в организации любых условных и циклических структур без использования оператора GOTO. Тем не менее для малоопытного программиста применение оператора GOTO в несложных по структуре программах может показаться наиболее простым и доступным способом организации условных и безусловных переходов. Оператор безусловного перехода **GOTO** <n> осуществляет переход выполнения программы к строке с номером или меткой n. Оператор GOTO, используемый в сочетании с конструкцией IF-THEN-ELSE, выполняет функцию оператора условного перехода. В таком случае ключевые слова THEN и второе GOTO (после ELSE) могут быть опущены, например:

```
IF F > 9 GOTO 10 ELSE 20
```

При выполнении условия F>9 управление передается строке с номером 10, в обратном случае – строке с номером 20.

И все же советуем отказаться от использования оператора GOTO, поскольку это свидетельствует о том, что программист не полностью овладел всем богатством управляющих структур языка QBasic или не знает их возможностей.

4.3 Организация условного перехода при наложении нескольких условий

Вышеописанная линейная условная структура IF-THEN-ELSE обычно используется, когда проверяется одно или два условия в программе. В случае же необходимости проверки более сложных условий (наличия в алгоритме более двух ветвей) применение нескольких таких конструкций значительно усложняет и загромождает программу. Для более легкой и корректной организации сложных условных структур в QBasic предусмотрены следующие конструкции.

4.3.1. Блочная форма оператора IF

Блочная конструкция оператора IF (**IF-THEN-ELSEIF-THEN…-ELSE-END IF**) применяется в том случае, если при несоблюдении "**условия 1**" следует сразу же проверить "**условие 2**"; а при несоблюдении "условия 2" – проверить "**условие 3**" и т. д., причем при выполнении какого-либо условия выполняется определенный блок операторов.

Общий вид такой конструкции:

```
IF <условие 1> THEN
   <операторы 1>
ELSE IF <условие 2> THEN
   <операторы 2>
.
.
.
ELSE
   <операторы n>
END IF
```

Рассмотрим использование блочной конструкции IF на примере.

Пример. Рассчитать производительность ленточного конвейера P, м^3/ч, используя выражение

$$P = 3600 \cdot F \cdot \gamma \cdot v$$

где γ – насыпная плотность материала, т/м^3;

v – скорость движения ленты, м/с;

F – площадь сечения материала, расположенного на ленте, м^2, эта величина принимается в зависимости от формы конвейерной ленты из соотношений:

– для плоской ленты $F = 0{,}05 \cdot B^2$;

– для плоской ленты с бортами $F = 0{,}05 \cdot B^2 + h \cdot (B - b_0)$;

– для желобчатой ленты $F = 0{,}11 \cdot B$

где B – ширина ленты, м;

h, b_0 – высота и толщина бортов, м.

Здесь, в зависимости от формы конвейерной ленты, изменяется не только расчетная формула, но и список вводимых переменных, причем алгоритм должен предусматривать сразу три ветви возможного хода программы, поэтому в данном случае целесообразнее применить блочный синтаксис оператора IF.

Программа расчета в будет выглядеть следующим образом:

```
REM Расчет производительности ленточного конвейера
CLS
GAM = 2.2: V = 1.2
INPUT "Введите ширину конвейерной ленты, м"; B
PRINT "Введите код, соответствующий типу ленты:"
PRINT "1 – для плоской ленты"
PRINT "2 – для плоской ленты с бортами"
INPUT "3 – для желобчатой ленты"; TL
IF TL = 1 THEN
    F = .05 * B ^ 2
ELSEIF TL = 2 THEN
    INPUT "Введите высоту и толщину бортов, м"; H, B0
    F = .05 * B ^ 2 + H * (B – 2 * B0)
ELSEIF TL = 3 THEN
    F = .11 * B
```

```
ELSE
    PRINT "Введен несуществующий код"
    END
END IF
P = 3600 * F * GAM * V
PRINT USING "Производительность конвейера – ####.# м куб./ч"; P
```

Из текста программы видны особенности применения блочного условного оператора. При исполнении этого оператора сначала проверяется первое условие (TL = 1, что соответствует плоской форме ленты). Если оно истинно, то выполняется оператор, следующий за ключевым словом THEN, в противном случае проверяется каждое из условий ELSEIF. При выполнении такого условия выполняются операторы данного блока (например, при выполнении условия TL = 2, что соответствует плоской ленте с бортами, запрашиваются параметры бортов, и по соответствующей формуле рассчитывается F). Если ни одно из условий ELSEIF не выполнено, выполняются операторы блока ELSE (в данном случае выдается сообщение о задании неверного кода, и происходит останов программы).

При использовании описанных операторов условной структуры **следует помнить**, что:

– в блочную структуру IF можно вставлять любое количество условий ELSEIF;

– каждый из блоков может также содержать вложенные блочные структуры;

– операторы IF, ELSE, ELSEIF и END IF должны быть первыми операторами в строке;

– каждый блок должен заканчиваться оператором END IF;

– условный оператор с блочной структурой имеет значительные преимущества перед однострочным оператором IF: возможность располагать блоки операторов в нескольких строках программы, возможность сложного логического ветвления программы, простое и компактное написание текста программы; более простая отладка программы. Поэтому линейный (однострочный) оператор условной структуры IF-THEN-ELSE целесообразно применять лишь для проверки одного или двух условий, в противном случае используется блочная форма IF.

Однако при проверке сложных условий лучше использовать не структуру IF, а более современную конструкцию языка QBasic SELECT…END SELECT, речь о которой пойдет ниже.

4.3.2. Блочная структура выбора вариантов SELECT CASE-CASE-END SELECT

При наложении большого количества условий целесообразно использовать конструкцию **SELECT CASE – CASE – CASE ELSE – END SELECT**, общий вид которой:

SELECT CASE <контрольное выражение>
    CASE <список выражений 1>
        <блок операторов 1>
    CASE <список выражений 2>
        <блок операторов 2>
    ⋮
    CASE ELSE
        <блок операторов n>
END SELECT

Здесь:

<контрольное выражение> – любое числовое или строковое выражение (например, имя числовой или символьной переменной);

<список выражений 1>, <список выражений 2> и т.д. – одно или несколько выражений для сравнения с контрольным выражением (например, числовые или символьные значения переменных);

<блок операторов 1>, <блок операторов 2> и т.д. – один или несколько операторов, располагаемых в одной или нескольких строках.

Различные формы использования блочной структуры SELECT CASE-CASE-END SELECT, отличающиеся, главным образом, формой задания аргументов списка выражений рассмотрим на конкретных примерах.

1. Задание списка выражений одним или несколькими значениями переменных, перечисленных через запятую:

```
REM Оптимальное расстояние между шкивами для ременной передачи
CLS
INPUT "Введите диаметр большого шкива ", D
INPUT "Введите передаточное число (=>2)", I
SELECT CASE I
    CASE 2
        A = 1.2 * D
    CASE 3
        A = D
    CASE 4
        A = .95 * D
    CASE 5
        A = .9 * D
    CASE ELSE
        A = .85 * D
END SELECT
PRINT USING "A = ##.### м"; A
```

Здесь контрольным параметром является переменная I (передаточное число ременной передачи), в зависимости от значения которой определяется расчетная формула. QBasic поочередно проверяет, равно ли значение параметра, заданного оператором SELECT CASE числовым значениям в каждом из операторов CASE, и при равенстве значений выполняется блок операторов, следующий за "верным" CASE. Так, при значении передаточного числа

I = 2 расчет будет производиться по формуле A = 1.2*D; при I = 3 – будет принято условие A = D и т.д. Если же ни одно из значений переменной, определяемых операторами CASE (в данном случае 2, 3, 4, 5), не равно заданному параметру, то выполняется блок операторов, следующий после CASE ELSE. Например, в случае задания передаточного числа I > 5, расчет будет произведен по формуле A = 0.85*D.

В рассмотренном примере в каждом из операторов CASE стоит только по одному числовому значению, однако, как указывалось выше после CASE может находиться несколько значений, перечисленных через запятую, например:

```
INPUT K
SELECT CASE K
    CASE 1, 3, 5, 7, 9
        <блок операторов 1>
    CASE 2, 4, 6, 8, 0
        <блок операторов 2>
    CASE ELSE
        <блок операторов 3>
END SELECT
```

В данном случае заданное значение параметра К "в первом" CASE поочередно сравнивается с 1, 3, 5, 7 и 9 и при выполнении любого из условий следует "блок операторов 1", при невыполнении – управление переходит к следующему CASE, в котором значение К поочередно сравнивается с 2, 4, 6, 8 и 0. В случае "невыполнения" ни одного из CASE работает "блок операторов 3".

2. Задание значений переменных в CASE интервалами. В этом случае интервал может определяться как двумя границами (например, 5 < X < 10; при этом для обозначения интервала используется ключевое слово **TO**), так и одной (например, F > 20 или H < 1000; для описания таких условий служит слово **IS**). Рассмотрим такие случаи на примере.

Пример. Определить максимально допустимое водоцементное отношение, которое с целью достижения долговечности бетона необходимо ограничить следующими значениями:

для сурового климата – 0,5; умеренного – 0,53; мягкого – 0,55.

Суждение о климате сделать по среднемесячной температуре наиболее холодного месяца в году:

суровый климат	ниже –15°С
умеренный	от –5°С до –15°С
мягкий	выше –5°С

Дополнительное условие: для районов, где температура не бывает ниже 0°С, водоцементное отношение не должно превышать 0,6 (по условию предохранения поверхности бетона от шелушения).

Часть программы, определяющая максимально допустимое значение

водоцементного отношения, с использованием структуры SELECT CASE, будет выглядеть следующим образом:

```
CLS
INPUT "Введите среднюю температуру наиболее холодного месяца"; T
SELECT CASE T
    CASE IS < -15
        vzmax = .5: REM суровый климат
    CASE -15 TO -5
        vzmax = .53: REM умеренный климат
    CASE -5 TO 0
        vzmax = .55: REM мягкий климат
    CASE ELSE
        INPUT "Введите минимальную температуру, °C"; Tmin
        IF Tmin > 0 THEN vzmax = .6 ELSE vzmax = .55
END SELECT
PRINT USING "Максимальное водоцементное отношение– #.##"; vzmax
```

Здесь по заданной среднемесячной температуре наиболее холодного месяца T определяется блок исполняемых операторов. При "несоблюдении" ни одного из CASE (в данном случае – при положительном значении T) выполняется блок, следующий за CASE ELSE, в котором запрашивается минимальная для района температура, и, уже в зависимости от нее, с помощью строчного IF производится выбор максимально допустимого водоцементного отношения бетона.

При использовании блочной структуры SELECT CASE…END SELECT **необходимо помнить следующее**:

– количество блоков CASE не ограничивается;

– если соблюдается условие, заданное в одном из блоков CASE, то после выполнения операторов данного блока, программа переходит сразу к строке END SELECT, минуя все последующие блоки CASE. В связи с этим могут возникнуть ошибки такого плана. При задании в CASE условий типа IS > 10, IS > 50, IS > 100 и т.п., необходимо следить за тем, чтобы данные условия располагались в порядке убывания значений, например:

```
INPUT V
SELECT CASE V
    CASE IS > 100
        <блок операторов 1>
    CASE IS > 50
        <блок операторов 2>
    CASE IS > 10
        <блок операторов 3>
END SELECT
```

А при наложении условий типа IS < 2, IS < 7, IS < 12 и т.п., наоборот, значения должны располагаться в порядке возрастания, т.е.:

```
INPUT F
SELECT CASE F
    CASE IS < 2
        <блок операторов 1>
    CASE IS < 7
        <блок операторов 2>
    CASE IS < 12
        <блок операторов 3>
END SELECT
```

В силу указанного принципа работы структуры SELECT CASE…END SELECT, обратный порядок расстановки условий может привести к ошибке, поскольку выполнение первого же условия автоматически прекращает проверку последующих.

При использовании блочной структуры выбора SELECT CASE … END SELECT **необходимо также помнить следующее**:

– если используется ключевое слово TO для определения пределов выражения, то меньшее значение должно быть первым. Например, операторы блока CASE -1 TO -5 не выполняются, если значение сравниваемого параметра равно -4. Эта строка должна быть записана как CASE -5 TO -1;

– операции сравнения можно использовать только с ключевым словом IS;

– блок операторов CASE ELSE выполняется только в том случае, если не соблюдается ни одно из условий CASE и обычно используется для обработки нежелательных значений;

– конструкция SELECT CASE…END SELECT допускает отсутствие блока CASE ELSE;

– в каждом условии CASE можно использовать несколько выражений или пределов, например,

```
CASE 1 TO 8, 12 TO 14, 22, 24, IS < Fmax, IS > Fmin  ;
```

– структура SELECT CASE может использоваться и для символьных выражений, при этом строки оцениваются в соответствии с кодами их символов;

– блоки SELECT CASE могут быть вложенными, причем каждый блок должен иметь завершающую строку END SELECT.

Посмотрим на примере организацию вложенных блоков SELECT CASE.

Пример. Рассчитать эксплуатационную производительность бульдозера $P_э$, м^3/ч, определяемую выражением

$$P_э = \frac{3600 \cdot q \cdot k_в \cdot k_у}{T_ц},$$

где q – объем призмы волочения, м^3;

$T_ц$ – время цикла, с;

$k_в$ – коэффициент использования рабочего времени;

k_y – коэффициент, учитывающий влияние уклона местности на производительность, принимаемый равным:

– при работе на подъемах

0 ÷ 5% $k_y = 1{,}0 \div 0{,}67$,
5 ÷ 10% $k_y = 0{,}67 \div 0{,}5$,
10 ÷ 15% $k_y = 0{,}5 \div 0{,}4$;

– при работе на уклонах

0 ÷ 5% $k_y = 1{,}0 \div 1{,}33$,
5 ÷ 10% $k_y = 1{,}33 \div 1{,}94$,
10 ÷ 15% $k_y = 1{,}94 \div 2{,}25$,
15 ÷ 20% $k_y = 2{,}25 \div 2{,}68$.

Здесь для корректной работы программы и ее удобного использования необходимо задать характер местности (подъем, уклон, равнина) и его величину в %. В зависимости от этих параметров будет изменяться значение коэффициента k_y, причем необходимо предусмотреть нахождение k_y не только для граничных, но и для любых промежуточных значений уклона.

Реализовать решение этой задачи в QBasic позволяет вложенная блочная структура SELECT CASE. Внешний оператор позволит задать характер местности, а внутренний – рассчитать заданный коэффициент k_y в зависимости от величины уклона. Расчет k_y можно осуществить методом интерполяции, считая изменение коэффициента линейным внутри каждого интервала.

Программа на QB будет иметь вид:

```
REM Расчет эксплуатационной производительности бульдозера
CLS
KV = .8
INPUT "Введите объем призмы волочения, м куб"; Q
INPUT "Введите продолжительность цикла работ, с"; T
PRINT "Введите код, соответствующий характеру работы:"
PRINT "1 – при работе на подъемах"
PRINT "2 – при работе на уклонах"
INPUT "3 – при работе на равнинной местности"; UKL
SELECT CASE UKL
   CASE 1
       REM работа на подъемах
       INPUT "Введите величину уклона, %"; I
       SELECT CASE I
          CASE IS < 5
               KU = 1 – .066 * I
          CASE IS < 10
               KU = .67 – .034 * (I – 5)
          CASE IS <= 15
               KU = .5 – .02 * (I – 10)
          CASE ELSE
               PRINT "Задана очень большая величина уклона"
               END
       END SELECT
```

```
    CASE 2
        REM работа на уклонах
        INPUT "Введите величину уклона, %"; I
        SELECT CASE I
           CASE IS > 20
                PRINT "Задана очень большая величина уклона"
                END
           CASE IS > 15
                KU = 2.25 + .086 * (I - 15)
           CASE IS > 10
                KU = 1.94 + .062 * (I - 10)
           CASE IS > 5
                KU = 1.33 + .122 * (I - 5)
           CASE ELSE
                KU = 1 + .066 * I
        END SELECT
    CASE 3
        REM работа на равнинной местности
        KU = 1
    CASE ELSE
        PRINT "Введен несуществующий код"
        END
END SELECT
P = 3600 * Q * KV * KU / T
PRINT USING "Коэффициент уклона KU = #.###"; KU
PRINT USING "Производительность бульдозера P = ###.# м куб./ч"; P
```

4.4. Логические операции, их использование в операторах условного перехода. Таблица истинности.

Логические ("Булевы") операции осуществляют манипуляции над битами. Результатом логической операции могут быть лишь два значения: "TRUE" (ИСТИНА – ненулевое значение, внутреннее представление QBasic = -1) или "FALSE" (ЛОЖЬ – нулевое значение, внутреннее представление QBasic = 0)

В QBasic существует шесть логических операторов, название и толкование которых приведены в табл. 6.

Таблица 6

Логические операторы в QBasic

Оператор	Название	Объяснение
NOT	Отрицание	NOT A истинно тогда и только тогда, когда A ложно
AND	Логическое умножение	A AND B истинно тогда и только тогда, когда истинно A и истинно B
OR	Логическое сложение	A OR B истинно тогда и только тогда, когда хотя бы одно из A и B истинно
XOR	Исключающие ИЛИ	A XOR B истинно тогда и только тогда, когда значения A и B не совпадают
EQV	Эквивалентность	A EQV B истинно только тогда, когда A и B одновременно истинны или одновременно ложны
IMP	Импликация	A IMP B принимает значение ложь, если A истинно, а B ложно, и значение "истина" в других случаях

Данные операторы могут быть использованы в сочетании с оператором IF при необходимости одновременного наложения двух или более условий. Поясним сказанное на примере.

Пример. Рассчитанное значение тока в электровзрывной цепи I, А, проверить по условию $I \geq I_{гар}$,

где $I_{гар}$ – минимальный гарантийный ток, значение которого принимается равным:

$I_{гар}$ = 1 А – при одновременном взрывании от постоянного тока до 100 электродетонаторов (ЭД);

$I_{гар}$ = 1,3 А – то же до 300 ЭД;

$I_{гар}$ = 2,5 А – при взрывании переменным током.

Здесь гарантийный минимальный ток определяется в зависимости от двух условий (типа тока и количества взрываемых ЭД)

Такую условную структуру можно организовать, используя в сочетании с оператором IF, операцию логического умножения. В этом случае блок операторов, организующий проверку условия, будет выглядеть следующим образом:

```
PRINT "Введите вид тока:"
PRINT "1 – постоянный,"
INPUT "2 – переменный"; TT
INPUT "Введите количество одновременно взрываемых ЭД"; N
IF TT = 1 AND N < 100 THEN IG = 1
IF TT = 1 AND N > 100 THEN IG = 1.3 ELSE IG = 2.5
```

4.5. Лабораторная работа №2

Тема. Программирование алгоритмов разветвляющейся структуры

Цель работы. Научиться составлять алгоритмы разветвляющейся структуры и программы расчета параметров по заданным формулам при возможности выполнения различных ветвей программы в зависимости от заданных значений исходных данных.

Задание. Согласно варианту задания составить блок-схему алгоритма и программу на языке QBasic по расчету указанных параметров. Программа должна предусматривать возможность расчета всех указанных вариантов. При необходимости одновременного наложения двух или более условий использовать либо структуру строчного IF в сочетании с логическими операторами, либо структуру блочного IF или же организовать структуру выбора SELECT CASE. Форму ввода исходных данных и вывода на печать результатов выполнения программы выбрать самостоятельно.

4.5.1. Варианты лабораторной работы

Вариант 1

Проверить сечение вертикального ствола на максимально допустимую скорость движения воздуха, если скорость движения воздуха по стволу определяется формулой

$$v = \frac{Q}{60 \cdot S \cdot \mu} \le \langle V \rangle,$$

где Q – количество воздуха, необходимое для проветривания шахты, м^3/мин,

$$Q = \frac{100 \cdot q \cdot A}{24 \cdot 60 \cdot d},$$

q – относительная газообильность шахты, м^3/т;

A – производственная мощность шахты, т/сут;

d – допустимая концентрация газа в исходящей струе, d = 0,75 %.

S – площадь поперечного сечения ствола в свету, м^2, $S = \frac{\pi \cdot D_{св}^2}{4}$;

$D_{св}$ – диаметр ствола в свету, м;

μ – коэффициент армировки, μ = 0,8;

$\langle V \rangle$ – максимально допустимая скорость движения вентиляционной струи, (для скиповых стволов принять $\langle V \rangle$ = 12 м/с, для клетевых – $\langle V \rangle$ = 8 м/с).

Исходные данные (тип подъема: клетевой или скиповой, q, A, $D_{св}$) – задать самостоятельно. Предусмотреть возможность изменения диаметра ствола в случае превышения допустимой скорости воздуха. На печать вывести сообщение о соответствии принятого сечения ствола условиям вентиляции.

Вариант 2

Определить возможную массу одновременно поднимаемого полезного груза, т, используя эмпирическую зависимость

$$Q = \frac{4 \cdot \sqrt{H} + \theta}{3600} \cdot A_ч,$$

где H – высота подъема, м; $H = h_{ст} + h_з + h_п$,

$h_{ст}$ – глубина ствола от устья ствола до уровня почвы околоствольного двора, м;

$h_з$ – высота загрузки скипа у подъемного бункера, м (для скиповых подъемов $h_з$ = 15 м; для клетевых $h_з$ = 0);

$h_п$ – высота приемной площадки, м (для скиповых подъемов $h_п$ = 20 м; для клетевых $h_п$ = 8 м);

θ – время, затрачиваемое на разгрузку и загрузку подъемных сосудов, с, *для скиповых подъемов*: при вместимости скипа до 5 м^3 принять θ = 7 с,

при вместимости скипа больше 7 м^3 число секунд на разгрузку и загрузку равно числу вместимости скипа в кубических метрах;

для клетевых подъемов: при вагонетке вместимостью 1,3 м^3 – $\theta = 12$ с; 2,5 ÷ 4 м^3 – $\theta = 15$ с; 5,6 ÷ 8 м^3 – $\theta = 20$ с.

$A_ч$ – производительность подъемной установки, т/ч.

Исходные данные (тип подъема: клетевой или скиповой, $h_{ст}$, $A_ч$, вместимость вагонетки или скипа) – задать самостоятельно.

Вариант 3

Рассчитать линию наименьшего сопротивления, м, по формуле

$$W = \frac{k}{\sqrt{f}},$$

где f – коэффициент крепости пород;

k – эмпирический коэффициент, принимаемый равным:

– для патронов скального аммонита № 1

диаметром 36 мм – $k = 2{,}3$,

диаметром 45 мм – $k = 2{,}45$;

– для патронов аммонита Т-19, ПЖВ-20, АП-5ЖВ – $k = 1{,}48$.

Исходные данные (диаметр патронов и тип ВВ, крепость взрываемых пород) задать самостоятельно.

Вариант 4

Рассчитать производительность бурения шпуров в стволе ручными перфораторами, м/ч, по формуле

$$Q_б = \frac{50 \cdot \varphi \cdot n_п \cdot k_д \cdot k_п \cdot k_в}{4{,}5 + f},$$

где φ – коэффициент одновременности работы перфораторов, $\varphi = 0{,}8 \div 0{,}9$;

$n_п$ – число перфораторов в забое, определяемое из соотношения

$$n_п = \frac{S_{пр}}{S_{уд}};$$

$S_{пр}$ – площадь поперечного сечения ствола в проходке, м^2;

$S_{уд}$ – площадь забоя на один перфоратор (на скоростных проходках $S_{уд} = 1{,}5 \div 2$ м^2);

$k_д$ – коэффициент, учитывающий диаметр шпура, $k_д = \frac{36}{d_ш}$,

$d_ш$ – диаметр шпура, мм;

$k_п$ – коэффициент, учитывающий тип перфоратора и равный 1,2 – для перфоратора ПП-24ЛС и 1 – для перфораторов ПП-30ЛС, ПП-30ЛБ;

$k_в$ – коэффициент, учитывающий приток воды в ствол и принимаемый согласно следующим данным в зависимости от величины водопритока:

Приток воды в ствол, м^3/ч	до 6	6-13	13-20
Коэффициент $k_в$	1	0,9	0,8

Предусмотреть ручной ввод следующих исходных данных: площади поперечного сечения ствола в проходке, крепости вмещающих пород, величины водопритока в ствол и типа применяемых перфораторов.

Вариант 5

Рассчитать общую продолжительность бурения шпуров бурильной установкой типа БУКС по формуле

$$T_б = \frac{N \cdot l_{шп}}{Q_б},$$

где N – число шпуров;

$l_{шп}$ – длина шпуров, м;

$Q_б$ – производительность бурения шпуров, м/мин, $Q_б = \dfrac{\varphi \cdot n \cdot k_п \cdot v_м}{1 + v_м \Sigma t_в}$,

φ – коэффициент одновременности работы бурильных машин,

φ = 0,7÷0,8;

n – число бурильных машин в установке, n = 4;

$k_п$ – коэффициент готовности установки, $k_п$ = 0,8÷0,9;

$\Sigma t_в$ – время вспомогательных работ на бурение 1 м шпура – замена коронок, обратный ход штанги, перестановка установки в забое и т.д;

при длине шпура 3÷4 м время $\Sigma t_в$ составляет:

– для крепости пород f < 10, $\Sigma t_в$ = 1,5 мин,

– для f > 10, $\Sigma t_в$ = 2 мин;

при длине шпура 2÷3 м:

– для крепости пород f < 10, $\Sigma t_в$ = 1 мин,

– для f > 10, $\Sigma t_в$ = 1,5 мин.

$v_м$ – механическая скорость бурения, м/мин, принимаемая согласно следующим данным в зависимости от крепости пород:

f	до 6	7 – 9	10 – 14	свыше 14
$v_м$, м/мин	1,1	0,8	0,6	0,25

Предусмотреть ручной ввод следующих исходных данных: числа и длины шпуров и крепости вмещающих пород. На печать вывести значение производительности бурения и общего времени работ, выраженного в часах и минутах.

Вариант 6

Рассчитать производительность грейферной погрузочной машины, м^3/ч, в первой фазе погрузки, используя выражение

$$P_{погр} = \frac{3600}{\tau_ц} \cdot q_{гр} \cdot k_{гр},$$

где $\tau_ц$ – продолжительность цикла черпания, принимаемая при погрузке сланцев равной 25 с и при погрузке песчаников 35 с;

$k_{гр}$ – коэффициент наполнения грейфера (для сланцев $k_{гр} = 1$, для песчаников $k_{гр} = 0{,}9$).

$q_{гр}$ – вместимость грейфера, м^3, принимаемая согласно следующим данным в зависимости от типа погрузочной машины:

Тип машины	КС-3	КС-2у / 40	КС-1м	2КС-1м
Объем грейфера, м^3	0,22	0,65	1	1

Предусмотреть возможность ручного ввода типа пород (песчаники или сланцы) и типа погрузочной машины. На печать вывести тип машины и величину ее производительности.

Вариант 7

Рассчитать продолжительность первой фазы погрузки породы, ч, по выражению:

$$T_{погр}^{1ф} = \frac{k_р \cdot S \cdot (\eta \cdot l_{шп} - h_{2ф})}{P_{погр}^{\max} \cdot k_{1ф}^{ср}},$$

где $k_р$ – коэффициент разрыхления породы ($k_р$ =1,8 ÷ 2);

S – площадь поперечного сечения ствола в проходке, м^2;

η – коэффициент использования шпура;

$l_{шп}$ – длина шпура, м;

$h_{2ф}$ – высота слоя породы во второй фазе погрузки, м;

$P_{погр}^{\max}$ – максимальная производительность погрузки породы, м^3/ч;

$k_{1ф}^{ср}$ – коэффициент средней производительности погрузочно-подъемного оборудования в первой фазе, принимаемый равным:

при погрузке без перецепки бадей $k_{1ф}^{ср} = 0{,}9$;

при погрузке с перецепкой бадей $k_{1ф}^{ср} = 0{,}8$.

Значения параметров $h_{2ф}$ и $P_{погр}^{\max}$ задать согласно нижеприведенным данным в зависимости от типа погрузочной машины:

Тип машины	КС-3	КС-2у / 40	КС-1м	2КС-1м
$h_{2ф}$, м	0,15	0,25	0,4	0,4
$P_{погр}^{\max}$, м^3/ч	20	75	130	200

Предусмотреть возможность ручного ввода площади поперечного се-

чения ствола в проходке, длины шпуров, коэффициента использования шпура, типа погрузочной машины и схемы подъема бадей (с перецепкой или без нее). На печать вывести тип принятой машины и величину ее производительности и продолжительность первой фазы погрузки породы, выраженную в часах и минутах.

Вариант 8

Рассчитать продолжительность второй фазы погрузки породы, ч, по формуле:

$$T_{погр}^{2ф} = \frac{k_p \cdot S \cdot h_{2ф}}{N_{уб} \cdot P'_{2ф}},$$

где k_p – коэффициент разрыхления породы (k_p =1,8 ÷ 2);

S – площадь поперечного сечения ствола в проходке, м^2;

$h_{2ф}$ – высота слоя породы во второй фазе погрузки, м;

$N_{уб}$ – число проходчиков, занятых непосредственно в забое на погрузке породы во второй фазе;

$P'_{2ф}$ – производительность погрузки во второй фазе, м^3/ч.

Принять $h_{2ф}$ = 0,25 м, $N_{уб}$ = 6 чел. Значение $P'_{2ф}$ задать согласно следующим данным в зависимости от коэффициента крепости пород и способа зачистки забоя:

Коэффициент крепости породы	Способ зачистки забоя	$P'_{2ф}$, м^3/ч
f = 3 ÷ 6	Пневмомонитором	2,6
	Вручную	1,6
f = 7 ÷ 10	Пневмомонитором	1,9
	Вручную	1,2
f = 12 ÷ 16	Пневмомонитором	1,3
	Вручную	0,9

Предусмотреть возможность ручного ввода площади поперечного сечения ствола в проходке, коэффициента крепости пород и способа зачистки забоя ствола (пневмомонитором или вручную). На печать вывести продолжительность второй фазы погрузки породы, выраженную в часах и минутах.

Вариант 9

Определить длину шпуров $l_{шп}$, м, исходя из заданных темпов проходки выработки, используя выражение

$$l_{шп} = \frac{V \cdot T_ц}{k_г \cdot m \cdot n \cdot \eta \cdot t_{см}},$$

где V – заданная скорость строительства выработки, м/мес.;

$T_ц$ – время цикла, ч;

$k_г$ – коэффициент готовности технологической схемы, принимаемый равным:

– при строительстве вертикальных выработок:

для последовательной и параллельной схем $k_г = 0,7$,

для совмещенной схемы $k_г = 0,8$,

для параллельно-щитовой схемы $k_г = 0,85$;

– при строительстве горизонтальных выработок $k_г = 1$.

m – число рабочих дней в месяц;

n – число рабочих смен в сутки;

η – коэффициент использования шпура, принимаемый равным:

для забоев вертикальных выработок $\eta = 0,8$;

для забоев горизонтальных и наклонных выработок, проходимых

– по породе $\eta = 0,85$,

– по пласту с присечкой боковых пород $\eta = 0,9$,

– по угольному пласту $\eta = 0,95$;

$t_{см}$ – продолжительность рабочей смены, ч.

Организовать ручной ввод следующих данных: расположения выработки в пространстве (вертикальная, наклонная, горизонтальная), типа пород, пересекаемых выработкой (уголь, пустые породы, смешанный забой); для вертикальных выработок предусмотреть ввод технологической схемы строительства (последовательная, параллельная, совмещенная, параллельно-щитовая). Параметры V, $T_ц$, m, n, $t_{см}$ задать самостоятельно. На печать вывести коэффициент готовности технологической схемы, коэффициент использования шпура и расчетную длину шпура.

Вариант 10

1. Определить возможность бадьевого водоотлива из забоя ствола исходя из следующего условия:

$$q \le 0,8 \cdot D_{пр}^2 \cdot \frac{0,4 \cdot l_з - 0,1}{t},$$

где q – водоприток в ствол, м^3/ч;

$D_{пр}$– диаметр ствола в проходке, м;

$l_з$ – глубина заходки, м;

t – продолжительность простоя водоотлива в связи со взрывными работами и проветриванием, ч.

2. Рассчитать величину максимального водопритока в ствол, м^3/ч, который может быть откачана бадьями, по формуле

$$W = 0,9 \cdot n \cdot V_б \cdot k,$$

где n – число подъемов в час;

$V_б$ – вместимость бадьи, м^3;

k – коэффициент, характеризующий объем полостей в разрыхленной породе, принимаемый согласно следующим данным в зависимости от крепости пород f:

f	Менее 4	4÷6	7÷9	Более 9
k	0,3	0,4	0,5	0,6

Расчет выполнить для следующих исходных данных: t = 1,5 ч; n = 15; $l_з$ = 4 м. Предусмотреть ручной ввод параметров q, $D_{пр}$, $V_б$, f, значения которых задать самостоятельно. На печать вывести значение максимального водопритока и сообщение о возможности (или невозможности) применения бадьевого водоотлива в заданных условиях.

Вариант 11

Рассчитать общую продолжительность процесса крепления заходки монолитным бетоном, ч, при сооружении вертикального ствола:

$$T_{кр} = \frac{\pi \cdot \delta \cdot (2 \cdot D_{св} - \delta)}{4 \cdot q_{бет}} \cdot h_{оп} + t_{кр}^{п.з}$$

где $D_{св}$ – диаметр ствола в свету, м;

δ – толщина крепи, м;

$q_{бет}$ – суммарная производительность подачи бетона, м^3/ч, принимаемая равной 7 м^3/ч при подаче бетона по одному бетонопроводу и 14 м^3/ч – по двум бетонопроводам. Число бетонопроводов принимается в зависимости от диаметра ствола: при диаметре ствола более 6 м – 2 бетонопровода, иначе – 1;

$h_{оп}$ – рабочая высота опалубки, м;

$t_{кр}^{п.з}$ – продолжительность подготовительно-заключительных операций, ч,

$$t_{кр}^{п.з.} = t_1 + t_2 + t_3 ,$$

t_1 – время разравнивания породы в забое для установки опалубки (t_1 = 1 ч);

t_2 – время на раскрытие створок, отрыв, спуск и центровку опалубки, включающее подготовку и заключительные операции (t_2 = 2 ч);

t_3 – время выдержки бетона при совмещении крепления с погрузкой (при совмещенной схеме проходки t_3 = 1 ч, при последовательной t_3 = 0).

Расчет произвести для $h_{оп}$ = 4 м. Предусмотреть ручной ввод параметров $D_{св}$, δ и схемы проходки ствола (последовательная или совмещенная). На печать вывести принятое количество бетонопроводов и продолжительность процесса крепления, выраженную в часах и минутах.

Вариант 12

Рассчитать коэффициент утечек воздуха $K_{ут.тр}$ в жестком вентиляционном трубопроводе по формуле

$$K_{ут.тр} = \left(\frac{1}{3} \cdot K_{ут.ст} \cdot d_{тр} \cdot \frac{l_{тр}}{l_{зв}} \cdot \sqrt{R_{тр}} + 1 \right)^2 ,$$

где $K_{ут.ст}$ – коэффициент удельной стыковой воздухопроницаемости трубопровода, $K_{ут.ст}$ = 0,002;

$d_{тр}$ – диаметр вентиляционного трубопровода, м;

$l_{тр}$ – полная длина трубопровода, м, $l_{тр} = l_c + l_{ву}$;

l_c – глубина ствола, м;

$l_{ву}$ – расстояние от устья ствола до вентилятора, м;

$l_{зв}$ – длина звена трубопровода, м;

$R_{тр}$ – расчетное аэродинамическое сопротивление трубопровода, кμ, определяемое из выражения $R_{тр} = \dfrac{6{,}5 \cdot \alpha \cdot l_{тр}}{d_{тр}^5}$,

α – коэффициент аэродинамического сопротивления трубопровода, Н·с^2/м^4.

Значения параметров и принять в зависимости от диаметра трубопровода согласно следующим данным:

$d_{тр}$, м	0,5	0,6	0,7	0,8	0,9	1
$l_{зв}$, м	3	3	3	3	4	4
α, Н·с^2/м^4	0,00035	0,00032	0,0003	0,00025	0,00024	0,00023

Организовать ручной ввод параметров l_c, $l_{ву}$, $d_{тр}$, значения которых задать самостоятельно. На печать вывести значения $l_{зв}$, α, $R_{тр}$, $K_{ут.ст}$.

Вариант 13

Согласно приведенным данным, по заданному коэффициенту крепости пород f и площади поперечного сечения выработки S, м^2, определить оптимальное число врубовых шпуров.

Коэффициент крепости пород f	Оптимальное число врубовых шпуров при площади сечения выработки	
	S < 12 м^2	S ≥ 12 м^2
2÷3	4	5
4÷6	5	7
7÷9	7	9
10÷20	10	13

Предусмотреть ручной ввод параметров f и S. На печать вывести коэффициент крепости, площадь поперечного сечения выработки и принятое число врубовых шпуров.

Вариант 14

Согласно приведенным данным, по коэффициенту крепости пород f и диаметру патрона ВВ d, мм, определить коэффициент заполнения шпуров.

Диаметр патрона ВВ, мм	Коэффициент заполнения шпуров при	
	$f = 2 \div 9$	$f = 10 \div 20$
28	0,75	0,8
32	0,6	0,65
36	0,5	0,6
45	0,45	0,55

Предусмотреть ручной ввод параметров f и d. На печать вывести коэффициент крепости, диаметр патрона ВВ и принятый коэффициент заполнения шпуров.

Вариант 15

Рассчитать силу тока I, А, проходящего через каждый ЭД в последовательно соединенной электровзрывной сети, и проверить соблюдение условия

$$I \geq I_{гар},$$

где $I_{гар}$ – минимальный гарантийный ток, А, согласно ЕПБ принимаемый:

– при одновременном взрывании до 100 ЭД $I_{гар} = 1$ А;

– при одновременном взрывании до 300 ЭД $I_{гар} = 1{,}3$ А;

– при взрывании переменным током $I_{гар} = 2{,}5$ А.

Ток в каждом ЭД определить по закону Ома для участка цепи:

$$I = \frac{U}{R},$$

где U – напряжение источника тока, В;

R – общее сопротивление электровзрывной сети, для последовательного соединения ЭД определяемое формулой

$R = m \cdot r_д + L_к \cdot r_к + L_у \cdot r_у + L_м \cdot r_м$,

m – число ЭД;

$r_д$– сопротивление одного ЭД, Ом;

$L_к$, $L_у$, $L_м$ – длина соответственно концевых, участковых и магистральных проводов, м;

$r_к$, $r_у$, $r_м$ – сопротивление 1 м соответственно концевых, участковых и магистральных проводов, Ом/м.

Расчет произвести для следующих исходных данных: $r_д$ = 4,2 Ом; $L_к = 4 \cdot m$; $L_у = 20$ м; $L_м = 400$ м; $r_к = 0{,}085$ Ом/м; $r_у = 0{,}04$ Ом/м; $r_м = 0{,}023$ Ом/м.

Организовать ручной ввод следующих параметров: m, U, вида тока (переменный или постоянный). На печать вывести значения сопротивления электровзрывной сети, тока в каждом ЭД и сообщение об обеспечении (необеспечении) минимального гарантийного тока. В случае несоблюдения минимального гарантийного тока предусмотреть возможность изменения напряжения источника тока и повторный расчет.

4.5.2. Пример выполнения лабораторной работы

Задание. Определить предел прочности цементного раствора при сжатии в 28-суточном возрасте R_{28}, МПа, используя формулу

$$R_{28} = K \cdot R_ц \cdot (Ц - 0{,}05) + 4 ,$$

где $Ц$ – расход цемента, т / м^3 песка;

$R_ц$ – активность цемента, МПа;

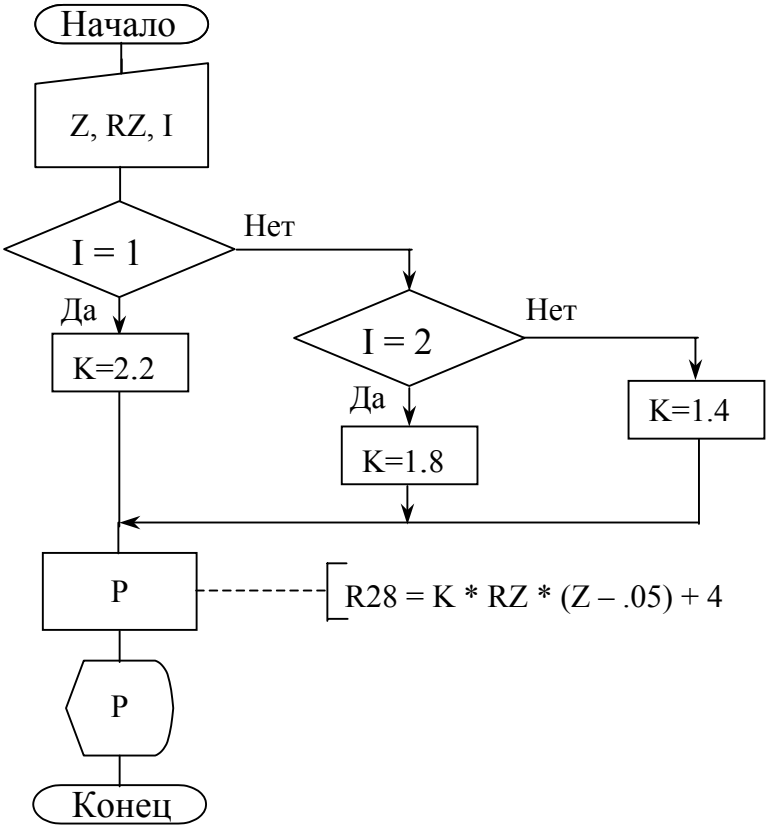
K – коэффициент, зависящий от качества песка, К = 2,2 – для крупного песка; 1,8 – для песка средней крупности; 1,4 – для мелкого песка.

Предусмотреть ручной ввод параметров Ц, $R_ц$, значения которых задать самостоятельно, а также качества песка (крупный, средней крупности или мелкий) .

Присваиваем имена переменным и составляем таблицу идентификаторов:

Переменная	Идентификатор	Примечание
R_{28}	R28	Рассчитываемый параметр
Ц	Z	Параметр, задаваемый вручную
$R_ц$	RZ	Параметр, задаваемый вручную
I	I	Код, соответствующий качеству песка
K	K	Коэффициент, определяемый программой в зависимости от параметра I

Составляем блок-схему алгоритма:



Программа на языке QBasic.

```
REM Расчет предела прочности бетона на сжатие
CLS
INPUT "Введите расход цемента, т/м куб. песка"; Z
INPUT "Введите активность цемента, МПа"; RZ
PRINT "Введите код, соответствующий качеству песка:"
PRINT "1 – для крупного"
PRINT "2 – для средней крупности"
INPUT "3 – для мелкого"; I
IF I = 1 THEN
     K = 2.2
  ELSEIF I = 2 THEN
     K = 1.8
  ELSEIF I = 3 THEN
     K = 1.4
  ELSE PRINT "Введен несуществующий код"
END IF
R28 = K * RZ * (Z – .05) + 4
PRINT USING "Прочность раствора R28 = ###.## МПа "; R28;
PRINT USING "при коэффициенте К= #.#"; K
```

Результат выполнения программы

при Z = 0,4; RZ = 15 и I = 2 (песок средней крупности):

Прочность раствора R28 = 13.45 МПа при коэффициенте К= 1.8

5. ПРОГРАММИРОВАНИЕ АЛГОРИТМОВ ЦИКЛИЧЕСКОЙ СТРУКТУРЫ

Алгоритмом циклической структуры будем считать алгоритм, имеющий один или более участков, выполнение которых повторяется многократно. Чаще всего циклический алгоритм организуется при необходимости неоднократного выполнения некоторых действий (расчете значений функции при различных значениях аргумента, реализации ряда численных методов, работе с массивами данных, построении графиков функций и графических изображений, выводе на печать таблиц и многом другом) при изменении внутри цикла одной или нескольких переменных. Операции, включенные в цикл, могут повторяться либо заданное число раз, либо неопределенное число раз до выполнения (невыполнения) некоторого условия. Конкретные способы и формы организации циклических структур будут рассмотрены нами ниже.

5.1. Массивы. Имя и описание массива

При работе с рядами однотипных числовых или символьных переменных, матрицами, таблицами целесообразно применять не простые (п. 2.3.2), а *индексированные* переменные, являющиеся элементами *одно–* , *двух* или *многомерных массивов*.

Массив – упорядоченная совокупность однотипных данных, с каждым из которых связан набор индексов. Массив предназначен для хранения в программе набора значений числовых или символьных переменных.

Любой массив характеризуется *именем*, *размерностью* и *размером*. Имя массива образуется по общему правилу присвоения имени обычной переменной (подробно об этом было изложено в п. 2.3.1), т.е. представляет собой идентификатор. Размерность определяет форму массива, а размер характеризует его объем. Так, *одномерный* массив представляет собой одну строку (столбец) переменных, т.е. последовательность вида

$$a_0 \;\; a_1 \;\; a_2 \;\; a_3 \;\; \ldots a_{n-1} \;\; a_n$$

Размер такого массива соответствует числу элементов, т. е. равен n.

Двумерные массивы задают матрицу чисел, например:

$$\begin{array}{ccccc} b_{11} & b_{12} & b_{13} & \ldots & b_{1n} \\ b_{21} & b_{22} & b_{23} & \ldots & b_{2n} \\ \ldots & \ldots & \ldots & \ldots & \ldots \\ b_{m1} & b_{m2} & b_{m3} & \ldots & b_{mn} \end{array}$$

Такой массив характеризуется числом строк m и числом столбцов n. Его размер можно выразить как $m \times n$.

Размерность массива может быть и большей. Так, например, трехмерный массив можно представить как p матриц размера $m \times n$. Таким образом, трехмерный массив имеет размер $m \times n \times p$. Однако в дальнейшем мы будем использовать только одномерные и двумерные массивы, наиболее широко встречающиеся в практике программирования.

Как видно из двух вышеприведенных примеров, индексы (т.е. числа, указывающие номер или местоположение переменных в массиве) могут начинаться как с 0 (например, a_0), так и с 1 (например, b_{11}). При отсутствии дополнительных команд (по умолчанию) индексация элементов массива в QB начинается с 0. Однако это не всегда удобно. Для того, чтобы установить нижнюю границу индекса массива, используют оператор **OPTION BASE**, в котором указывается 0 или 1. Например, после строки

```
OPTION BASE 1
```

все встречающиеся в программе элементы массивов по умолчанию будут иметь минимальный индекс, равный 1.

Для резервирования места в памяти ЭВМ с целью помещения туда численных значений элементов массива необходимо произвести описание каждого массива, используемого в программе. Для этого в QB служит оператор **DIM**, который задает имя, размерность, размер и, при необходимости, тип массива. Например, оператор

```
DIM RF (5, 15)
```

описывает двумерный массив (на это указывает количество индексов в скобках) с именем RF, максимальные индексы которого – 5 (по строкам) и 15 (по столбцам). Кроме указанных параметров оператор DIM позволяет описать тип данных массива (**INTEGER**, **LONG**, **SINGLE**, **DOUBLE**, **STRING**), подробнее о типах данных – см. п. 2.3.2. Например, оператор

```
DIM A(10) AS LONG
```

описывает одномерный массив длинных целых, пронумерованных от 0 до 10, а команды

```
OPTION BASE 1
DIM PR26(6,9)
```

описывают массив с именем PR26, имеющий шесть строк и девять столбцов, причем индексы массива принимают значения, начиная с единицы.

При необходимости задания любых минимальных и максимальных значений индексов (в том числе и отрицательных) переменных массива можно использовать оператор DIM с ключевым словом **TO**. Например, оператор

```
DIM SUN(-1 TO 1, 5 TO 20)
```

описывает двумерный массив SUN, пронумерованный по строкам от -1 до 1, по столбцам от 5 до 20.

Один оператор DIM позволяет описать несколько, причем разнотипных, массивов, например:

```
DIM A(5,2), PROV$(10,4), QUER(0 TO 6, 10 TO 20)
```

При описании и использовании массивов **необходимо также помнить следующее**:

– после оператора OPTION BASE может стоять либо 0, либо 1; любые другие значения будут считаться ошибкой;

– оператор OPTION BASE должен стоять в начале программы до использования массивов. Попытка изменить значение OPTION BASE в ходе программы или задать его после описания массивов приведет к ошибке "Array already dimensioned" (массив уже определен);

– несоответствие используемых индексов значениям, заданным оператором DIM, или попытка применить нулевой индекс для переменной массива при наличии в начале программы оператора OPTION BASE 1 приведет к ошибке "Subscript out of range" (индекс вне диапазона);

– если параметры одномерного массива не будут объявлены до его использования, то при запуске программы размер этого массива по умолчанию принимается равным 10 элементам. Поэтому использование одномерного массива без описания допускается только тогда, когда количество его элементов не превышает 10, в противном случае возникает ошибка;

– использование двумерных и многомерных массивов без описания их оператором DIM допускается также только в тех случаях, когда ни один из индексов не превышает 10. И все же во избежание ошибок рекомендуем всегда описывать массивы оператором DIM;

– попытка описания двух массивов с одинаковым именем и типом, но разными параметрами (размером, размерностью) приведет к ошибке "Array already dimensioned";

– однако при указании различных типов массивы с одинаковыми именами будут восприниматься как разные; например, допустимо задавать в одной программе массивы a(5,5), a%(5,5), a&(5,5), a#(5,5), a$(5,5), но недопустимо a(5,5) и a!(5,5), поскольку любой массив по умолчанию уже считается состоящим из действительных чисел одинарной точности;

– при попытке произвести действия с элементом массива, количество индексов которого не совпадает с размерностью массива, описанного под данным именем в DIM, возникает ошибка "Wrong number of dimension" (неверное число определения);

– при задании массива необходимо учитывать следующие ограничения QBasic:

максимальный размер массива – 65635 байт (64 К);

максимальная размерность – 8;

максимальный номер индекса – 32768.

Действия с массивом, т.е. набором данных, объединенным одним

именем, более удобны, чем аналогичные действия с разрозненными данными, имеющими разные имена, поскольку в первом случае можно использовать циклические структуры языка, позволяющие при компактной записи текста программы осуществлять большое количество однотипных операций сразу для всех элементов массива.

Рассмотрим различные способы организации циклов в QBasic.

5.2. Организация циклов в QBasic

Цикл – участок программы, который выполняется несколько раз (возможно при других значениях входящих в него переменных) в процессе работы программы.

5.2.1. Циклы с заданным числом повторений (оператор FOR – TO – STEP – NEXT)

Если количество повторений операций известно заранее, то целесообразно использовать простейшую циклическую структуру QBasic **FOR – TO - STEP – NEXT**.

Общий вид такой структуры выглядит следующим образом:

```
FOR J = α TO β STEP γ
.
.  < тело цикла>
.
NEXT J
```

Здесь J – управляющая переменная (ее имя образуется по тем же правилам, что и имя обычной переменной);

α – числовое значение или выражение, задающее начальное значение управляющей переменной при выполнении цикла;

β – числовое значение или выражение, определяющее последнее значение управляющей переменной при выполнении цикла;

γ – числовое значение или выражение, определяющее шаг, с которым управляющая переменная изменяется от своего начального значения к последнему.

Рассмотрим порядок работ такой структуры на конкретном примере.

Найти усилие F, Н, в каждой ветви канатной стропы подъемного крана (для симметричной подвески груза) при изменении угла наклона стропы к вертикали α от 0 до 60° с шагом 10°, используя выражение:

$$F = \frac{M_г \cdot g}{n \cdot \cos\alpha},$$

где $M_г$ – масса поднимаемого груза, кг;

n – количество ветвей;

g – ускорение свободного падения, м/с^2,

и вывести на печать значения α и соответствующие им значения F.

Здесь заранее заданы начальное $x_н$, конечное значения $x_к$ и шаг изменения h управляющей переменной, т.е. известно число повторений цикла N ($N = \frac{x_к - x_н}{h} + 1$), следовательно, целесообразно использовать циклическую структуру FOR-TO-STEP-NEXT.

Программа в данном случае будет выглядеть так:

```
REM Расчет усилия в ветви стропы
CLS
PI = 3.1415926# : g = 9.81
INPUT "Введите массу поднимаемого груза, кг"; m
INPUT "Введите количество ветвей стропы"; n
PRINT "alfa,°        F, H"
FOR alfa = 0 TO PI / 3 STEP PI / 36
    F = m * g / (n * COS(alfa))
    PRINT USING "  ##       #####.#"; alfa * 180 / PI; F
NEXT alfa
```

После присвоения переменным PI и g соответствующих значений констант и ручного ввода параметров m и n начинается выполнение цикла FOR-TO-STEP-NEXT. Данный цикл работает следующим образом:

– вычисляются конечное значение (PI/3) и шаг изменения alfa (PI/36);

– проверяется возможность выполнения цикла (для выполнения цикла необходимо, чтобы конечное значение управляющей переменной alfa было больше начального при положительном шаге ее изменения, и наоборот – при отрицательном шаге);

– при соблюдении предыдущего условия выполняются строки тела цикла, т.е. строки, заключенные между операторами FOR и NEXT, таким образом определяется значение F при начальном alfa и выводятся на печать начальное alfa и полученное значение F;

– затем значение управляющей переменной изменяется на величину шага (увеличивается, если шаг положительный, или уменьшается, если шаг отрицательный), после чего оператор NEXT проверяет условие выхода из цикла. Цикл прекращается, когда значение управляющей переменной оказывается строго больше (при положительном шаге приращения) или строго меньше (при отрицательном шаге приращения) конечного значения alfa. Если условие выхода из цикла не выполнено, управление передается к строке, следующей за оператором FOR этого цикла, т.е. в начало тела цикла. Расчет F производится уже при измененном значении alfa, и выводятся на печать следующие значения alfa и F. После окончания выполнения тела цикла все вышесказанное повторяется и т.д.

Результат вышеописанной программы (при m = 2000 кг и n = 4):

alfa,°	F, H
0	4905.0
5	4923.7
10	4980.7
15	5078.0
20	5219.8
25	5412.1
30	5663.8
35	5987.9
40	6403.0
45	6936.7
50	7630.8
55	8551.6
60	9810.0

Рассматриваемая циклическая структура может не содержать ключевого слова STEP. В этом случае шаг изменения управляющей переменной по умолчанию принимается равным +1. Например, в цикле

```
FOR I = 1 TO 10
   <блок операторов>
NEXT I
```

"блок операторов" будет выполняться при 10 значениях I (от 1 до 10 с шагом 1).

Допускается вкладывать циклы FOR ... NEXT, то есть помещать цикл FOR ... NEXT внутри другого цикла FOR ... NEXT. При этом оператор NEXT для внутреннего цикла должен предшествовать оператору NEXT для внешнего цикла. Имена управляющих переменных во вложенных циклах должны быть разными. Обычно при использовании циклов FOR ... NEXT первой управляющей переменной дается имя i, вложенному в него – j, затем k, l и далее по алфавиту.

Необходимость использования вложенных циклов возникает при обработке двумерных и многомерных массивов. Пример организации вложенной циклической структуры приведен ниже:

```
FOR I = 1 TO 10
    FOR J = 1 TO 20 STEP 2
        FOR K = 10 TO -10 STEP -5
        < блок операторов >
        NEXT K
    NEXT J
NEXT K
```

Здесь внешним циклом является цикл, определяемый переменной I, а вложенными – циклы по J и K.

При необходимости досрочного выхода из цикла используется оператор альтернативного выхода **EXIT FOR**. Например, при выполнении части программы

```
.
.
INPUT "Введите допустимое значение H"; Hdop
FOR Q = 1 TO 10 STEP 0.5
    H = R/K*Q^2
    IF H > Hdop THEN EXIT FOR
    PRINT USING " ##.#       ###.#"; Q, H
NEXT Q
IF H > Hdop THEN PRINT USING "Значение H=###.# выше допустимого !"; H
```

В данном случае возможен двоякий результат выполнения программы. Если при каком-либо значении Q будет получено такое H, что H > Hdop, то выполнение цикла прервется, а управление будет передано строке, следующей за NEXT. Если же ни при одном из Q условие H > Hdop не выполнится, цикл завершится, как обычно, при достижении управляющей переменной Q своего конечного значения.

При использовании циклической структуры FOR-TO-STEP-NEXT **необходимо также помнить следующее**:

– важно следить за точным совпадением количества операторов FOR и NEXT. При несоответствии возникает ошибка "FOR without NEXT" (FOR без NEXT) или "NEXT without FOR" (NEXT без FOR);

– недопустимо входить в цикл, минуя заголовок FOR (например, с помощью операторов условного или безусловного перехода). Это может привести к "зацикливанию", то есть бесконечному выполнению одной или нескольких строк;

– во избежание сбоев или неверного выполнения цикла не рекомендуется изменять в теле цикла шаг и управляющий параметр цикла;

– из цикла можно выходить досрочно, используя, как оператор альтернативного выхода EXIT FOR (управление передается строке, следующей за NEXT), так и конструкцию IF – GOTO – ELSE (управление передается строке с меткой, указанной оператором GOTO);

– завершающий оператор NEXT может не содержать имени управляющей переменной. В этом случае, по умолчанию, он будет принадлежать ближайшему открытому оператору FOR. Однако из соображений удобочитаемости текста программы оставлять оператор NEXT без имени переменной (особенно при наличии вложенных циклов) не рекомендуется;

– если вложенные циклы заканчиваются в одной строке, целесообразно вместо нескольких операторов NEXT использовать один, в котором через запятую, в порядке завершения циклов, перечислить имена управляющих переменных. Так, операторам NEXT K, NEXT J и NEXT I эквивалентен один оператор NEXT K, J, I;

– если начальное, конечное значения и шаг изменения управляющего параметра – целые числа, имеет смысл определить их, как целочисленные одинарной точности. Это значительно сокращает время работы цикла, особенно в программах, требующих интенсивных математических вычислений.

5.2.2. Циклическая структура с неизвестным числом повторений WHILE – WEND

Если количество повторений операторов тела цикла заранее неизвестно (например, при использовании некоторых математических методов, основанных на последовательном приближении к решению с заданной точностью), можно использовать циклическую структуру **WHILE-WEND**, выполняющую заключенный внутри нее блок операторов до тех пор, пока не будет выполнено условие, заданное оператором WHILE. Общий вид такой конструкции:

WHILE < условие >
    < блок операторов >
WEND

Рассмотрим работу данной структуры на примере следующего фрагмента программы:

```
X = 1
WHILE X < 8
   A = A + 1
   X = 3 * A ^ 2 – 10 * A
   Y = X ^ 2 + 3 * X – 5
   PRINT A, X, Y
WEND
```

До начала цикла производится проверка условия X<8. Так как данное условие выполняется, то начинают выполняться операторы тела цикла. Данный цикл будет повторяться до тех пор, пока верно условие, стоящее после WHILE.

При использовании оператора WHILE-WEND **необходимо помнить следующее**:

– циклы WHILE-WEND могут вкладываться друг в друга любое число раз, но каждому оператору WHILE должен соответствовать свой WEND; WHILE без WEND приводит к ошибке "WHILE without WEND", а WEND без WHILE – к ошибке "WEND without WHILE";

– альтернативный выход из цикла WHILE-WEND невозможен;

– необходимо оценивать реальность выполнения (невыполнения) условия, стоящего после WHILE, поскольку наложение некорректного условия может привести либо к бесконечному циклу, либо к невыполнению операторов тела цикла вообще;

– параметр, стоящий в условии после WHILE, должен изменяться внутри тела цикла;

– оператором WHILE можно задавать несколько условий, объединенных логическими операторами, например, в части программы

```
S = 6: L = 200
WHILE S < 22 OR L < 500
    < блок операторов >
WEND
```

<блок операторов> будет выполняться до тех пор, пока истинно одно из условий: S < 22 или L < 500. Здесь внутри цикла должно быть предусмотрено изменение определяющих параметров S и L, в противном случае "блок операторов" будет выполняться бесконечно.

5.2.3. Управляющий оператор DO-LOOP

Организовать цикл на QBasic можно с помощью управляющего оператора DO-LOOP, имеющего следующие формы:

1. Цикл DO-LOOP с проверкой выражения в начале:

DO WHILE … LOOP;
DO UNTIL … LOOP.

Рассмотрим логику такого варианта цикла на примере.

Рассчитать скорость движения воздуха в стволах различного диаметра и при получении недопустимого значения скорости выдать сообщение об этом. Здесь целесообразно использовать циклическую структуру DO-LOOP с условием. При несоблюдении условия – выйти из цикла и выдать сообщение о нахождении диаметра ствола, при котором скорость движения воздуха будет превышать допустимую. Вся программа будет выглядеть следующим образом:

```
CLS
PI = 4 * ATN(1): mu = .7
D = 9: Q = 6700
INPUT "Введите допустимое значение Vmax, м/с"; Vmax
DO WHILE V < Vmax
      S = PI * D ^ 2 / 4
      V = Q / (60 * S * mu)
      PRINT USING "При D=#.# м   V = ##.## м/с "; D; V
      D = D - 1
LOOP
PRINT "(что больше допустимой)"
```

После ввода исходных данных начинается выполнение циклической структуры DO-LOOP. Сначала осуществляется проверка условия цикла (в данном случае условие V < Vmax будет выполнено при задании пользователем любого положительного значения Vmax, поскольку переменная V до начала цикла не была определена, и по умолчанию ей будет присвоено значение 0). Затем выполнятся операторы тела цикла (т.е. операторы, заключенные между DO WHILE и LOOP). Здесь и происходит изменение управляющей переменной V. После окончания первого выполнения операторов цикла повторно проверяется условие V < Vmax. При его выполнении цикл повторяется, при невыполнении – управление передается оператору, следующему за LOOP и т.д. В данном случае в результате последнего выполнения цикла будет получено значение скорости, превышающее допустимую, после чего выполнится оператор PRINT, выдающий сообщение об этом.

Результат выполнения программы (при Vmax = 8 м/с):

```
При D=9.0 м   V= 2.51 м/с
При D=8.0 м   V= 3.17 м/с
При D=7.0 м   V= 4.15 м/с
При D=6.0 м   V= 5.64 м/с
При D=5.0 м   V= 8.12 м/с
(что больше допустимой)
```

Цикл DO UNTIL – LOOP отличается от DO WHILE – LOOP только тем, что цикл повторяется до тех пор, пока условие ложно (т.е. не выполняется), например, выполнение цикла

```
DO UNTIL Kpn > 1.2
     < блок операторов >
LOOP
```

будет выполняться до тех пор, пока значение управляющей переменной *Kpn* будет меньше 1,2.

2. Цикл DO-LOOP с проверкой выражения в конце:

DO … LOOP WHILE;
DO … LOOP UNTIL.

При организации такого цикла сначала выполняется блок операторов, заключенный между операторами DO и LOOP, затем проверяется условие цикла: если оно истинно (для оператора DO … LOOP WHILE) или ложно (для оператора DO … LOOP UNTIL), блок операторов цикла повторяется, в обратном случае выполнение цикла прерывается, и передача управления передается строке, следующей за строкой LOOP WHILE (LOOP UNTIL).

Например, при выполнении цикла

```
CLS
DO
   i = i + 1
   PRINT "Введите мощность "; i; "-го слоя"
   INPUT m(i)
   CLS
LOOP WHILE m(i) <> 0
PRINT "Общее количество слоев -"; i - 1
```

блок операторов, заключенный между DO и LOOP WHILE, повторяется до тех пор, пока при запросе "Введите мощность … -го слоя" не будет введено значение 0. Такая форма организации цикла удобна тем, что в данном случае не нужен дополнительный ввод общего числа задаваемых параметров, как этого требует циклическая структура FOR-TO-STEP-NEXT.

Вышеприведенный фрагмент программы можно (и логически правильнее) реализовать, используя структуру DO-LOOP UNTIL. В этом случае условие выхода из цикла изменится на противоположное, и программа примет вид:

```
CLS
DO
   i = i + 1
   PRINT "Введите мощность "; i; "-го слоя"
   INPUT m(i)
   CLS
LOOP UNTIL m(i) = 0
PRINT "Общее количество слоев -"; i - 1
```

3. Цикл DO-LOOP с альтернативным выходом

Если выполнение цикла должно быть прервано не в конце или начале тела цикла, а в его середине, то можно использовать циклическую структуру DO LOOP с альтернативным выходом. В этом случае цикл ограничивается служебными словами DO и LOOP (условия WHILE или UNTIL отсутствуют), а внутри цикла (в месте его прерывания) ставится оператор IF с условием, по которому происходит выход из цикла, обозначаемый служебными словами **EXIT DO**.

Например, в программе

```
CLS
INPUT "Введите начальный диаметр, м"; D
DO
    S = ATN(1) * D ^ 2
    h = .05 * D ^ 2 + .1 * D - .1
    i = i + 1
    IF S > 42 THEN EXIT DO
    PRINT USING "###  D=#.##  S=##.## "; i; D; S
    D = D + h
LOOP
PRINT "Расчет окончен"
```

выполнение цикла DO-LOOP будет продолжаться до тех пор, пока не выполнится условие S > 42, организующее альтернативный выход. Оператор EXIT DO передает управление строке, следующей за LOOP.

Альтернативный выход может быть использован и в циклах DO-LOOP с начальным или конечным условием WHILE (UNTIL). В этом случае завершение цикла осуществляется либо по условию WHILE (UNTIL), либо по условию IF … THEN EXIT DO, в зависимости от того, какое условие выполнится первым.

При использовании блочной структуры DO-LOOP **необходимо также помнить следующее**:

– как и другие циклические структуры, структура DO LOOP может быть вложенной, при этом количество операторов DO должно соответствовать числу LOOP, иначе возникает ошибка "LOOP without DO" (LOOP без DO) или "DO without LOOP";

– параметр, входящий в условие WHILE (UNTIL), должен изменяться внутри цикла, иначе цикл DO-LOOP (при выполнении данного условия) будет бесконечным;

– при несоблюдении условия WHILE в циклах DO WHILE … LOOP (т.е. с проверкой условия вначале) блок операторов, входящих в тело цикла не выполнится ни разу, а в циклах DO … LOOP WHILE (т.е. с проверкой условия в конце) – только один раз;

– при использовании альтернативного выхода из цикла, условный оператор IF … THEN EXIT DO должен находиться внутри цикла DO … LOOP, иначе возникает ошибка "EXIT DO not within DO … LOOP" (EXIT DO вне DO … LOOP).

5.3. Автоматический ввод исходных данных с применением операторов цикла

При использовании в программе большого количества числовых или символьных переменных, значение которых заранее известно (характеристики стандартных прокатных профилей, типовых изделий и конструкций, наименования строительных машин и механизмов и др.), целесообразно организовать автоматический ввод исходных данных, сформированных в качестве массивов. Для этого оператор считывания блока данных READ можно поместить внутри цикла и, изменяя номер (индекс) элемента массива, поочередно присваивать переменным значения, перечисленные в операторе DATA.

Например, при выполнении программы

```
REM Задание площади поперечного сечения двутавра
DIM FR(12)
DATA 21.5,32.9,35.6,39.6,42.1,48.7,54.5,59.9,64,73.8,90.9,98.8
FOR I = 1 TO 12
   READ FR(I)
NEXT N
```

первому элементу массива FR(1) будет присвоено значение 21,5; FR(2) = 32,9 и т.д.

Аналогичным образом можно поступать и при задании значений элементам двумерного массива. В этом случае необходимо организовать вложенный цикл, а присваиваемые значения для удобства составления и отладки программы целесообразно записывать, используя несколько операторов DATA, причем их количество определять числом строк матрицы, например:

```
REM Задание линейной плотности прокатных профилей
DATA 16.9,25.8,27.9,31.1,33.1,38.3,42.8,47,50.2,57.9,71.3,77.6,0,0,0
DATA 20.2,23,22.6,39.1,0,0,0,0,0,0,0,0,0,0,0
DATA 38.8,43,51,49.4,54,58.8,68,77,61,66.2,74,79.8,85.6,97.4,120.2
DATA 39.6,39.8,59.6,58.3,58.3,90.5,0,0,0,0,0,0,0,0,0
FOR PROF = 1 TO 4
   FOR NP = 1 TO 15
     READ LP (PROF, NP)
NEXT NR, RAS
```

Здесь внешний цикл (с управляющей переменной PROF) изменяет тип

прокатного профиля (двутавр, швеллер, коробчатый из неравнобоких уголков, коробчатый из равнобоких уголков), а внутренний цикл (с управляющей переменной NP) изменяет порядковый номер профиля внутри каждого из перечисленных типов. Таким образом, число строк (4) определяет количество различных типов прокатных профилей, а число столбцов (15) – максимальное количество различных типоразмеров одного профиля. Поскольку 15 элементов будет содержаться только в 3-й строке (т.е. при PROF = 3), а в остальных строках количество элементов будет меньшим, поэтому недостающие значения (т.е. линейные плотности несуществующих профилей) могут быть заполнены нулями. В таком случае сохранится верный размер матрицы (4×15), а наглядный вид операторов DATA, при котором в каждой из строк содержатся значения только одного типа профилей, позволит избежать ошибок в организации ввода исходных данных. Аналогично может быть организован автоматический ввод и других постоянных величин (основных размеров профилей, их моментов инерции, моментов сопротивления, площади поперечного сечения и др.)

Такой ввод данных удобен для пользователя тем, что по заданному типу и типоразмеру профиля программа может "сама" определить все необходимые для расчета значения, взяв их из заранее сформированных двумерных массивов.

Подобным образом можно вводить и любую символьную информацию, для этого в цикле значения должны присваиваться символьным переменным, например:

```
REM Типоразмер двутавра
DATA 14C,18M,20C,20Ca,22C,24M,27C,27Ca,30M,36M,36C,45M
FOR TD = 1 TO 12
   READ Dv$(TD)
NEXT TD
```

Такая организация присвоения значений символьным переменным соответствует следующим операторам присвоения: Dv$(1) = "14C", Dv$(2) = "18M", Dv$(3) = "20C" и т.д.

5.4. Организация печати таблиц с использованием псевдографики и операторов цикла

Операторы цикла позволяют просто и удобно организовать вывод на печать всевозможных таблиц. Для этого внутрь цикла вставляется только одна строка, определяющая выводимые переменные по заданному формату. Для получения непрерывных границ таблицы целесообразно использовать символы псевдографики. Ввод таких символов в программу может быть произведен путем ввода соответствующих им кодов. Для этого необходимо, удерживая клавишу Alt, набрать на дополнительной цифровой клавиатуре трехзначный код. Кодировки символов для построения таблиц и линий приведены в табл. 7 и 8.

Таблица 7

Кодировки символов для построения таблиц

┌	218	┬	194	┐	191
├	195	┼	197	┤	180
└	192	┴	193	┘	217
╔	218	╦	194	╗	191
╠	195	╬	197	╣	180
╚	192	╩	193	╝	217

Таблица 8

Кодировки символов для рисования линий

│	179
─	196
║	186
═	205

Рассмотрим организацию построения таблицы на примере следующего фрагмента программы:

```
CLS
DIM Stm$(7), ro(7), por(7), tp(7)
DATA тяжелый бетон, легкий бетон, ячеистый бетон
DATA кирпич, гранит, стеклопластик, сосновая доска
FOR i = 1 TO 7
   READ Stm$(i)
NEXT i
DATA 2.4,1,0.5,1.8,2.67,2.65,2
FOR i = 1 TO 7
   READ ro(i)
NEXT i
DATA 10,61.5,81,32,1.4,0,0
FOR i = 1 TO 7
   READ por(i)
NEXT i
DATA 1.16,0.35,0.2,0.8,2.8,0.58,0.5
FOR i = 1 TO 7
   READ tp(i)
NEXT i
.
.
.
REM Печать исходных данных
PRINT "┌──────────────────┬──────────┬──────────┬──────────────────┐"
PRINT "│ Строительный     │ Плотность,│ Пористость,│ Теплопроводность,│"
PRINT "│ материал         │ г/см куб. │ %          │ Вт/(м·°C)        │"
PRINT "├──────────────────┼──────────┼──────────┼──────────────────┤"
FOR i = 1 TO 7
```

```
PRINT USING"│\          \│   ##.##   │    ##.#   │     ##.##      │";Stm$(i);ro(i);por(i);tp(i)
NEXT i
PRINT "└──────────┴─────────┴──────────┴────────────────┘"

FOR i = 1 TO 7
.
.       < Блок вычислительных операций
          и печать результатов расчета >
NEXT
```

Такая организация ввода исходных данных позволяет не только удобно выводить на печать таблицы, но и производить любые, многократно повторяющиеся вычислительные действия, поскольку исходные данные хранятся в соответствующих массивах в течение всего времени выполнения программы. Если же такой необходимости нет, вывод таблиц на печать может быть организован и без создания массивов, в этом случае выводимые в цикле на печать переменные изменяют свои значения, и одновременное хранение всех исходных данных не предусматривается. Это позволяет освободить память и ускорить выполнение программы.

Рассмотрим вариант построения этой же таблицы без применения массивов исходных данных.

```
CLS
DATA тяжелый бетон,2.4,10,1.16
DATA легкий бетон,1,61.5,0.35
DATA ячеистый бетон,0.5,81,0.2
DATA кирпич,1.8,32,0.8
DATA гранит,2.67,1.4,2.8
DATA стеклопластик,2.65,0,0.58
DATA сосновая доска,2,0,0.5
PRINT "┌──────────┬─────────┬──────────┬────────────────┐"
PRINT "│Строительный│Плотность,│Пористость,│Теплопроводность,│"
PRINT "│  материал  │г/см куб. │    %     │   Вт/(м·°C)    │"
PRINT "└──────────┴─────────┴──────────┴────────────────┘"
FOR i = 1 TO 7
READ Stm$, ro, por, tp
PRINT USING"│\          \│   ##.##   │    ##.#   │     ##.##      │";Stm$; ro; por; tp
NEXT i
PRINT "└──────────┴─────────┴──────────┴────────────────┘"
```

Результат выполнения данного фрагмента программы приведен ниже.

Строительный материал	Плотность, г/см куб.	Пористость, %	Теплопроводность, Вт/(м·°C)
тяжелый бетон	2.40	10.0	1.16
легкий бетон	1.00	61.5	0.35
ячеистый бетон	0.50	81.0	0.20
кирпич	1.80	32.0	0.80
гранит	2.67	1.4	2.80
стеклопластик	2.65	0.0	0.58
сосновая доска	2.00	0.0	0.50

Как видно, использование циклических структур позволяет при компактной записи программы осуществлять ввод большого объема исходной информации, производить многократно повторяющиеся вычислительные

операции, организовывать удобный вывод на печать произвольных таблиц, а также применять некоторые типовые приемы программирования, речь о которых пойдет в главе 6.

5.5 Лабораторная работа №3

Тема: Программирование алгоритмов циклической структуры

Цель работы: Научиться составлять алгоритмы циклической структуры и программы расчета параметров по заданным формулам при необходимости многократного повторения операций.

Задание: Согласно варианту задания составить блок-схему алгоритма и программу на языке QBasic по расчету заданных параметров при изменении исходных данных с указанным шагом. Результаты расчета вывести на печать в форме таблицы.

5.5.1. Варианты лабораторной работы

Вариант 1

По условиям варианта 1 лабораторной работы № 2 рассчитать скорость движения воздуха v, м/с, при изменении диаметра ствола в свету $D_{св}$ от 9 м до минимального значения $D_{св\ min}$, при котором выполняется условие $v \le \langle V \rangle$. Шаг изменения $D_{св}$ принять равным 0,5 м. На печать вывести таблицу значений $D_{св}$ и v. Построить график зависимости v от $D_{св}$.

Вариант 2

Рассчитать величину давления горных пород, МПа, используя выражение

$$P_г = \sum_{i=1}^{n} \gamma_i \cdot h_i \cdot tg^2\left(\frac{90 - \varphi_i}{2}\right),$$

где γ_i – удельный вес пород i-го слоя геологического разреза, МН/м^3;

h_i – мощность пласта пород i-го слоя, м;

φ_i – угол внутреннего трения породы пласта, в котором определяют давление, °.

Исходные данные для расчета принять согласно следующей таблицы:

i	γ_i, МН/м^3	h_i, м	φ_i, °
1	0,0196	22	29
2	0,0228	35	27
3	0,027	44	45
4	0,025	8	38
5	0,0155	2	36
6	0,024	12	29

Результаты выполнения программы вывести на печать в форме таблицы, содержащей столбцы исходных данных и столбец значений давлений, оказываемых каждым слоем породы. Ниже последнего столбца поместить значение суммарного давления всех слоев.

Вариант 3

Рассчитать сметную стоимость горнопроходческих работ, используя выражение:

$$C_{см} = \sum_{i=1}^{n} V_i \cdot P_{ед.i} \cdot \left(1 + \frac{Ш}{100}\right) \cdot \left(1 + \frac{Н}{100}\right) \cdot \left(1 + \frac{П}{100}\right),$$

где V_i – объем работ *i*-го вида;

$P_{ед.i}$ – единичная расценка *i*-го вида, руб.;

$Ш$ – норматив общешахтных расходов, %;

$Н$ – норма накладных расходов, %;

$П$ – норма плановых накоплений, %.

Исходные данные принять согласно следующей таблицы:

i	1	2	3	4	5
V_i	1450	233	500	1235	1235
$P_{ед.i}$	2,55	3,18	4,20	1,25	2,50

В качестве нормативов принять следующие значения: Ш = 78%, Н = 23%, П = 8%.

Результаты выполнения программы вывести на печать в форме таблицы, содержащей столбцы исходных данных и столбец значений затрат по каждому виду работ. Ниже последнего столбца поместить значение сметной стоимости горнопроходческих работ.

Вариант 4

Рассчитать средневзвешенную плотность пород $\gamma_{ср}$, кг/м^3, пересекаемых стволом, используя выражение

$$\gamma_{ср} = \frac{\sum_{i=1}^{n} \gamma_i \cdot h_i}{\sum_{i=1}^{n} h_i},$$

где γ_i – плотность *i*-го слоя пород, кг/м^3;

h_i – мощность *i*-го слоя пород, м.

Исходные данные для расчета принять согласно следующей таблицы:

i	1	2	3	4	5	6	7	8	9
γ_i, кг/м^3	2310	2440	2510	2780	2500	2460	1960	1620	2450
h_i, м	12	28	41	13	24	33	18	2	47

На печать вывести таблицу исходных данных и значение средневзвешенной плотности пород.

Вариант 5

Определить суммарные трудозатраты $T_ц = \Sigma Ti$, чел-ч, и расценку $З_ц = \Sigma З_i$, руб., на цикл, если:

T_i – трудозатраты по i-му нормируемому процессу, чел-ч:

$$T_i = H_{врi} \cdot W_i \cdot K_i \ ;$$

$З_i$ – расценка по i-му нормируемому процессу, руб.,

$$З_i = P_i \cdot W_i \cdot K_i \ ,$$

где $H_{врi}$ – норма времени по ЕНиР на единицу объема работ по i-му процессу, чел.-ч/ед. об. работ;

P_i – расценка по ЕНиР на единицу объема работ по i-му процессу, руб./ед. об. работ;

W_i – объем работ по i-му процессу, ед. об. работ;

K_i – поправочный коэффициент к норме времени и расценке по i-му процессу.

Исходные данные для расчета принять согласно следующей таблицы:

i	W_i	$H_{врi}$, чел-ч	P_i, руб.-коп.	K_i
1	3,6	1	2-23	1
2	10,6	0,23	0-51,3	1
3	6,51	1,1	2-45	1,11
4	24	0,27	0-60,2	1,11
5	2	9	20-07	1,05
6	2	0,62	1-38	1,05
7	1,25	3,8	8-47	1
8	2	0,78	1-74	1
9	0,2	0,54	1-20	1
10	0,2	0,19	0-42,4	1

Результаты выполнения программы вывести на печать в форме таблицы, содержащей столбцы исходных данных и столбцы значений трудозатрат и расценки по каждому процессу. Ниже последних столбцов поместить значения суммарных трудозатрат и расценки на проходческий цикл.

Вариант 6

Рассчитать продолжительность каждого процесса проходческого цикла B_i, мин, используя выражение:

$$B_i = \frac{T_i \cdot K_{взр}}{N_я \cdot K_{п.н.}},$$

где T_i – трудоемкость i-го процесса проходческого цикла, чел-ч;

$K_{взр}$ – коэффициент взрывных работ,

$$K_{взр} = \frac{T_ц - T_{взр}}{T_ц},$$

где $T_ц$, $T_{взр}$ – соответственно трудоемкость работ проходческого цикла и взрывных работ, чел-ч;

$N_я$ – явочный состав звена проходчиков, чел;

$K_{п.н}$ – коэффициент превышения норм выработки.

Трудоемкость взрывных работ рассчитать по формуле

$$T_{взр} = B_{взр} \cdot N_я \cdot K_{п.н},$$

где $B_{взр}$ – продолжительность взрывных работ, час,

$$B_{взр} = \frac{N_{шп} \cdot t_{шп}}{60 \cdot N_з},$$

где $N_{шп}$ – число шпуров;

$N_з$ – число проходчиков, занятых при заряжании;

$t_{шп}$ – время заряжания одного шпура, мин.

Трудоемкость проходческого цикла складывается из трудоемкостей отдельных процессов: $T_ц = \Sigma T_i$.

Расчет произвести для следующих исходных данных: $K_{п.н} = 1{,}12$; $N_я = 4$ чел., $N_з = 3$ чел., $t_{шп} = 2$ мин. Число шпуров задать самостоятельно. Трудоемкость процессов принять согласно следующим данным:

i	1	2	3	4	5	6	7	8
T_i, чел-ч	4,2	2,44	7,16	6,48	18,8	1,24	4,75	1,56

На печать вывести трудоемкость и продолжительность каждого процесса и суммарные трудоемкость и продолжительность проходческого цикла. Продолжительность выразить в часах и минутах.

Вариант 7

Для построения аэродинамической характеристики вентиляционного трубопровода в координатах *Q-H* рассчитать значения давления вентилятора H_i, даПа, для различных значений подачи воздуха Q_i, м^3/с, если параметры Q и H связаны соотношением

$$H_i = \frac{R_{тр}}{K_{ут.тр.}} \cdot Q_i^2,$$

где $R_{тр}$ – полное аэродинамическое сопротивление трубопровода, кμ;

$K_{ут.тр}$ – коэффициент утечек трубопровода.

Расчет произвести для $R_{тр} = 7{,}17$ кμ, $K_{ут.тр} = 1{,}13$. Для построения аэродинамической характеристики трубопровода Q_i изменять от 1 до 10 м^3/с с шагом $\Delta Q_i = 1$ м^3/с. На печать вывести таблицу значений H_i и Q_i . По результатам расчета построить аэродинамическую характеристику вентиляционного трубопровода.

Вариант 8

Рассчитать число шпуров в стволе для каждой из n окружностей, если число шпуров i-й окружности определяется формулой

$$N_i = \frac{4 \cdot \pi \cdot \Delta R \cdot n_i}{a},$$

где ΔR – расстояние между окружностями, м,

$$\Delta R = \frac{0{,}5 \cdot D_{пр}}{n},$$

где $D_{пр}$ – диаметр ствола в проходке, м;

n – число окружностей расположения шпуров;

n_i – порядковый номер окружности от центра ствола;

a – расстояние между шпурами в окружности, м.

Диаметр ствола в проходке $D_{пр}$ и расстояние между шпурами в окружности a принять самостоятельно, число окружностей выбрать с таким расчетом, чтобы расстояние между окружностями находилось в пределах 0,5÷0,8 м.

Вариант 9

Рассчитать среднюю скорость проведения камер околоствольного двора $v_{ср}$, м/мес, используя выражение

$$v_{ср} = \frac{W}{K \cdot \sum_{i=1}^{n} \frac{W_i}{v_i}},$$

где W – суммарный объем камер в свету, м^3;

W_i – объем i-го участка камер, м^3;

v_i – скорость проведения i-го участка, м^3/мес;

K – коэффициент совмещения работы в различных забоях (принять $K = 0{,}8$).

Расчет произвести для данных, приведенных в следующей таблице:

W_i, м^3	275	290	163	110	110	350	400
v_i, м^3/мес	230	420	330	450	430	520	600

Вариант 10

Рассчитать производительность подъема при проходке ствола P_n, м^3/ч, для всех указанных типоразмеров проходческих бадей, используя выражение

$$P_n = \frac{3600 \cdot V_б \cdot k_б}{t_ц},$$

где $t_ц$ – продолжительность полного цикла подъема бадьи по стволу, с;

$k_б$ – коэффициент заполнения бадьи, $k_б = 0{,}9$;

$V_б$ – вместимость бадьи, принимаемая согласно следующей таблицы в зависимости от типоразмера бадьи, м^3:

Типоразмер	БПСМ-0,75	БПС-1	БПСМ-1,5	БПС-2	БПС-2,5	БПС-3
$V_б$, м^3	0,75	1	1,5	2	2,5	3
Типоразмер	БПСМ-3,5	БПС-4	БПСМ-4,5	БПС-5	БПС-6,5	
$V_б$, м^3	3,5	4	4,5	5	6,5	

На печать вывести таблицу типоразмеров бадей и соответствующих им производительностей подъема.

Вариант 11

По условиям варианта 11 лабораторной работы № 2 рассчитать продолжительность крепления ствола $T_{кр}$, ч-мин., при изменении толщины монолитной бетонной крепи δ от 200 до 500 мм с шагом 50 мм. На печать вывести таблицу значений $T_{кр}$ и δ.

Вариант 12

Рассчитать лобовую жесткость консольной расстрельной балки $C^л$, Н/м, закрепленной анкерами, при изменении длины консоли l от 0,3 до 1,5 м с шагом 0,1 м, используя выражение

$$C^л = \frac{3 \cdot E \cdot I_z \cdot \nu}{l^3},$$

где E – модуль продольной упругости материала балки, Н/м^2;

I_z – момент инерции поперечного сечения балки относительно центральной вертикальной оси, м^4;

ν – коэффициент ослабления жесткости балки в лобовом направлении вследствие влияния заделки анкеров (принять равным $\nu = 1/exp\ l$);

Исходные данные задать самостоятельно. По результатам расчета построить график зависимости $C^л$ от l. На печать вывести таблицу значений l и соответствующих им $C^л$.

Вариант 13

По условиям варианта 13 лабораторной работы № 2 рассчитать коэффициент утечек вентиляционного трубопровода $k_{ут.тр.}$ при изменении его диаметра $d_{тр}$ от 0,4 до 1,1 м с шагом 0,1 м и построить график зависимости $k_{ут.тр}$ от $d_{тр}$.

На печать вывести таблицу значений заданных диаметров $d_{тр}$ и соответствующих им $k_{ут.тр.}$

Вариант 14

Рассчитать скорость сооружения скважины V_c, м/смену, используя выражение:

$$V_c = \frac{l}{\Sigma t + \frac{l}{V_{пр}}},$$

где l – глубина скважины, м;

Σt – время монтажа и демонтажа бурового станка, смен (Σt = 15 смен);

$V_{пр}$ – техническая скорость проходки скважины, м/смену,

$$V_{пр} = 3 + \frac{2{,}4}{D - 0{,}2},$$

где D – диаметр скважины, м.

Расчет произвести для следующих диаметров скважин, м: 1,02 ; 1,2 ; 1,56 ; 2,08 ; 2,25 ; 2,6 ; 3. Глубину скважины задать самостоятельно.

На печать вывести таблицу значений D и V_c.

Вариант 15

Рассчитать мощность пород b, м, для скрепления анкерами при изменении угла внутреннего трения боковых пород φ от 26 до 60° с шагом 2°:

$$b = \frac{a + h \cdot ctg\dfrac{90 + \varphi}{2}}{tg\varphi_1} \cdot K_m,$$

где a – полупролет выработки, м;

h – высота выработки, м;

φ, φ_1 – соответственно угол внутреннего трения боковых пород и пород кровли;

K_m – коэффициент запаса, зависящий от трещиноватости пород, принять K_m = 1,8.

Предусмотреть ручной ввод параметров a, h, и φ_1. На печать вывести таблицу значений φ и b.

5.5.2. Примеры выполнения лабораторной работы

Пример 1. Известно, что прочность бетона при твердении в нормальных условиях возрастает по закону:

$$R_n = R_{28} \cdot \frac{\lg n}{\lg 28},$$

где R_n – прочность бетона (предел прочности при сжатии) в возрасте n сут;

R_{28} – прочность бетона в возрасте 28 сут;

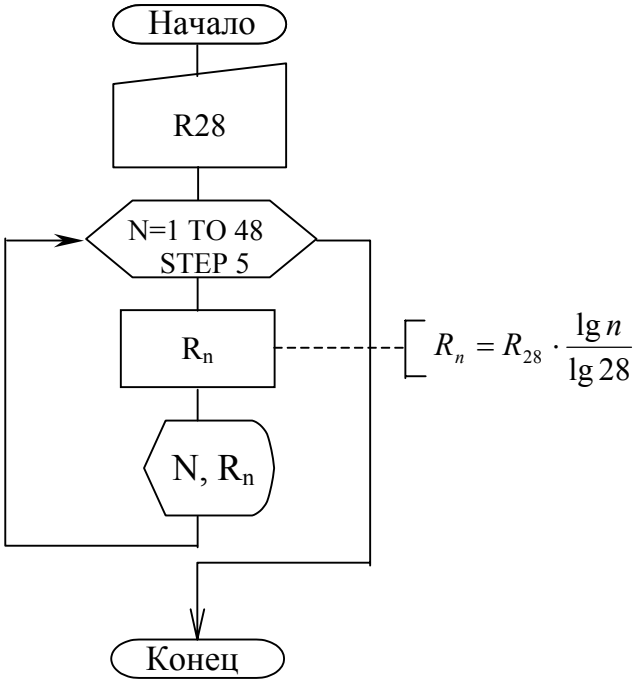
n – возраст бетона, сут.

Определить значение прочности бетона через каждые 5 сут. Начальное значение R_n принять равным 3 сут, конечное – 50 сут. На печать вывести таблицу значений возрастов и соответствующих им прочностей бетона. Предусмотреть ручной ввод значения прочности бетона в 28-суточном возрасте.

Присваиваем имена переменным и составляем таблицу идентификаторов:

Переменная	Идентификатор	Примечание
R_n	RN	Рассчитываемый параметр
R_{28}	R28	Параметр, задаваемый вручную
–	k	Значение lg 28

Составляем блок-схему алгоритма:



Программа на QBasic:

```
CLS
INPUT "Введите прочность бетона в возрасте 28 сут, МПа"; R28
k = LOG(28) / LOG(10)
PRINT "┌──────────────────────┬──────────────────────┐"
PRINT "│      N, сут          │      Rn, МПа   "
PRINT "├──────────────────────┼──────────────────────┤"
FOR N = 3 TO 48 STEP 5
   RB = R28 * LOG(N) / LOG(10) / k
   PRINT USING "│      ##         │        ##.##         │"; N; RB
NEXT N
PRINT "└──────────────────────┴──────────────────────┘"
```

Результат выполнения программы (при R28 = 15 МПа):

N, сут	Rn, МПа
3	4.95
8	9.36
13	11.55
18	13.01
23	14.11
28	15.00
33	15.74
38	16.37
43	16.93
48	17.43

Пример 2. Рассчитать расход ВВ – Q, кг, на цикл для указанных типов ВВ, используя выражение

$$Q = q \cdot S \cdot l,$$

где q – удельный расход ВВ,

$$q = q_1 \cdot f_1 \cdot v \cdot e\,,$$

где q_1 – нормальный удельный расход ВВ, $q_1 = 0{,}1 f$

f – крепость взрываемых пород по шкале проф. Протодьяконова;

f_1 – коэффициент структуры породы;

v – коэффициент зажима породы, $v = \dfrac{6{,}5}{\sqrt{S}}$,

S – площадь поперечного сечения выработки, м^2;

e – коэффициент работоспособности ВВ.

Принять равным S = 12,5 м^2; f = 7; f_1 = 1,3; l = 2,2 м. Расчет произвести для следующих типов ВВ:

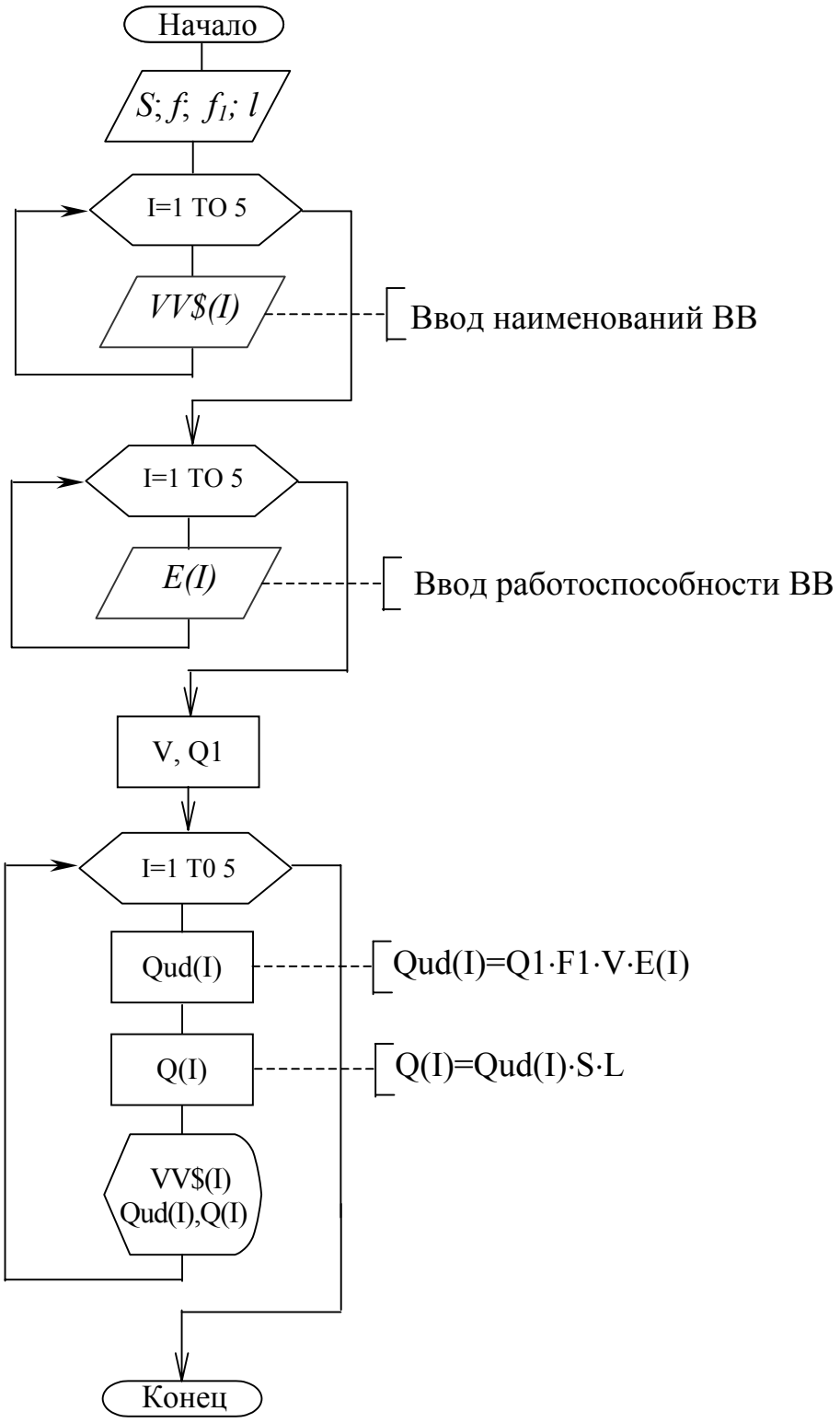
Тип ВВ	Коэффициент e
Аммонит скальный №1	0,8
Детонит М	0,82
Аммонит № 6ЖВ	1
Динафталит-200	1,12
Аммонит АП-5ЖВ	1,17

На печать вывести таблицу, содержащую следующие столбцы: наименование ВВ, его работоспособность, удельный и общий расход ВВ на цикл.

Присваиваем имена переменным и составляем таблицу идентификаторов:

Переменная	Идентификатор	Примечание
S	S	Параметры, вводимые автоматически
l	L	
f_1	F1	
f	F	
v	V	Рассчитываемые параметры
q_1	Q1	
–	I	Счетчик цикла
–	VV$(I)	Элементы массивов исходных данных (наименование и работоспособность ВВ)
e	E(I)	
q	Qud(I)	Параметры, рассчитываемые в цикле
Q	Q(I)	

Составляем блок-схему алгоритма:



Программа на языке QBasic:

```
CLS
DIM VV$(5), E(5)
READ S, F, F1, L
FOR I = 1 TO 5
   READ VV$(I)
NEXT I
FOR I = 1 TO 5
   READ E(I)
NEXT I
Q1 = .1 * F
V = 6.5 / SQR(S)
PRINT "┌──────────────────────────┬──────────────────┬──────────────────┐"
PRINT "│   Наименование ВВ        │Удельный расход,  │   Общий расход,  │"
PRINT "│                          │   кг/м куб       │        кг        │"
PRINT "├──────────────────────────┼──────────────────┼──────────────────┤"
FOR I = 1 TO 5
   Qud(I) = Q1 * F1 * V * E(I)
   Q(I) = Qud(I) * S * L
   PRINT USING "│\                \│"; VV$(I);
   PRINT USING "    ##.##       │      ###.##     │"; Qud(I); Q(I)
NEXT I
PRINT "└──────────────────────────┴──────────────────┴──────────────────┘"
DATA 12.5,7,1.3,2.2
DATA аммонит скальный №1,детонит М, аммонит 6ЖВ,динафталит-200,аммонит АП-5ЖВ
DATA 0.8,0.82,1,1.12,1.17
```

Результат выполнения программы:

Наименование ВВ	Удельный расход, кг/м куб	Общий расход, кг
аммонит скальный №1	1.34	36.81
детонит М	1.37	37.73
аммонит 6ЖВ	1.67	46.01
динафталит-200	1.87	51.53
аммонит АП-5ЖВ	1.96	53.83

6. ХАРАКТЕРНЫЕ ПРИЕМЫ ПРОГРАММИРОВАНИЯ

В этой главе будут рассмотрены приемы, часто применяемые для решения прикладных задач при разработке программ. Данные приемы часто позволяют значительно сократить текст программы, сделать ее нагляднее и проще, наиболее удобно для пользователя организовать ряд вычислительных и других операций.

6.1. Организация функций, определяемых пользователем

Кроме стандартных функций, имеющихся в QBasic (п. 2.5), пользователь может определить любую собственную функцию. Это делается с помощью оператора **DEF**, после которого указывается имя функции, начинающееся обязательно с букв **FN**. Общий вид этой функции:

DEF FN<*имя*> (x_1, x_2, … x_n) = <*выражение*>,

где <*имя*> – задается как имя простой переменной, но пишется слитно с FN и вместе образует имя функции пользователя;

x_1, x_2, … x_n – список параметров функции пользователя;

<*выражение*> – любое арифметическое выражение, определяющее какие действия производит данная функция.

Например, для расчета значения логарифма числа x по основанию a (а такой функции нет среди стандартных), может быть задана следующая функция пользователя:

```
DEF FNlg(x,a) = LOG(x)/LOG(a)
```

Переменные x и a, которые указываются в списке параметров в определении функции, называются формальными параметрами. При вызове функции каждый формальный параметр заменяется некоторым конкретным значением (фактическим параметром). Например, при расчете $\log_2 18$, можно вызвать функцию FNlg с указанием конкретных значений x и a:

```
Y = FNlg(18,2)
```

Функции пользователя целесообразно также использовать в том случае, когда какое-либо громоздкое выражение рассчитывается несколько раз при различных значениях аргументов, причем расчет выполняется не в цикле, а в различных частях программы. Например:

```
DEF FNA(X, Y)=(1-SIN(X))*(1000+Y*TAN(X))
```

Далее в программе достаточно написать FNA(0.5,600), и программа в действительности вычислит выражение (1–SIN(0.5))*(1000+600*TAN(0.5)). При вызове функции пользователя вместо формальных параметров (в последнем случае – это X и Y), могут стоять не только численные значения, но любые стандартные функции QBasic, например: FNA(SIN(X),500). В этом случае будет вычислено выражение (1–SIN(SIN(X)))*(1000+500*TAN(SIN(X))).

Кроме того, функции пользователя могут обращаться к самим себе, то есть допускают рекурсию, например:

```
FNA(0.5,FNA(0.8,Y))
```

В этом случае вначале рассчитывается значение внутренней функции, а затем внешней.

При использовании функции пользователя **следует также помнить следующее**:

– оператор DEF FN должен располагаться в программе до первого вызова определяемой им функции, в противном случае возникает ошибка "Function not defined" (функция не определена);

– к моменту вычисления функции пользователя ее формальные параметры, в том числе заданные в виде стандартных функций или других функций пользователя, должны быть определены. Если аргументы, стоящие на месте формальных параметров функции пользователя не определены к моменту ее вызова, им автоматически присваивается значение 0;

– тип функции указывается, как и тип переменной. При этом используются символы объявления типа, например, DEF FNm%(x)=INT(SQR(x));

– при несоответствии количества аргументов в вызываемой функции пользователя с числом формальных параметров, заданных при определении данной функции, возникает ошибка "Argument – count mismatch" (несоответствие количества аргументов).

6.2. Организация подпрограмм

Часто при составлении программы возникает необходимость выполнения в разных ее частях одинаковой последовательности некоторых операций. Для сокращения текста программы в этом случае целесообразно использовать *подпрограммы* – обособленные от основной программы группы операторов, которые имеют собственное имя (метку).

Для обращения к подпрограмме используют оператор **GOSUB <n1>**, где <n1> – имя подпрограммы или метка первого оператора подпрограммы.

Любая подпрограмма должна завершаться оператором возврата **RETURN <n2>**, где <n2> – метка строки, куда возвращается управление после выхода из подпрограммы. Метка <n2> может отсутствовать, в этом случае оператор RETURN передает управление оператору, следующему за оператором GOSUB в основном тексте программы.

Подпрограмма может находиться в любом месте программы, однако удобнее помещать ее в конце основного текста программы после оператора END.

При использовании оператора вызова подпрограммы **необходимо помнить следующее**:

– при вызове подпрограммы все необходимые значения переменных, используемых в ней должны быть определены;

– при многократном обращении к подпрограмме необходимо сохранять результаты ее выполнения, иначе они будут изменены при повторных обращениях к ней;

– в качестве метки, обозначающей начало подпрограммы, может стоять либо число, либо имя, начинающееся с буквы и состоящее из любого набора букв и цифр. В последнем случае метка должна отделяться от текста подпрограммы двоеточием, иначе возникает ошибка "Label not defined" (метка не определена);

– ошибка "Label not defined" также может возникнуть при отсутствии подпрограммы с меткой, имя которой указано оператором GOSUB;

– внутри подпрограммы можно использовать переход на другую подпрограмму низшего уровня.

В качестве примера использования подпрограммы приведем следующий способ решения одной из прикладных задач.

Задача. Определить площади поперечного сечения и объемы вынимаемой за цикл породы при проведении трех горных выработок полуциркульной формы, если известна ширина B_i каждой из выработок, высота их прямолинейной части H_i, глубина шпуров l_i, коэффициент использования шпуров η_i, коэффициент излишка сечения μ_i, коэффициент разрыхления пород K_i. Рассчитать необходимое количество вагонеток на каждый забой N_i и общее количество вагонеток N при одновременной погрузке породы в трех подготовительных забоях. Предусмотреть ручной ввод объема принятой вагонетки Q_i. Коэффициент заполнения вагонетки $k_з$ принять равным 0,9.

Алгоритм расчета площади поперечного сечения, объема вынимаемой за цикл породы и необходимого количества вагонеток одинаков для всех выработок. Площадь сечения S_i определяется как сумма площади прямоугольника $S_{npi} = B_i \cdot H_i$ и половины площади круга диаметром B_i, $S_{ki} = \pi \cdot B_i^2 / 8$. Объем вынимаемой за цикл породы в разрыхленном состоянии определяется формулой $V_i = S_i \cdot l_i \cdot \eta_i \cdot \mu_i \cdot K_i$. Количество вагонеток рассчитывается из соотношения $N_i = \dfrac{V_i}{k_з Q_i}$.

Данный алгоритм будет использоваться трижды при различных исходных данных, поэтому его можно организовать в виде подпрограммы. В основной программе остается только изменять исходные данные и обращаться к подпрограмме. Необходимо учесть, что при каждом выходе из подпрограммы необходимо запоминать значения N_i для последующего их суммирования.

Присвоим подпрограмме имя SPS, тогда текст программы будет выглядеть следующим образом:

```
CLS
PI=3.1416: KZ=0.9
DATA 4,1.8, 2.2, 0.9, 1.05, 2.2,1
DATA 3.5, 1.6, 2, 0.95, 1.04, 1.95,2
DATA 3.2, 1.4, 2.4, 0.9, 1.08, 2.4,3
INPUT "Введите объем принятой вагонетки "; Q
READ B, H, L, TETA, MU, K, I: GOSUB SPS: N1=N
READ B, H, L, TETA, MU, K, I: GOSUB SPS: N2=N
READ B, H, L, TETA, MU, K, I: GOSUB SPS: N3=N
NO=N1+N2+N3
PRINT USING "Общее количество вагонеток – ###";NO
END

SPS:
S=B*H+PI*B^2/32
V=S*L*TETA*MU*K
N=CINT(V/(KZ*Q)+0.5)
PRINT I;"- я выработка:"
PRINT USING "S=##.# м кв.  V=###.# м куб.   N=## вагон."; S;V;N
RETURN
```

Результат выполнения программы (при Q = 2,5 м куб.):

```
1 - я выработка:
S=13.5 м кв.  V= 61.7 м куб.   N=28 вагон.
2 - я выработка:
S=10.4 м кв.  V= 40.1 м куб.   N=18 вагон.
3 - я выработка:
S= 8.5 м кв.  V= 47.6 м куб.   N=22 вагон.
Общее количество вагонеток -  68
```

В данной программе перед каждым обращением к подпрограмме SPS производится считывание необходимых данных оператором READ, а после каждого возврата из подпрограммы переменным N1, N2 и N3 присваивается текущее значение количества вагонеток N. Переменная I определяет номер обращения к подпрограмме или номер выработки. Таким образом, использование подпрограммы позволило значительно сократить текст программы.

Следует заметить, что данная задача может быть успешно решена и с использованием операторов цикла. В этом случае считывание исходных данных и все расчетные формулы необходимо включить в тело цикла и трижды повторить заданный цикл, накапливая сумму значений N внутри цикла (о данном приеме программирования более подробно см. пп. 6.3 и 6.4).

Оптимальный способ решения задачи должен выбираться в каждом конкретном случае исходя из индивидуальных навыков и способностей программиста. Основными критериями выбора того или иного способа должны быть наглядность и максимальная простота программы, позволяющие быстро и удобно набрать и, главное, откорректировать программу, а также наибольшее удобство пользователя при работе с Вашей программой.

6.3. Подсчет количества элементов массива с заданными свойствами

При обработке массива числовых или символьных данных часто возникает необходимость подсчета общего количества элементов, либо части элементов, удовлетворяющих некоторым условиям. Для этой цели применяют классический прием программирования – использование переменной-счетчика. Например, из массива численных значений, включающих измеренные смещения кровли штрека, необходимо выбрать и подсчитать количество значений, находящихся в интервале от 50 до 100 мм.

Данную задачу целесообразно решать используя цикл, внутри которого поместить переменную-счетчик, увеличивающуюся каждый раз на 1 при выполнении заданного условия:

```
DIM A(300)
OPEN "A:\STRECK.DAT" FOR INPUT AS #1
FOR I=1 TO 300
  INPUT #1, A(I)
  IF A(I)=>50 AND A(I)<=100 THEN
     PRINT A(I),
     K=K+1
  END IF
NEXT I
CLOSE #1
```

Здесь из 300 числовых значений переменной A(I), считанных из файла STRECK.DAT, подсчитывается количество значений, удовлетворяющих условию $50 \le A(I) \le 100$. Для этой цели используется переменная-счетчик К. На примере данного фрагмента хорошо видно отличие операции присвоения от математического равенства величин. Действительно, если трактовать операцию К=К+1 как простое арифметическое выражение, то, сокращая на К, получаем бессмысленное равенство 0=1. На самом деле оператор присвоения К=К+1 следует интерпретировать следующим образом: значение переменной К увеличивается на 1, и полученный результат присваивается в качестве нового значения переменной К.

Иногда до начала цикла переменной-счетчику присваивают значение 0, однако данная операция может быть опущена (что и сделано в вышеприведенной программе), так как при обнаружении программой переменной, значение которой ранее не было определено, ей автоматически будет присвоено значение 0. Так при первом выполнении строки К=К+1, выполнится операция К=0+1=1, т.е. переменной К будет присвоено значение 1, при втором – К=1+1=2 и т.д.

Остается дополнить, что увеличение значения переменной К на единицу и вывод значений A(I) на печать происходят только в том случае, когда одновременно выполняются условия $A(I) \ge 50$ и $A(I) \le 100$. Таким образом, установив соответствующие условия, можно подсчитывать количество удовлетворяющим им значений в массиве. Такой прием широко применяется при обработке массивов числовых или символьных данных.

6.4. Определение суммы и произведения элементов массива

Тот же принцип переменной-счетчика используется для нахождения суммы элементов массива, однако в данном случае каждому последующему значению суммы прибавляется не единица, как в предыдущем примере, а величина значения текущего элемента массива.

Вычисление суммы рассмотрим на примере расчета суммарной трудоемкости горнопроходческого цикла:

```
CLS
INPUT "Введите количество процессов"; N
FOR I = 1 TO N
   LOCATE 2, 15
   PRINT "Введите трудоемкость"; I; "-го процесса";
   INPUT TR
   LOCATE I + 3, 20
   PRINT USING "#  ##.##"; I; TR
   S = S + TR
NEXT I
PRINT USING "Трудоемкость цикла - ###.## чел·ч"; S
```

Здесь суммирование всех вводимых значений осуществляется строкой S=S+TR, т.е. каждому последующему значению S присваивается значение, большее на величину TR. Результат выполнения данного фрагмента программы:

```
               1     3.45
               2     6.78
               3     2.78
               4     8.21
               5     1.35
               6     1.19
               7     2.79
Трудоемкость цикла  - 26.55 чел·ч
```

Аналогично сумме может быть рассчитано и произведение элементов массива, однако, здесь до начала цикла необходимо задать первоначальное значение произведения, равное 1. В противном случае использование данного приема при любых количествах и значениях перемножаемых переменных приведет к тому, что произведение останется равным нулю.

Рассмотрим пример расчета факториала произвольного числа. С такой задачей часто приходится сталкиваться в комбинаторике и теории вероятностей.

```
CLS
P = 1
INPUT "Введите N"; N
FOR I = 1 TO N
   P = P * I
NEXT I
PRINT I - 1; "! ="; P
```

Результат выполнения данной программы (при N=8):

```
8 ! = 40320
```

Как видно из текста программы, здесь переменная I является изменяемым сомножителем, а P – промежуточным произведением, которое в результате выполнения последнего цикла принимает искомое значение факториала заданного числа.

6.5. Определение максимального и минимального элементов массива

Данная задача часто возникает в инженерных расчетах, когда, например, из целого ряда возможных технических решений необходимо выбрать одно, обеспечивающее максимальные технико-экономические показатели или определить фактор, оказывающий максимальное влияние на исследуемый процесс или явление. Будем считать, что в результате решения некоторой задачи или проведения эксперимента сформирован массив числовых данных.

Для нахождения максимального или минимального элемента массива используют другой типовой прием программирования, сущность которого заключается в следующем. Первоначально переменной YMAX, определяющей максимальное значение, задается некоторое числовое значение, заведомо меньшее, чем предполагаемые величины элементов массива. Затем внутри цикла с помощью условного оператора производится поочередное сравнение всех элементов с YMAX. Если текущее значение элемента окажется больше, чем YMAX, то переменной YMAX присваивается значение сравниваемой переменной, в противном случае – величина YMAX не изменяется. В результате перебора всех элементов переменной YMAX будет присвоено максимальное значение массива.

Аналогично находят и минимальное значение элемента.

Рассмотрим фрагмент программы, определяющий максимальный и минимальный элемент массива (предполагается, что все значения элементов массива находятся в пределах от 1 до 100).

```
CLS
DIM Y(20)
YMAX = 0
YMIN = 100
FOR I = 1 TO 20
   READ Y(I)
   IF Y(I) > YMAX THEN YMAX = Y(I)
   IF Y(I) < YMIN THEN YMIN = Y(I)
NEXT I
PRINT USING "YMAX=##  YMIN=##"; YMAX; YMIN
DATA 34,45,67,92,24,35,66,11,44,88,65,32,16,65,21,54,32,65,43,22
```

Результат выполнения данной программы:

```
YMAX=92    YMIN=11
```

При нахождении максимального и минимального элементов массива **необходимо обратить внимание на следующее**: в вышеприведенной программе первоначальные значения YMIN=100 и YMAX=0 были заданы произвольно, в расчете на то, что в массиве найдутся числа, большие, чем YMAX и меньшие, чем YMIN. Однако, если бы значения всех элементов оказались больше 100, мы получили бы неверный результат YMIN=100. Во избежание такой ошибки (особенно, когда заранее неизвестны ориентировочные значения рассчитываемых величин) целесообразнее в качестве начальных значений YMAX и YMIN принимать не произвольные числа, а значение первого элемента массива, которое необходимо рассчитать (определить) до начала цикла, производящего выбор максимального и минимального значений, а сам цикл начать со второго элемента. В этом случае начало программы будет иметь вид:

```
DIM Y(20)
READ Y(1)
YMAX = Y(1)
YMIN = Y(1)
FOR I = 2 TO 20
. . .
```

и далее аналогично вышеприведенному варианту.

6.6. Лабораторная работа №4

Тема. Использование типовых приемов программирования при решении задач шахтного строительства.

Цель работы. Научиться основным приемам программирования, приобрести навыки составления алгоритмов и программ смешанной структуры применительно к задачам шахтного строительства.

Задание. Согласно варианту задания составить блок-схему алгоритма и программу на языке QBasic по расчету заданных параметров. Программа должна обеспечивать расчет для любых вариантов исходных данных. Предусмотреть возможность вывода результатов выполнения программы на экран и в файл.

6.6.1. Варианты лабораторной работы

Вариант 1

Определить необходимое число шпуров для взрывания породного забоя, определяемое формулой

$$N = 12{,}7 \cdot \frac{q \cdot S \cdot \eta}{\gamma \cdot d^2 \cdot \rho},$$

где q – удельный расход ВВ,

$$q = q_1 \cdot f_1 \cdot v \cdot e;$$

q_1 – нормальный удельный расход ВВ, принимаемый равным $q_1 = 0{,}1 \cdot f$;

f – крепость взрываемых пород по шкале проф. Протодьяконова;

f_1 – коэффициент структуры породы, $f_1 = 1{,}3 \div 1{,}4$;

v – коэффициент зажима породы, $v = \dfrac{6{,}5}{\sqrt{S}}$;

S – площадь поперечного сечения выработки, м^2;

η – коэффициент использования шпура, $\eta = 0{,}8 \div 0{,}9$;

γ – коэффициент заполнения шпура, $\gamma = 0{,}4 \div 0{,}5$;

d – диаметр патрона ВВ, см;

ρ – гравиметрическая плотность ВВ в патронах, г/см^3, принимаемая согласно приведенной таблицы в зависимости от типа ВВ;

e – коэффициент работоспособности ВВ, принимаемый согласно следующей таблице в зависимости от типа ВВ:

Тип ВВ	Коэффициент e	ρ, г/см^3
Аммонит скальный №1	0,8	1,10
Детонит М	0,82	1,15
Аммонит АП-5ЖВ	1,17	1,15
Аммонит № 6ЖВ	1,00	1,20

Организовать ручной ввод следующих исходных данных: типа ВВ, крепости пород, площади взрываемого забоя, КИШ и диаметра патрона ВВ. На печать вывести наименование выбранного пользователем ВВ, его основные характеристики и рассчитанное число шпуров на забой.

Вариант 2

По заданной производственной мощности шахты определить нормативную продолжительность подготовительного периода строительства согласно приведенной таблице.

Производственная мощность шахты, млн. т / год	0,6	0,9	1,2	1,5	1,8	2,1	2,4	3,0	3,6
Нормативная продолжительность подготовительного периода, мес.	12	12	15	15	15	16	16	16	16

Для случая задания пользователем производственной мощности, отличной от типовой, предусмотреть выдачу сообщения о некорректности исходных данных и возможность повторного выбора производственной мощности шахты.

Вариант 3

Проверить соблюдение запаса прочности каната подъемной машины по условию:

$$Z_g = \frac{\Sigma F_p}{Q_г + Q_c + m_k \cdot H_0} \geq Z_1,$$

где Z_1 – запас прочности каната, принять $Z_1 = 10$;

ΣF_p – суммарное разрывное усилие всех проволок в канате, Н;

$Q_г$ – масса породы и воды в бадье, кг,

$$Q_г = V_б \cdot \gamma_п + \left(V_б - \frac{V_б}{k_р} \right) \cdot \gamma_в \cdot k_з ;$$

$V_б$ – вместимость бадьи, м^3;

$\gamma_п$ – плотность породы в разрыхленном состоянии, кг/м^3;

$k_р$ – коэффициент разрыхления породы, $k_р = 1{,}8 \div 2$;

$\gamma_в$ – плотность воды, кг/м^3;

$k_з$ – коэффициент заполнения пустот водой, $k_з = 0{,}5$;

$m_к$ – масса 1м каната, определяемая из соотношения,кг,

$$m_к = \frac{Q_г + Q_c}{\frac{\sigma}{z_1 \cdot \gamma_0} - H_0};$$

σ – предел прочности материала проволок каната при растяжении, МПа, $\sigma = 1500$;

γ_0 – фиктивный объемный вес материала каната, кН/м^3, $\gamma_0 = 90$;

H_0 – максимальная длина отвеса каната, м, $H_0 = H_{ст} + h_к$;

$H_{ст}$ – конечная глубина ствола, м;

$h_к$ – высота копра, м;

Q_c – масса бадьи с направляющей рамой и прицепным устройством, кг, определяемая по следующей таблице в зависимости от типа бадьи, $Q_c = Q_б + Q_{н.р} + Q_{п.у}$:

Тип бадьи	Вместимость V, м^3	Масса бадьи $Q_б$, кг	Масса направляющей рамы $Q_{н.р}$, кг	Масса прицепного устройства $Q_{п.у}$, кг
БПС-1	1,0	356	394	92
БПС-1,5	1,5	605	590	92
БПС-2	2,0	730	590	92
БПС-2,5	2,5	878	600	97
БПС-3	3,0	938	600	97
БПС-4	4,0	1465	835	97
БПС-5	5,0	1696	1000	118

Расчет произвести для следующих данных: $\Sigma F_р$ = 268000 Н, $H_{ст}$ = 450 м, $h_к$ = 25 м, $\gamma_п$ = 1600 кг/м^3. Предусмотреть ручной ввод типа применяемой бадьи. На печать вывести тип бадьи, концевую нагрузку, величину запаса прочности и сообщение о соблюдении или несоблюдении требуемого запаса прочности.

Вариант 4

Произвести выбор вместимости проходческой бадьи $V_б$, м^3, по продолжительности полного цикла подъема бадьи по стволу и производительности подъемной установки, используя формулу

$$V_б = \frac{P_п \cdot t_ц}{3600 \cdot k_б},$$

где $P_п$ – необходимая производительность подъемной установки, м^3/ч, $P_п = min\{P_{п1}; P_{п2}\}$;

$P_{п1}$ – производительность подъемной установки, определяемая исходя из заданной технической скорости проходки ствола,

$$P_{п1} = \frac{v_т \cdot S_{пр} \cdot k_н \cdot k_р}{t \cdot m};$$

$v_т$ – заданная скорость проходки ствола, м/мес;

$S_{пр}$ – площадь сечения ствола в проходке, м^2, $S_{пр} = \frac{\pi \cdot D_{пр}^2}{4}$;

$k_н$ – коэффициент неравномерности работы подъема, $k_н$ = 1,15;

$k_р$ – коэффициент разрыхления породы, $k_р$ = 1,8 ÷ 2;

t – продолжительность работы подъема в сутки по выдаче породы, ч;

m – число рабочих дней в месяце по проходке ствола;

$P_{п2}$ – производительность подъемных установок, определяемая исходя из возможной производительности погрузочных машин в забое, м^3/ч, $P_{п2} = 1{,}15 \cdot P_р$;

$t_ц$ – продолжительность полного цикла подъема бадьи по стволу, с;

$k_б$ – коэффициент заполнения бадьи, $k_б$ = 0,9.

По расчетной вместимости бадьи принять ближайшую большую вместимость бадьи из следующего ряда значений:

0,5; 0,75; 1,0; 1,5; 2,0; 2,5; 3,0; 4,0; 5,0; 6,5

Расчет произвести для следующих исходных данных: $t_ц$ = 400 с; t = 14 ч; m = 25. Предусмотреть ручной ввод параметров $v_т$, $D_{пр}$ и $P_р$. На печать вывести расчетное и принятое из типового ряда значения вместимости бадьи и значения производительности подъема $P_{п1}$ и $P_{п2}$.

Вариант 5

Рассчитать производительность погрузочной машины Q, м^3/ч, при различных технологических схемах транспортирования горной массы из забоя и определить схему, при которой производительность будет максимальной:

$$Q = max\ \{Q_в, Q_п, Q_к\},$$

где $Q_в$, $Q_п$, $Q_к$ – производительность погрузочной машины соответственно при погрузке породы в одиночные вагонетки, при применении перегружателей и при транспортировании породы конвейером, м^3/ч. Данные параметры определяются следующими формулами:

$$Q_в = \frac{1}{\varphi \cdot \alpha \cdot \left(\frac{1}{Q_т} + \frac{t_з}{V_в \cdot k_з} \right) + \frac{(1-\alpha) \cdot \beta \cdot \varphi}{n_р \cdot P_п}};$$

$$Q_п = \frac{1}{\varphi \cdot \alpha \cdot \left(\frac{1}{Q_т} + \frac{t_с}{V_в \cdot k_з \cdot n_с} \right) + \frac{(1-\alpha) \cdot \beta \cdot \varphi}{n_р \cdot P_п}};$$

$$Q_к = \frac{1}{\frac{\varphi \cdot \alpha}{Q_т} + \frac{(1-\alpha) \cdot \beta \cdot \varphi}{n_р \cdot P_п}},$$

где φ – коэффициент, учитывающий технологические простои машины, φ = 1,15÷1,20;

α – доля объема породы первой фазы, α = 0,85÷0,90;

$Q_т$ – техническая производительность погрузочной машины, м^3/ч;

$V_в$ – объем вагонетки, м^3;

$k_з$ – коэффициент заполнения вагонетки, $k_з$ = 0,9;

$t_з$ – время замены груженой вагонетки на порожнюю, ч;

$t_с$ – время замены груженой партии вагонеток на порожнюю, ч;

$n_с$ – число вагонеток, установленных под перегружателем;

$n_р$ – число рабочих, занятых на подкидке породы;

$P_п$ – производительность рабочего на подкидке породы, м^3/ч, принимаемая по следующей таблице в зависимости от крепости пород:

f	< 4	4÷7	>7
$P_п$, м^3/ч	1,1	0,9	0,8

Расчет выполнить для следующих исходных данных: $t_з$ = 0,02 ч; $t_с$ = 0,03 ч; $n_с$ = 5; $n_р$ = 2. Предусмотреть ручной ввод параметров $Q_т$, $V_в$, f. На печать вывести три значения производительности и сообщение о способе транспортировки породы, при котором получена максимальная производительность.

Вариант 6

Сравнить сменную производительность скреперной установки, при разгрузке горной массы на конвейер $Q_к$ и в одиночные вагонетки $Q_в$ и выбрать схему транспорта, обеспечивающую максимальную производительность Q, м^3/смен:

$$Q = max\ \{ Q_к, Q_в \},$$

где $Q_к$ и $Q_в$ определяются формулами:

$$Q_к = \frac{(T - t_{пз} - t_л) \cdot V_в \cdot k_{з.с}}{\left(\frac{l}{v_г} + \frac{l}{v_п} + t_{з.р} \right) \cdot k_р},$$

$$Q_в = \frac{(T - t_{пз} - t_л) \cdot V_в \cdot k_{з.в}}{\frac{V_в \cdot k_{з.в}}{V \cdot k_р} \cdot \left(\frac{l}{v_г} + \frac{l}{v_п} + t_{з.р} \right) + \frac{2 \cdot L}{v_в} + t_р};$$

T – продолжительность смены, мин, $T = 360$;

$t_{п.з}$ – продолжительность подготовительно-заключительных операций, мин, $t_{п.з} = 40$;

$t_л$ – продолжительность опробования лебедки, мин, $t_л = 10$;

V – объем скрепера, м^3;

$k_{з.с}$ – коэффициент заполнения скрепера, принимаемый равным: для крупнокусковатой горной массы – 0,5÷0,7, для средней – 0,7÷0,8, для мелкой – 0,8÷1;

l – расстояние скреперования, м;

$v_г$ и $v_п$ – соответственно скорости движения груженого и порожнего скрепера, м/мин;

$t_{з.р}$ – время загрузки и разгрузки скрепера с учетом пауз на переключение и неравномерности хода, мин;

$k_р$ – коэффициент разрыхления породы, $k_р$ =1,8÷2;

$V_в$ – объем вагонетки, м^3;

$k_{з.в}$ – коэффициент заполнения вагонетки, $k_{з.в} = 0,9$;

L – длина транспортировки породы в вагонетке, м;

$v_в$ – скорость движения вагонетки, м/мин;

$t_р$ – время разгрузки вагонетки, мин.

Расчет произвести для следующих исходных данных: l = 30 м, $v_г$ = 66 м/мин, $v_п$ = 90 м/мин, $t_{з.р}$ = 0,7 мин, L = 50 м, $v_в$ = 120 м/мин, $t_р$ = 1 мин. Организовать ручной ввод параметров V, $V_в$ и степень кусковатости горной массы (крупная, средняя, мелкая). На печать вывести значения производительности для обеих схем транспортирования горной массы и сообщение о более эффективной схеме.

Вариант 7

Определить необходимое количество воздуха $Q_{з.п}$, м^3/мин, для проветривания тупиковой выработки по всем указанным факторам. В качестве расчетного принять максимальное из полученных значений

$$Q_{з.п} = max\ \{Q_1, Q_2, Q_3, Q_4, Q_5\},$$

где Q_1 – расход воздуха в забое по выделению метана, м^3/мин,

$$Q_1 = \frac{100 \cdot I_{з.п.}}{C - C_0};$$

$I_{з.п}$ – суммарное метановыделение с обнаженной поверхности угольного пласта и из отбитого угля, м^3/мин;

C – максимально допустимое по ПБ содержание метана в исходящей вентиляционной струе, %, $C = 1$;

C_0 – содержание метана в поступающей вентиляционной струе, %;

Q_2 – расход воздуха в забое по максимальному числу людей, одновременно работающих в забое выработки, м^3/мин,

$$Q_2 = 6 \cdot n_л;$$

$n_л$ – максимальное число людей, одновременно работающих в забое выработки;

Q_3 – расход воздуха в забое по минимальной скорости движения воздуха, м^3/мин,

$$Q_3 = 60 \cdot V_{min} \cdot S_{св};$$

V_{min} – минимально допустимая скорость движения воздуха в призабойном пространстве, м/с, $V_{min} = 0{,}15$;

$S_{св}$ – площадь поперечного сечения выработки в свету, м^2;

Q_4 – расход воздуха в забое по тепловому фактору, м^3/мин,

$$Q_4 = 20 \cdot V_{n.min} \cdot S_{св};$$

$V_{n.min}$– минимально допустимая скорость движения воздуха в выработке по тепловому фактору, м/с;

Q_5 – расход воздуха в забое по количеству одновременно взрываемого ВВ, м^3/мин,

$$Q_5 = \frac{2{,}25}{T} \cdot \sqrt[3]{\frac{V_{ВВ} \cdot S_{св}^2 \cdot l_п^2 \cdot K_{обв}}{K_{ут.тр.}^2}};$$

T – время проветривания выработки после взрывания, мин;

$V_{ВВ}$ – объем вредных газов, образующихся после взрывания, л,

$$V_{ВВ} = 100 \cdot B_{уг} + 40 \cdot B_{пор};$$

$B_{уг}$, $B_{пор}$ – масса одновременно взрываемых ВВ по углю и породе, соответственно, кг;

$l_п$ – длина тупиковой выработки, м, для горизонтальных и наклонных выработок протяженностью 500 м и более вместо $l_п$ подставляется ее критическая длина $l_{кр}$ = 500;

$K_{обв}$ – коэффициент, учитывающий обводненность выработки и принимаемый равным 0,8 – при проведении выработки по сухим породам, 0,6 – по влажным и 0,3 – по обводненным или с применением водяных завес;

$K_{ут.тр}$ – коэффициент утечек воздуха в вентиляционных трубопроводах.

Расчет произвести для следующих исходных данных: $I_{з.п}$ = 1 м^3/мин, C_0 = 0,05%, $n_л$ = 6 чел., $V_{п.min}$ = 0,5 м/с, T = 20 мин, $K_{ут.тр}$= 1,41.

Организовать ручной ввод параметров $l_п$, $S_{св}$, $B_{уг}$, $B_{пор}$, степени обводненности проходимой выработки (сухая, влажная, обводненная). На печать вывести значения расхода воздуха по всем факторам, принятое значение $Q_{з.п}$ и сообщение об определяющем факторе.

Вариант 8

1. Рассчитать толщину $B_п$, м, и количество ступеней N тампонажной подушки, сооружаемой при цементации пород из забоя ствола, если

$$B_п = \frac{\lambda \cdot P \cdot \left(r_{пр}^2 + \eta^2\right)^2}{4 \cdot r_{пр}^2 \cdot \eta \cdot m \cdot R_б},$$

где λ – коэффициент перегрузки, λ = 1,1÷1,2;

P – давление нагнетания цементного раствора, МПа, $P = P_г + P_и$;

$P_г$ – гидростатический напор подземных вод в интервале цементации, МПа;

$P_и$ – избыточное давление цементного раствора, МПа;

$r_{пр}$ – радиус ствола в проходке, м;

η – высота сферической поверхности тампонажной подушки, м;

m – коэффициент условий работы, m = 0,7÷0,8;

$R_б$ – расчетное сопротивление бетона в раннем возрасте на сжатие, МПа, $R_б = n_б \cdot R_{28}$;

$n_б$ – коэффициент относительной прочности бетона в раннем возрасте;

R_{28} – расчетное сопротивление бетона на сжатие в 28-суточном возрасте, МПа.

Расчет произвести для следующих исходных данных: $P_г$ = 3 МПа; $P_и$ = 5 МПа; η = 1,5 м; $n_б$ = 0,82; R_{28} = 20 МПа. Значение параметра $r_{пр}$ задать самостоятельно.

Если расчетное значение толщины подушки $B_п$ > 2,5 м, необходимо предусмотреть многоступенчатую тампонажную подушку, число ступеней которой определить из выражения

$$N = \frac{B_n}{B_c},$$

где B_c – толщина одной ступени, принимаемая равной 1,5÷2,5 м. Если значение N получается дробным, его необходимо округлить до ближайшего большего целого.

2. Для рассчитанной толщины и конструкции подушки определить напряжение сжатия бетона σ_c, МПа, по формуле:

$$\sigma_c = \frac{0{,}01 \cdot \lambda \cdot P \cdot \left(r^2 + \eta^2\right)^2}{4 \cdot r^2 \cdot \eta \cdot m \cdot N \cdot B_c},$$

где r – радиус ствола в свету, м;

Проверить конструкцию по условию прочности $\sigma_c \le R_б$.

На печать вывести значения толщины тампонажной подушки, количество ступеней, напряжение сжатия бетона и сообщение о выполнении (невыполнении) условия прочности. В случае невыполнения условия прочности предусмотреть возможность изменения параметров R_{28} или $n_б$.

Вариант 9

Рассчитать приближенную длину цементационной заходки l, м, при нагнетании тампонажного раствора с земной поверхности, используя уравнение

$$g \cdot \left(\gamma_р - \gamma_в\right) \cdot l^2 + 2 \cdot \left[h_н + g \cdot \gamma_р \cdot h_{ст} + g \cdot \left(\gamma_р - \gamma_в\right) \cdot h_п\right] \cdot l - 2 \cdot K \cdot \frac{Q_{ск}}{q} = 0,$$

где $\gamma_р$, $\gamma_в$ – плотность соответственно раствора и воды, кг/м^3;

$h_{ст}$ – расстояние от статического уровня подземных вод до манометра на скважине, м;

$h_п$ – расстояние от кровли водоносного горизонта до статического уровня подземных вод, м;

$Q_{ск}$ – поглощающая способность скважины, м^3/с;

K – коэффициент учета увеличения сопротивления скважины и трещин при переходе от течения в них воды к течению раствора;

q – удельное водопоглощение горных пород, м^3/(с·м·Па);

$h_н$ – начальное давления нагнетания, Па, определяемое по следующей таблице в зависимости от величины раскрытия трещин δ:

δ, мм	До 1	1 ÷ 5	5 ÷ 20	Более 20
$h_н$, Па	$7 \cdot 10^5$	$5 \cdot 10^5$	$3 \cdot 10^5$	$2 \cdot 10^5$

Расчет произвести для следующих исходных данных: $\gamma_р$ = 1150 кг/м^3,

$h_{ст}$ = 20 м, h_n = 10 м, $Q_{ск}$ = 0,15 м^3/с, K = 1,2, q = 1,2·10^{-7} м^3/(с·м·Па). Предусмотреть ручной ввод параметра δ. На печать вывести величину начального давления нагнетания и значение длины цементационной заходки.

Вариант 10

Произвести расчет пневматической сети участка шахты, используя выражение:

$$V = k_{ут} \cdot \psi \cdot \sum_{i=1}^{n} m_i \cdot q_i \cdot \xi_i ,$$

где $k_{ут}$ – коэффициент, учитывающий потери воздуха от утечек через неплотности трубопровода, $k_{ут}$ = 1,2;

ψ– коэффициент, учитывающий износ оборудования, ψ = 1,1;

n – число однотипных групп воздухоприемников;

m_i – число однотипных (i-х) потребителей;

q_i – расход воздуха одним (i-м) потребителем, м^3/мин;

ξ_i – коэффициент неравномерности работы потребителей, принимаемый равным согласно следующей таблице:

m_i	1	2÷4	5÷10	11÷30
ξ_i	1	0,95	0,90	0,80

Расчет произвести для следующих исходных данных:

i	Наименование воздухоприемников	m_i	q_i, м^3/мин
1	Перфоратор ПП-30	12	3,0
2	Оборудование для очистки шпуров	5	1,0
3	Пневмолом	5	1,2
4	Отбойный молоток МО-10	4	1,15
5	Стволовая погрузочная машина КС-2у/40	2	50,0
6	Насос забойный Н-1м	2	6,0
7	Пневмотельфер	1	8,0

Результаты выполнения программы вывести на печать в форме таблицы, содержащей столбцы исходных данных и столбец значений расхода воздуха по каждому виду оборудования. Ниже последнего столбца поместить значение общего расхода воздуха на участке шахты.

Вариант 11

По заданным значениям относительной влажности P, %, и скорости движения воздуха V, м/с, определить допустимую температуру воздуха T, °С, в забое, регламентируемую Правилами безопасности:

Скорость воздуха, м/с	Допустимая температура, °С, при относительной влажности, %		
	до 75	76 – 90	свыше 90
До 0,25	24	23	22
0,26 – 0,5	25	24	23
0,51 – 1,0	26	25	24
Более 1,0	26	26	25

Предусмотреть ручной ввод значений скорости движения и влажности воздуха. На печать вывести значения всех параметров.

Вариант 12

Проверить соответствие Правилам Безопасности содержания газов в шахтной атмосфере, руководствуясь следующими требованиями:

Газ	Допустимая концентрация газа, % (объемная доля)
Кислород	≥ 20
Углекислый газ: – в исходящих струях выемочных участков и тупиковых выработок; – в выработках с исходящей струей крыла, горизонта, шахты в целом; – при проведении и восстановлении выработок по завалу	 ≤ 0,5 ≤ 0,75 ≤ 1
Водород	≤ 0,5
Окись углерода (IV)	≤ 0,0017
Окислы азота	≤ 0,00026
Сернистый ангидрид	≤ 0,00038
Сероводород	≤ 0,00071

Предусмотреть возможность выбора газа и при необходимости места замера концентрации. Организовать ручной ввод значений измеренных концентраций. На печать вывести наименование выбранного газа (газов), значение измеренной концентрации и сообщение о превышении (непревышении) предельно допустимой концентрации.

Вариант 13

Определить категорию шахты по метану, присваиваемую в зависимости от относительной газообильности q_i, м^3/т, и вида выделения метана согласно следующей таблице:

Категория шахты по метану	Относительная метанообильность, м^3/т
I	До 5
II	От 5 до 10
III	От 10 до 15
Сверхкатегорные	15 и более; шахты, опасные по суфлярным выделениям

Относительную метанообильность определить по формуле

$$q_i = \frac{1440 \cdot \sum_{i=1}^{n} I_i \cdot N_i}{\sum_{i=1}^{n} A_i},$$

где n – число месяцев работы объекта в году;

I_i – расход газа на шахте в i-м месяце, м^3/мин;

N_i – число фактически отработанных дней в месяц по добыче угля;

A_i – добыча на объекте за каждый месяц в истекшем году, т.

Исходные данные для расчета принять согласно следующей таблице:

i	1	2	3	4	5	6	7	8	9	10	11	12
I_i, м^3/мин	6,5	6,8	7,6	8,0	7,9	7,1	7,0	7,0	6,6	6,9	6,3	6,0
N_i	26	26	29	29	27	29	30	29	29	25	27	30
A_i, тыс. т	57,5	54,2	61,0	60,5	52,6	55,0	61,5	60,0	52,0	41,5	49,5	66,0

На печать вывести относительную метанообильность шахты и сообщение о присвоении соответствующей категории по метану.

Вариант 14

Рассчитать толщину и произвести выбор (монолитный бетон или набрызгбетон) крепи вертикальной выработки $\delta_к$, мм, при использовании бетонов различных классов по формуле

$$\delta_к = m_y \cdot r_{св} \cdot \left(\sqrt{\frac{0{,}85 \cdot R_{пр}}{0{,}85 \cdot R_{пр} - 2 \cdot K_p \cdot P}} - 1 \right) - \delta_{пб},$$

где $r_{св}$ – радиус вертикальной выработки в свету, мм;

m_y – коэффициент условий работы крепи, $m_y = 1{,}25$;

K_p – коэффициент концентрации напряжений в конструкции крепи, $K_p = 1$ на протяженных участках ствола (при $z > 20$ м) и $K_p = 2 - 0{,}05 \cdot z$ в районе сопряжений;

z – расстояние от узла сопряжения до рассматриваемого сечения, м;

P – горизонтальное давление на крепь ствола, МПа;

$\delta_{пб}$ – толщина породобетонной оболочки, образующейся за счет проникновения бетона в окружающие нарушенные породы, принимаемая равной 500 мм для набрызгбетона и нулю для монолитной бетонной крепи;

$R_{пр}$ – расчетное сопротивление бетона сжатию, МПа, принимаемое по следующей таблице в зависимости от класса бетона:

Класс бетона	B10	B15	B20	B25	B30	B40	B50
$R_{пр}$, МПа	4,6	7,0	9,0	11,0	13,5	17,5	21,5

Расчет произвести для всех классов бетона, приведенных в таблице. Организовать ручной ввод параметров $r_{св}$, P, z, а также предполагаемого типа крепи (монолитная бетонная или набрызгбетонная). При составлении программы следует учесть, что толщина набрызгбетонной крепи не должна превышать 150 мм. Результаты выполнения программы вывести на печать в форме таблицы, содержащей классы бетона и соответствующие им значения толщины и типа крепи.

Вариант 15

Рассчитать лобовую горизонтальную силу $P^{л}$, Н, действующую на проводник в динамической системе "подъемный сосуд – армировка", используя формулу:

$$P^{л} = \frac{2 \cdot \pi^2 \cdot \delta \cdot K_p^2 \cdot m \cdot V^2}{h^2} \cdot n^{л},$$

где K_p – коэффициент влияния типа рабочих направляющих подъемного сосуда, равный 1 – при жестких направляющих скольжения (принять для рельсовых и деревянных проводников) и 0,85 – при упругих роликовых направляющих (принять для коробчатых проводников);

m, V – соответственно масса, кг, и скорость, м/с, груженого подъемного сосуда;

$n^{л}$ – коэффициент, учитывающий жесткость проводников и эксцентриситет центра масс груженого сосуда, принять равным для рельсовых проводников 1,25, для коробчатых 1,15, для деревянных 1,35;

h – шаг армировки, м;

δ – зазор на сторону между рабочими или предохранительными направляющими скольжения и проводником, м, принимаемый по таблице в зависимости от типа проводников:

Тип проводников	Рельсовые	Коробчатые	Деревянные
δ, м	0,010	0,015	0,020

Расчет произвести для следующих значений шага армировки h, м:

– для рельсовых проводников – 3,126; 4,168; 6,252;

– для коробчатых проводников – 3; 4; 5; 6;

– для деревянных проводников – 2; 3; 4.

Предусмотреть ручной ввод массы и скорости движения груженого подъемного сосуда. Результаты выполнения программы вывести на печать в виде таблицы, содержащей тип проводника, шаг армировки и соответствующее им значение горизонтальной силы на проводник.

Примеры выполнения лабораторной работы

Пример 1. По заданным значениям коэффициента крепости *f* и площади поперечного сечения вертикального ствола в проходке *S*, м^2, определить расход *Q*, кг, скального аммонита №1 на 100 м^3 взорванной породы в массиве, если значения *Q* заданы следующей таблицей:

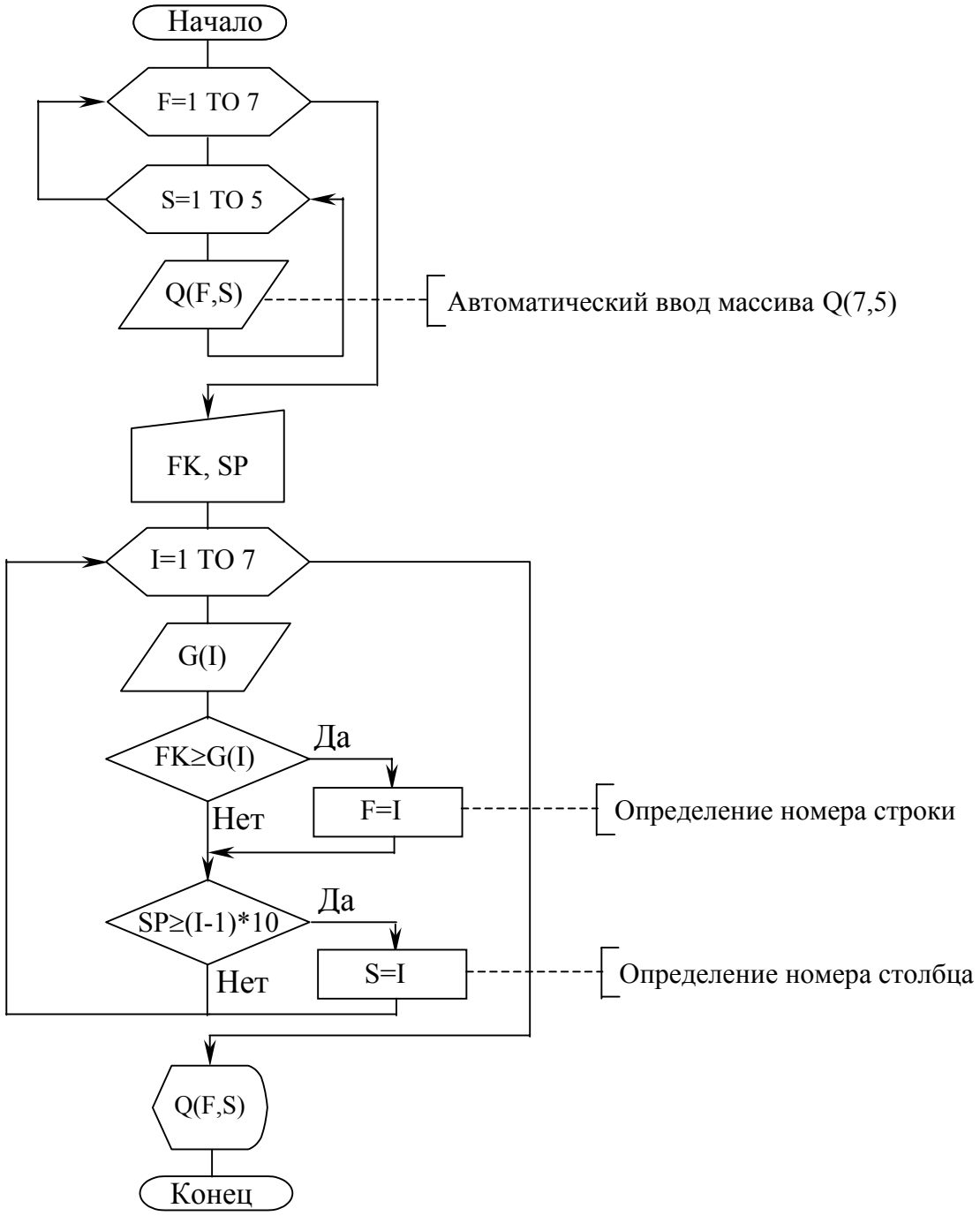
Коэффициент крепости пород *f*	Расход скального аммонита №1 на 100 м^3 взорванной породы в массиве, кг, при площади поперечного сечения ствола в проходке *S*, м^2				
	≤10	≤20	≤30	≤40	>40
1,5	137	124	108	91	52
2–3	182	170	143	115	90
4–6	213	200	170	140	120
7–9	247	235	205	175	150
10–14	311	280	250	220	195
15–18	366	330	300	270	245
19–20	404	365	335	305	275

Предусмотреть ручной ввод значений *f* и *S*. На печать вывести принятое значение удельного расхода ВВ.

Присваиваем имена переменным и составляем таблицу идентификаторов:

Переменная	Идентификатор	Примечание
S	SP	Площадь поперечного сечения ствола, м^2 (задается вручную)
f	FK	Крепость пород (задается вручную)
-	F	Номер строки, определяющий величину удельного заряда в зависимости от крепости
-	S	Номер столбца, определяющий величину удельного заряда в зависимости от площади
Q	Q(F,S)	Величина заряда (элемент массива Q)
-	I	Переменная-счетчик
-	G(I)	Нижняя граница каждого из интервалов крепости
-	A	Переменная, определяющая место вывода результатов на печать (на экран или в файл)

Составляем блок-схему алгоритма:



Программа на QBasic:

```
CLS
DIM Q(7,5)
DATA 137, 124, 108, 91, 52
DATA 182, 170, 143, 115, 90
DATA 213, 200, 170, 140, 120
DATA 247, 235, 205, 175, 150
DATA 311, 280, 250, 220, 195
DATA 366, 330, 300, 270, 245
DATA 404, 365, 335, 305, 275
```

```
DATA 1.5, 2, 4, 7, 10, 15, 19
FOR F = 1 TO 7
   FOR S = 1 TO 5
     READ Q(F, S)
NEXT S, F
INPUT "Введите крепость пород"; FK
INPUT "Введите площадь сечения ствола в проходке, м кв."; SP
F = 1: S = 1
FOR I = 1 TO 7
   READ G(I)
   IF FK >= G(I) THEN F = I
   IF I < 6 AND SP > 10 * (I - 1) THEN S = I
NEXT I
INPUT "Результаты вывести: 1-на экран; 2 – в файл"; A
IF A=1 THEN
   PRINT USING "Расход скального аммонита - ### кг/ 100 м куб."; Q(F, S)
ELSE
   OPEN "C:\LINA\LAB4.TXT" FOR OUTPUT AS #1
   PRINT #1, USING "Расход скального аммонита - ### кг/ 100 м куб."; Q(F, S)
   CLOSE #1
END IF
```

Результат выполнения программы (при F = 7 и S = 28 м^2):

```
Расход скального аммонита - 205 кг/ 100 м куб.
```

Пример 2. Рассчитать среднюю стоимость строительства 1 м ствола $C_{ср}$, руб/м, определяемую формулой:

$$C_{c\text{р}} = \frac{\sum_{i=1}^{n} c_i \cdot h_i}{\sum_{i=1}^{n} h_i},$$

где c_i – стоимость 1 м выработки на i-м участке с определенным коэффициентом крепости пород, руб.; h_i – мощность i-го участка, м.

Исходные данные для расчета принять согласно следующей таблицы:

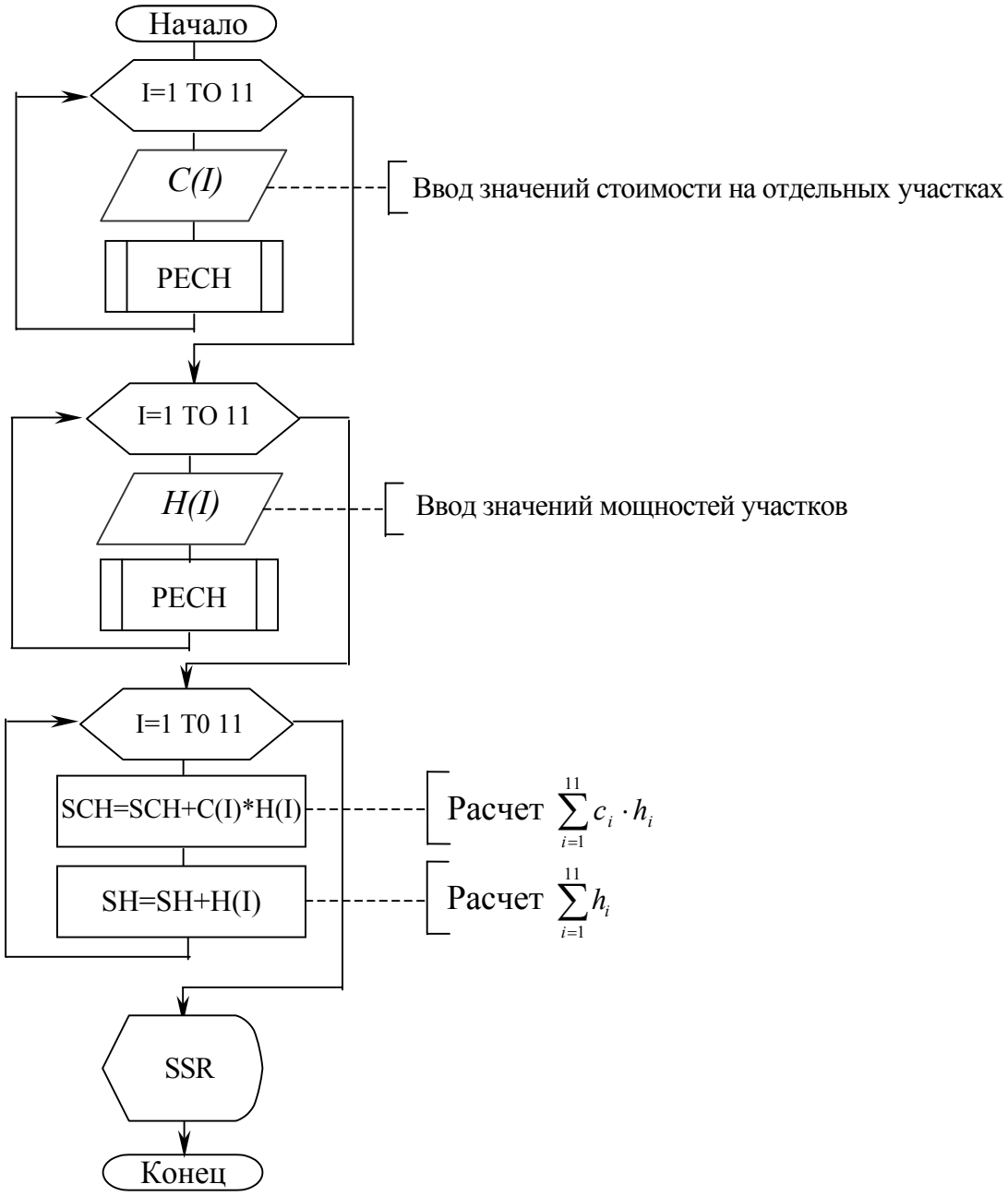
i	1	2	3	4	5	6	7	8	9	10	11
c_i, руб.	620	675	681	695	730	613	680	624	595	655	665
h_i, м	22	14	33	8	28	17	18	5	3	14	48

На печать (на экран или в файл – по указанию пользователя) вывести таблицу исходных данных и значение средней стоимости сооружения 1 м ствола.

Присваиваем имена переменным и составляем таблицу идентификаторов:

Переменная	Идентификатор	Примечание
i	I	Переменная-счетчик
c_i	C(I)	Элементы массива C(11) исходных данных
h_i	H(I)	Элементы массива H(11) исходных данных
	X(I)	Элементы текущего массива, выводимые на печать в подпрограмме
-	SCH	Сумма произведений $c_i \cdot h_i$ (числитель дроби)
-	SH	Сумма h_i (знаменатель дроби)
C_{cp}	SSR	Искомый параметр
-	A	Переменная, определяющая место вывода результатов на печать (на экран или в файл)

Составляем блок-схему алгоритма:



Программа на языке QBasic:

```
CLS
DIM C(11), H(11), X(11)
DATA 620, 675, 681, 695, 730, 613, 680, 624, 595, 655, 665
DATA 22, 14, 33, 8, 28, 17, 18, 6, 3, 14, 48
INPUT "Результаты вывести: 1-на экран, 2-в файл"; A
IF A = 2 THEN OPEN "A:\LAB4.TXT" FOR OUTPUT AS #1
FOR I = 1 TO 11
   READ C(I)
   X(I) = C(I)
   GOSUB PECH
NEXT I
FOR I = 1 TO 11
   READ H(I)
   X(I) = H(I)
   GOSUB PECH
NEXT I
FOR I = 1 TO 11
   SCH = SCH + C(I) * H(I)
   SH = SH + H(I)
NEXT I
SSR = SCH / SH
IF A = 1 THEN
   PRINT USING "Средняя стоимость 1 м ствола - ###.## руб."; SSR
ELSE
   PRINT #1, USING "Средняя стоимость 1 м ствола - ###.## руб."; SSR
   CLOSE #1
END IF
END

PECH:
IF A = 1 THEN
    PRINT USING " ### "; X(I);
    IF I = 11 THEN PRINT
ELSE
    PRINT #1, USING " ### "; X(I);
    IF I = 11 THEN PRINT #1,
END IF
RETURN
```

Результат выполнения программы:

```
620  675  681  695  730  613  680  624  595  655  665
 22   14   33    8   28   17   18    6    3   14   48
Средняя стоимость 1 м ствола - 667.50 руб.
```

Как видно, рассмотренный вариант, в котором исходные данные из соображений логичности и корректности построения программы разбиты на 2 отдельных массива, описывающих различные по смыслу данные, потребовал некоторого усложнения программы. Это связано, прежде всего, со сложностью организации печати двух разных массивов исходных данных, которая требует к тому же возможности выбора места вывода результатов (на экран или в файл). Чтобы сократить текст программы, можно несколько "пожертвовать" логичностью, объединив в одном массиве C(22) сразу все (различные по смыслу) исходные данные, присвоив при этом первым 11 элементам массива значения c_i, а последующим 11 – h_i. В этом случае программа будет иметь вид:

```
CLS
DIM C(22)
DATA 620, 675, 681, 695, 730, 613, 680, 624, 595, 655, 665
DATA 22, 14, 33, 8, 28, 17, 18, 6, 3, 14, 48
INPUT "Результаты вывести: 1-на экран, 2-в файл"; A
IF A = 2 THEN OPEN "A:\LAB4.TXT" FOR OUTPUT AS #1
FOR I = 1 TO 22
     READ C(I)
     IF A = 1 THEN
         PRINT USING " ### "; C(I);
         IF I = 11 OR I = 22 THEN PRINT
     ELSE
         PRINT #1, USING " ### "; C(I);
         IF I = 11 OR I = 22 THEN PRINT #1,
     END IF
NEXT I
FOR I = 1 TO 11
     SCH = SCH + C(I) * C(I + 11)
     SH = SH + C(I + 11)
NEXT I
SSR = SCH / SH
IF A = 1 THEN
     PRINT USING "Средняя стоимость 1 м ствола - ###.## руб."; SSR
ELSE
     PRINT #1, USING "Средняя стоимость 1 м ствола - ###.## руб."; SSR
     CLOSE #1
END IF
```

7. ПРОГРАММИРОВАНИЕ ГРАФИКИ НА QBASIC

В этой главе будут рассмотрены основы компьютерной графики, описаны некоторые операторы и приемы, позволяющие выводить на экран разнообразные графические изображения.

7.1. Режимы работы монитора. Установка режима и цветовых параметров

Мониторы компьютера могут работать в одном из двух режимов: текстовом и графическом. В **текстовом режиме** экран монитора условно разбивается на отдельные участки – знакоместа, чаще всего на 25 строк по 80 символов (знакомест). В каждое знакоместо может быть введен один из 256 заранее заданных символов. **Графический режим** монитора предназначен для вывода на экран графиков, рисунков и т. д. В этом режиме экран монитора состоит из точек. Для реализации этой технологии компьютер содержит в своем составе видеоадаптер, который хранит в своей памяти изображение. Наиболее широкое распространение в компьютере IBM PC получили адаптеры типов MDA, CGA, Hercules, EGA, VGA и Super-VGA. Количество точек по горизонтали и вертикали, обеспечиваемое адаптером, называется *разрешающей способностью монитора* в данном режиме.

QBasic поддерживает работу различных видеоадаптеров и допускает использование различных режимов работы экрана. Режим экрана устанавливается оператором **SCREEN <n>**, где n – код режима. Характеристики режимов работы монитора приведены в табл. 9.

Таблица 9

Режимы работы мониторов, устанавливаемые оператором SCREEN

Режим работы	Тип видеоадаптера	Текстовый режим	Графический режим
SCREEN 0	MDPA, CGA, EGA, VGA, SVGA	40×25, 40×43, 40×50, 80×25, 80×43, 80×50, 16 цветов (MDPA, CGA), 64 цвета (EGA, VGA, SVGA)	НЕТ
SCREEN 1	CGA, EGA, VGA, MCGA, SVGA	40×25, 16 цветов	320×200, 16 цветов
SCREEN 2	CGA, EGA, VGA, MCGA, SVGA	80×25, 2 цвета	640×200, 2 цвета
SCREEN 3	Hercules, монохромный монитор	80×25, 2 цвета	720×348, 2 цвета
SCREEN 4	Hercules, монохромный монитор	80×25, 2 цвета	640×400, 2 цвета
SCREEN 7	EGA, VGA, SVGA	80×25, 16 цветов	320×200, 16 цветов
SCREEN 8	EGA, VGA, SVGA	80×25, 16 цветов	640×200, 16 цветов
SCREEN 9	EGA, VGA, SVGA	80×25, 16 цветов	640×350, 16 цветов
SCREEN 10	EGA, VGA, SVGA, монохромный монитор	80×25, 80×43, 9 оттенков серого	640×350, 9 оттенков серого
SCREEN 11	VGA, MCGA, SVGA	80×30, 80×60, 256 цветов	640×480, 256 цветов
SCREEN 12	VGA, SVGA	80×30, 80×60, 256 цветов	640×480, 256 цветов
SCREEN 13	VGA, MCGA, SVGA	40×25, 256 цветов	320×200, 256 цветов

Наиболее часто используются режимы SCREEN 0, SCREEN 1 и SCREEN 7, доступные на большинстве мониторов. Однако наилучшее качество графических изображений обеспечивает режим SCREEN 12, обладающий самой высокой разрешающей способностью, но доступный только для видеоадаптеров типа VGA и SVGA.

При использовании оператора SCREEN **необходимо помнить следующее**:

– задаваемый режим работы должен поддерживаться используемым видеоадаптером, в противном случае возникает ошибка "Illegal function call" ("Неверный вызов функции");

– оператор SCREEN, устанавливающий графический режим монитора, должен стоять в программе до начала вывода любых графических изображений (основные графические операторы QBasic будут рассмотрены ниже), иначе возникает та же ошибка "Illegal function call";

– при использовании только текстового режима работы монитора оператор SCREEN 0 является необязательным, так как данный режим устанавливается по умолчанию;

– в одной программе может быть несколько допустимых операторов SCREEN, однако, при переключении режима монитора вся текстовая и графическая информация, выведенная в предыдущем режиме, будет стерта;

– в операторе SCREEN вместо численного значения, определяющего режим работы монитора, может стоять имя переменной, например: SCREEN RR. В этом случае к моменту использования оператора SCREEN переменная RR должна иметь определенное значение, допустимое для оператора SCREEN. Такая необходимость может возникнуть при использовании программы на компьютерах с различными видеоадаптерами. Нижеприведенный фрагмент программы демонстрирует выбор графического режима монитора в зависимости от применяемого видеоадаптера.

```
CLS
PRINT "Введите тип видеоадаптера:"
INPUT "1-CGA, 2-EGA, 3-VGA или SVGA"; TV
IF TV = 3 THEN
  RR = 12
ELSEIF TV = 2 THEN
  RR = 7
ELSE
  RR = 1
END IF
SCREEN RR
…
```

Такая организация установки режима работы монитора позволяет при составлении программы удобно переключаться с текстового режима (SCREEN 0) на графический (SCREEN RR) и обратно с учетом возможности использования различных типов видеоадаптеров. Однако в данном случае

при выводе графики необходимо учитывать несоответствия получаемых в разных режимах графических изображений, связанных с различной разрешающей способностью монитора в данных режимах. Это несоответствие легко устраняется введением поправочных коэффициентов к координатам точек, определяющим положение графических объектов на экране.

При работе в текстовом и графических режимах, предназначенных для цветных мониторов и видеоадаптеров, в программе на QBasic можно устанавливать цвет изображения и фона. Для этого используют оператор **COLOR**, который имеет следующий формат в зависимости от режима работы, устанавливаемого оператором SCREEN:

COLOR <N_1,N_2> – для режимов 0 и 7 – 10,

COLOR <N_1> – для режимов 4, 12, 13,

где N_1 – число, которое задает цвет текста и графических изображений (основной цвет);

N_2 – число, задающее цвет фона (фоновый цвет).

Соответствие цифровых атрибутов и воспроизводимых цветов при использовании перечисленных режимов приведено в табл. 10.

Таблица 10

Соответствие цифровых атрибутов и воспроизводимых цветов

Номер цвета	Воспроизводимый цвет	Номер цвета	Воспроизводимый цвет
0	Черный	8	Серый
1	Синий	9	Ярко синий
2	Зеленый	10	Светло-зеленый
3	Голубой	11	Светло-голубой
4	Красный	12	Светло-красный
5	Фиолетовый	13	Светло-фиолетовый
6	Коричневый	14	Желтый
7	Белый	15	Яркий белый

При использовании графического режима SCREEN 1 оператор COLOR имеет особый синтаксис:

COLOR <N1,N2>

где N_1 – номер цвета в палитре;

N_2 – номер палитры (0 или 1), определяющий, какой из двух наборов цветов используется. Пример использования оператора COLOR:

```
SCREEN 7
COLOR 15, 1
LOCATE 12, 10
PRINT "Шахтное строительство"
```

В результате выполнения данного фрагмента программы в центре экрана на синем фоне белыми буквами будут напечатаны слова "Шахтное строительство".

А следующий фрагмент программы позволяет вывести на печать столбец чисел от 0 до 15, причем каждое числовое значение имеет тот цвет, код которого оно задает:

```
SCREEN 7
FOR I=1 TO 15
  COLOR I
  PRINT I
NEXT I
```

При установке атрибутов цвета **необходимо помнить следующее**:

– использование оператора COLOR в режимах SCREEN 2, 3 и 11 недопустимо;

– в текстовом режиме SCREEN 0 номер основного цвета может находиться в пределах от 0 до 31, причем номера с 16 по 31 обеспечивают вывод на экран мерцающего текста. Мерцающий вариант цвета можно выбрать путем добавления 16 к основному цвету. Например, номер мерцающего света 7 равен 23 (7+16). Мерцание фонового цвета не поддерживается, поэтому значение цвета фона не должно превышать 15. Попытка задать значение основного цвета большее, чем 31, а фонового – большее, чем 15, приводит к ошибке "Illegal function call";

– при использовании режимов SCREEN 7 – SCREEN 10 значения основного и фонового цветов не должны превышать 15;

– в графическом режиме при установке фонового цвета в заданный цвет окрашивается вся область экрана, в текстовом же – только часть экрана, занятая текстовой информацией;

– при задании основного и фонового цветов необходимо выбирать цвета, различные по контрасту. Если основной и фоновый цвета одинаковы или близки по контрасту, то выводимые символы становятся невидимыми или плохо различимыми;

– если фоновый цвет не задан, то для всех типов видеоадаптеров и всех экранных режимов по умолчанию устанавливается черный (нулевой) фон экрана;

– при использовании режимов SCREEN 12 и SCREEN 13 оператором COLOR устанавливается только основной цвет.

7.2. Построение графических изображений на экране

Рассмотрим простейшие операторы и конструкции языка QBasic, позволяющие выводить на экран различные графические изображения.

7.2.1. Построение точки

Как уже упоминалось ранее, при включении графического режима

монитор разбивается на отдельные пиксели, количество которых по горизонтали и вертикали равно соответствующей разрешающей способности монитора в заданном режиме. В этом случае экран представляет собой как бы координатную плоскость, поэтому положение каждого пикселя на экране задается двумя координатами – X и Y. Началом координат считается верхняя левая точка экрана, положительным направлением оси X – направление слева направо, оси Y – направление сверху вниз, рис. 1.

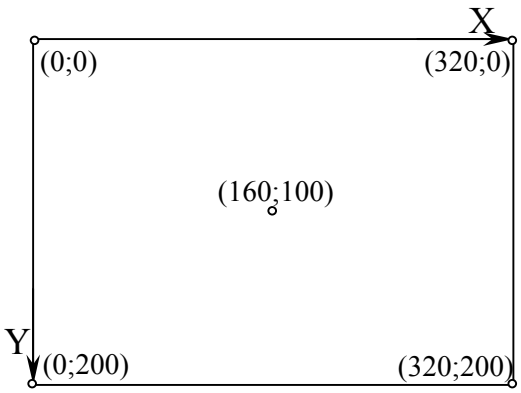


Рис. 1. Расположение координатной сетки экрана в режиме SCREEN 7

Аналогичное расположение имеет координатная сетка и в других режимах работы экрана, отличие заключается лишь в разрешающей способности монитора.

Такое разбиение экрана позволяет однозначно определять положение графических объектов путем задания их координат. Так, для построения точки на экране в QBasic используются операторы **PSET** и **PRESET**, имеющие одинаковый формат:

PSET [STEP] (X,Y), C

PRESET [STEP] (X,Y), C

где STEP – необязательный параметр, указывающий на то, что X и Y заданы относительно текущего графического положения курсора (в других операторах, которые будут рассмотрены нами в дальнейшем, смысл данного параметра аналогичен, поэтому не требует дополнительных пояснений); X,Y – координаты точки; C – параметр, задающий ее цвет.

Отличие указанных операторов заключается лишь в том, что при отсутствии параметра C оператор PSET для построения точки использует (по умолчанию) основной цвет, а оператор PRESET – фоновый цвет, установленный последним оператором COLOR, в остальном же работа данных операторов идентична.

При использовании операторов PSET или PRESET **необходимо помнить следующее**:

– при задании цвета точки необходимо учитывать режим работы монитора, поскольку в различных режимах используется разное количество возможных цветов. Так, в режиме SCREEN 1 кроме фонового и основного

могут применяться еще только два цвета (параметр цвета имеет смысл при С от 0 (фоновый цвет) до 3 (основной цвет), в режиме SCREEN 2 – два цвета (фоновый – черный и основной – белый), в режимах SCREEN 7, 8, 9, 12, 13 – все 16 цветов;

– попытка указать в качестве параметра С, число, большее чем количество цветов, используемое в заданном режиме, не приводит к ошибке. В этом случае цвет точки принимается по умолчанию (согласно последнему оператору COLOR);

– использование оператора PRESET без указания параметра цвета С приводит к построению невидимой точки (т.к. цвет точки в этом случае совпадает с фоновым цветом). Такую форму PRESET целесообразно использовать в цикле и только в сочетании с оператором PSET, когда необходимо последовательно "стирать" нарисованные точки, создавая, таким образом, иллюзию движения объекта на экране;

– если координаты точки находятся вне экрана, то никаких действий не производится и сообщений об ошибке не выдается;

– с помощью операторов PSET или PRESET можно строить не только точки, но и произвольные линии, например, графики функций. Для этого оператор, рисующий точку, помещают в цикл, в котором по определенному закону изменяют значения координат точки (подробнее об этом - см. п. 7.3).

7.2.2. Построение прямой линии и прямоугольника

Для построения прямой линии в QBasic используют оператор **LINE**, имеющий следующий формат:

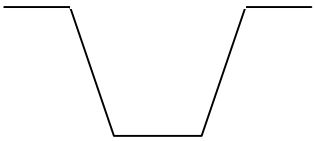
LINE [STEP] (X_1,Y_1) – [STEP] (X_2,Y_2), C

где X_1,Y_1 – координаты начала линии; X_2,Y_2 – координаты конца линии; С – параметр, задающий цвет линии.

При построении ломаной кривой (т.е. при совпадении координат конца предыдущего отрезка с координатами начала последующего) поочередно указывают несколько операторов LINE, причем координаты начал второго и последующих отрезков могут быть опущены, например:

```
SCREEN 7
COLOR 15, 1
LINE (60, 50)-(70, 50)
LINE -(80, 90)
LINE -(100, 90)
LINE -(110, 50)
LINE -(120, 50)
```

В результате выполнения программы получим следующее изображение:



Для построения прямоугольника можно использовать один оператор LINE, в этом случае он имеет следующий формат:

LINE [STEP] (X_1,Y_1) – [STEP] (X_2,Y_2), C, **B**

где X_1,Y_1 – координаты верхнего левого угла прямоугольника;

X_2,Y_2 – координаты нижнего правого угла прямоугольника;

C – параметр, задающий цвет контура прямоугольника;

B – опция, рисующая прямоугольник. Вместо опции B на ее месте может находиться опция **BF**. В этом случае на экран выводится изображение прямоугольника, закрашенного в цвет, заданный параметром C. Пример построения прямоугольников:

```
SCREEN 7
COLOR 7,4
LINE (50, 50)-(100, 80), 3, B
LINE (100, 100)-(130, 140), 9, BF
LINE (220, 100)-(250, 120), , B
```

При использовании оператора LINE **необходимо помнить следующее**:

– если указанные координаты начала или конца линии выходят за установленный экран, то "лишняя" линия обрезается у границ вывода;

– при построении линии параметр цвета C может быть опущен, в этом случае цвет линии определяется последним оператором COLOR, а при его отсутствии по умолчанию принимается белый цвет;

– при построении прямоугольника параметр цвета также может быть опущен, но для него обязательно выделяется место, ограниченное запятыми (как, например, в третьем операторе LINE вышеприведенного фрагмента), в противном случае на экран не выводится никаких изображений;

– для построения объектов, имеющих повторяющиеся элементы (например, осей, пунктирных линий, сеток и т.п.), оператор LINE целесообразно помещать в цикл с указанием закономерностей изменения начальных и конечных координат линии.

7.2.3. Построение окружности, эллипса и произвольной дуги

Для построения окружности, эллипса или любой их части (дуги) в QBasic используется оператор **CIRCLE**, имеющий следующий формат:

CIRCLE [STEP] (X,Y), R, C, XN, XK, K

где: X,Y – координаты центра окружности (эллипса);

R – радиус окружности;

C – параметр, задающий цвет окружности (эллипса);

XN, XK – начальный и конечный угол дуги окружности (эллипса);

K – параметр, задающий отношение длины оси Y к длине оси X эллипса, т.е. определяющий форму эллипса.

Обязательными в данном операторе являются только координаты центра и радиус окружности, остальные параметры могут указываться по мере

необходимости или приниматься по умолчанию.

При построении дуги окружности обязательно указываются начальный и конечный угол, причем начало отсчета совпадает с положительным направлением оси X, а угол поворота отсчитывается против часовой стрелки, рис. 2.

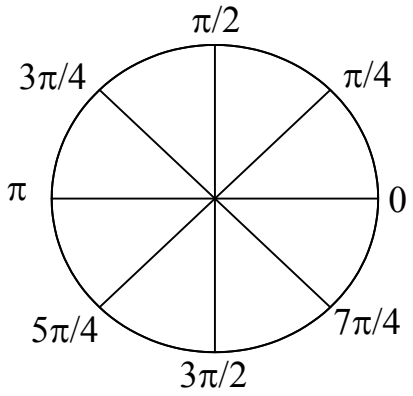
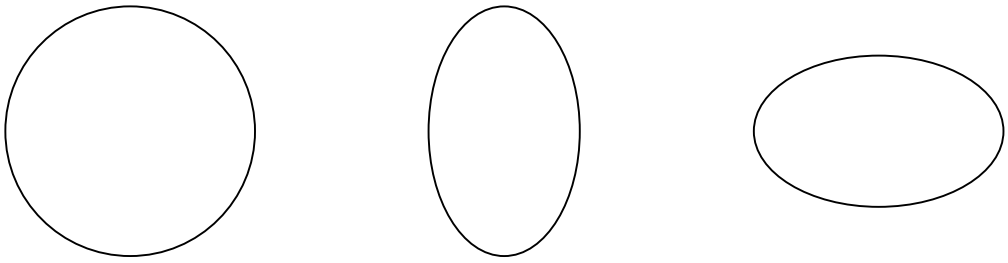


Рис. 2. Схема определения начального и конечного углов дуги

При построении эллипса в конце оператора CIRCLE обязательно указывается значение параметра K, определяющего соотношение длин осей эллипса. Влияние параметра K на форму эллипса показано на рис. 3.



а б в

Рис. 3. Результат выполнения оператора CIRCLE:
а – при K = 1; б – при K > 1; в – при K < 1

Приведем примеры использования оператора CIRCLE:

```
SCREEN 12
pi = 3.1416
CIRCLE STEP(0, 0), 30
CIRCLE (100, 50), 50, 4
CIRCLE (400, 300), 25, 9, 0, pi
CIRCLE STEP(50, 20), 30, , , , 2
CIRCLE (155, 180), 20, 1, 0, 3 * pi / 2, .5
CIRCLE (250, 200), 160, 5, 1.22, 2.73, 2
```

Поясним работу приведенных операторов CIRCLE (напомним, что при включении графического режима курсор автоматически устанавливается в центр экрана):

–1-й оператор рисует окружность в центре экрана (на это указывают координаты (0,0) в сочетании с параметром STEP) радиусом 30 пикселей белого (по умолчанию) цвета;

– 2-й – рисует окружность с центром в точке с координатами (100,50) радиусом 50 пикселей красного цвета (C=4);

– 3-й – рисует дугу (верхнюю половину) окружности с центром в точке (400,300) радиусом 25 пикселей ярко-синего цвета;

– 4-й – рисует эллипс, вытянутый вдоль вертикальной оси, с центром в точке с координатами (450, 320) белого (по умолчанию) цвета;

– 5-й – рисует часть эллипса (без четвертой четверти), вытянутого вдоль горизонтальной оси с центром в точке (155, 180) синего цвета;

– 6-ой – рисует дугу (часть эллипса, вытянутого вдоль вертикальной оси с центром в точке (250,200) фиолетового цвета.

Любой параметр оператора CIRCLE может быть задан не только в виде конкретного числового значения, но и виде переменной, значение которой должно быть определено к моменту вызова оператора CIRCLE. Параметры оператора CIRCLE, заданные переменными, могут изменяться в ходе выполнения программы (например, в цикле), что позволяет создавать более сложные графические изображения при достаточной простоте текста программы. Например:

```
SCREEN 12
FOR K = .5 TO 1.5 STEP .5
    CIRCLE (50, 100), 50, 5, , , K
NEXT K
```

Результат выполнения данного фрагмента программы представлен на рис.4:

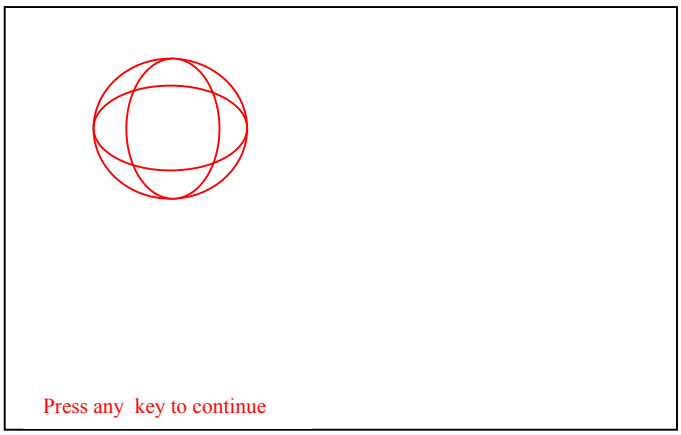


Рис. 4 Результат выполнения программы

При использовании оператора CIRCLE **необходимо помнить следующее**:

– некоторые аргументы в середине оператора могут быть опущены (в этом случае они принимаются по умолчанию), однако их позиция должна

быть обязательно задана наличием запятых. При опускании последних аргументов запятые могут отсутствовать;

– при задании начальной и конечной величин углов дуги их значение может находиться в пределах от -2π до 2π, в противном случае возникает ошибка *"Illegal function call"*;

– если задан конец дуги без начала, то рисуется дуга от 0° до указанного значения конца. Если указано начало без конца, то рисуется дуга от заданного начального значения до 2π;

– использование знака "–" при указании углов дуги означает, что в соответствующую концевую точку из центра окружности будет проведен радиус. Значение углов в этом случае отсчитывается так же, как и при положительных углах. Значение –0 не воспринимается как отрицательное, поэтому, для того чтобы нарисовать радиус с углом 0, можно вместо –0 указать очень малое отрицательное значение, например, –0,001;

– после изображения окружности курсор находится в точке с координатами ее центра;

– центр окружности или ее часть может находиться вне пределов экрана. В этом случае сообщение об ошибке не выдается а на экран выводится только "видимая" часть окружности;

– при отсутствии параметра C, задающего цвет окружности, изображение выводится в соответствии с установкой основного фона последним оператором COLOR;

– значение параметра C в операторе CIRCLE не ограничивается, однако при C, большем количества цветов для заданного режима, параметр цвета игнорируется, и изображение выводится белым (по умолчанию).

7.2.4. Построение произвольного графического изображения

С помощью операторов LINE и PSET (PRESET) удобно строить несложные графические изображения или изображения, координаты точек которых имеют некоторую закономерность, описываемую математически, что позволяет использовать данные операторы, помещая их в цикл. Однако, в тех случаях, когда выводимое изображение состоит из множества точек или линий, закономерность изменения координат которых установить затруднительно или невозможно, использование оператора LINE приводит к значительному увеличению программы, поскольку каждый отдельный прямой отрезок выводимого рисунка должен быть описан своим оператором LINE.

В этом случае на помощь программисту приходит оператор **DRAW**, который объединяет многие возможности графических операторов в своем макроязыке. Этот оператор позволяет выводить на экран произвольные графические изображения, используя для этого компактную командную строку.

Формат данного оператора :

DRAW "< командная строка >"

Командная строка может включать в себя указание последовательного направления движения курсора или рисования линии, а также команды цвета, вращения и масштаба. Команды, используемые в строке оператора DRAW приведены в табл. 11.

Таблица 11

Команды оператора DRAW

Команда	Описание
Команды движения относительно текущей точки	
U [n]	Вверх на n единиц
D [n]	Вниз на n единиц
L [n]	Влево на n единиц
R [n]	Вправо на n единиц
E [n]	Диагонально вверх и вправо на n единиц
F [n]	Диагонально вниз и вправо на n единиц
G [n]	Диагонально вниз и влево на n единиц
H [n]	Диагонально вверх и влево на n единиц
M x,y	В точку с координатами x,y (при отсутствии знаков перед x и y)–абсолютное M. Изменить положение курсора на x позиций по горизонтали и на y позиций по вертикали (при наличии перед x и y знаков "+" или "–") – относительное M
Команды установки угла, цвета и масштаба	
A [n]	Установка угла поворота n. Значение n находится в пределах от 0 до 3, где 0 соответствует 0°, 1 – 90°, 2 – 180°, 3 – 270°.
TA [n]	Поворачивает изображение на угол в n градусов. Значение n находится в пределах от –360 до 360. Если n>0, то поворот производится против часовой стрелки, иначе – по часовой стрелке
C [n]	Установка цвета n
S [n]	Установка масштабного фактора n, где n находится в пределах от 1 до 255. Масштабный фактор увеличивает единицы перемещения команд U, D, L, R и относительной M
P [n, m]	Установка цвета фигуры (n – цвет области, m – цвет границ)

Примечание. Приведенные команды движения курсора U, D, L, R, E, F, G, H осуществляют передвижение курсора в заданном направлении с одновременным рисованием линии, причем после выполнения команды курсор попадает в точку с новыми координатами. Если же перед командой поставить атрибут N, то соответствующие команды (NU, ND, NL, NR, NE, NF, NG, NH) кроме рисования линии будут возвращать графический курсор в начальную позицию). Если указан атрибут B, то соответствующие команды (BU, BD, BL, BR, BE, BF, BG, BH) осуществляют перевод курсора по экрану без рисования.

Приведем несколько примеров, наглядно иллюстрирующих работу команд движения оператора DRAW.

```
SCREEN 12
DRAW "D60 L6 H10 U40 E10 R6 "'противовес
DRAW "BM -100,-100 D80 F20 R30 E20 U80 L70"'бункер
DRAW "BM -50,+100 M-60,0 M+15,-50 M+30,0 M+15,+50"'трапеция
DRAW "BM 400,400 NM -45,0 NM -10,-3  NM -10,+3"'стрелка
```

Результат выполнения данного фрагмента программы представлен на рис. 5.

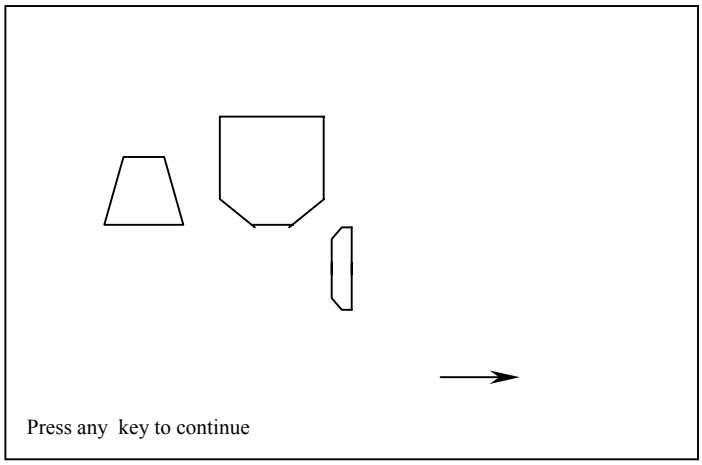


Рис. 5. Результат выполнения программы

Поясним работу вышеприведенных операторов DRAW (все они выводят графические изображения белым цветом по черному фону, установленными по умолчанию):

– 1-й оператор DRAW начинает рисование с центра экрана и согласно командам движения курсора рисует шестиугольник, возвращаясь в результате в центр экрана;

– 2-й – начинает рисование с точки, смещенной от предыдущей позиции курсора (в данном случае от центра) на 100 пикселей влево (ось X) и 100 пикселей вверх (ось Y), также используя команды движения;

– 3-й – смещает позицию курсора (без рисования) на 50 пикселей влево и 100 пикселей вниз, а затем использует команду относительного движения M, которая последовательно задает приращения координат по X и по Y;

– 4-й – устанавливает курсор в точку с координатами (400;400) и, используя команду относительного движения, рисует стрелку, причем атрибут N позволяет возвращать курсор в правый конец стрелки после рисования каждой из трех линий.

Рассмотрим использование команд установки угла, цвета и масштаба оператором DRAW:

```
SCREEN 7
COLOR 9, 10
DRAW "C9"
DRAW "S50"
DRAW "TA 120"
DRAW "BM 100,100 E5 F5 G5 H5"
DRAW "BR3 P12,9"
```

Результат выполнения данного фрагмента программы представлен на рис. 6.

Здесь, 1-й оператор DRAW устанавливает ярко-синий цвет выводимого графического изображения;

– 2-ой – масштабирует команды движения курсора, поэтому все значения последующих команд движения увеличиваются в 50 раз;

– 3-ий – указывает на то, что следующее изображение необходимо повернуть относительно его начального положения на 120° против часовой стрелки;

– 4-ый – устанавливает курсор в точку с координатами (100,100) и с помощью команд движения курсора рисует ромб;

– 5-ый – переводит курсор внутрь полученной фигуры и закрашивает ромб в светло-красный цвет.

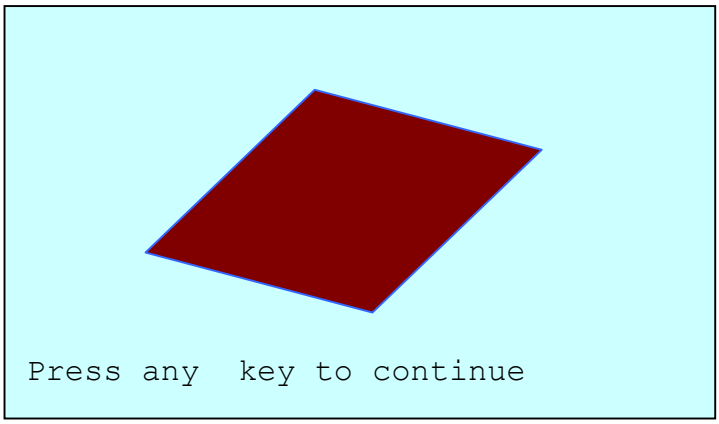


Рис. 6. Результат выполнения программы

Если некоторый графический объект требуется многократно выводить на экран в ходе выполнения программы, то для сокращения текста программы можно ввести символьную переменную, которая включала бы в себя последовательность команд, выводящих на экран требуемое изображение, а затем, использовать конструкцию

DRAW "<командная строка> **X**" + **VARPTR$**(<имя символьной переменной>)

где X – команда, выполняющая после основной командной строки DRAW подстроку, записанную в качестве символьной переменной;

VARPTR$ – функция преобразующая символьную переменную в последовательность команд для оператора DRAW.

Например, в ходе выполнения программы по выбору схемы армировки вертикального ствола несколько раз (или в разных частях экрана) необходимо вывести изображение поперечного сечения рельса. В этом случае целесообразнее задать команду его построения в виде символьной переменной, например, с именем RELS$, а затем с помощью конструкции DRAW "X" + VARPTR$(RELS$) "расставлять" изображения в нужные места. Фрагмент программы в этом случае будет выглядеть следующим образом:

```
SCREEN 12
RELS$ = "L9 R5 U5 E2 U2 L5 D1 R5 D1 L5 F1 NR2 F1 D5"
DRAW "BM 315,150 X" + VARPTR$(RELS$)
DRAW "BM 395,150 X" + VARPTR$(RELS$)
DRAW "TA 180 BM 307,160 X" + VARPTR$(RELS$)
DRAW "TA 180 BM 387,160 X" + VARPTR$(RELS$)
LINE (250, 151)-(460, 159), , B
```

Результат выполнения программы представлен на рис. 7.

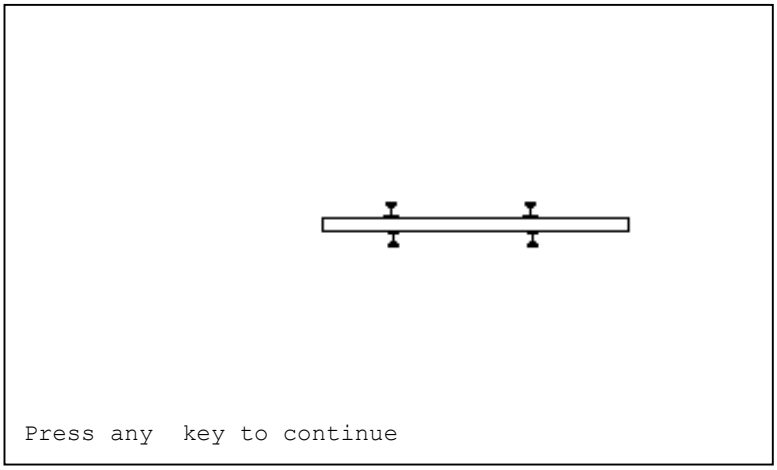


Рис. 7. Результат выполнения программы

Здесь символьной переменной RELS$ задана последовательность команд движения, рисующая поперечное сечение рельса. Первые два оператора DRAW выводят изображение рельса таким, каким оно задано переменной RELS$, предварительно установив курсор в нужную точку (1-й – в точку с координатами (315, 180), 2-й – (395, 180). Последующие два оператора рисуют, начиная с заданной точки, изображения рельса, повернутые на 180°. Оператор LINE рисует незакрашенный прямоугольник (расстрел).

При использовании оператора DRAW **необходимо помнить следующее**:

– команды установки цвета, поворота и масштабирования объекта должны стоять перед командами, выводящими на экран данный объект;

– различные команды оператора DRAW могут использоваться как последовательно в одном операторе DRAW, так и разбиваться на несколько операторов исходя из соображений удобства отладки программы;

– цвет выводимого изображения устанавливается самим оператором DRAW (командой C) независимо от наличия или отсутствия оператора COLOR;

– при отсутствии команды C оператора DRAW изображение выводится по умолчанию (белым) независимо от основного цвета, установленного оператором COLOR;

– при использовании команды установки цвета фона и границ нарисованных фигур P необходимо следить за тем, чтобы параметр цвета, задаваемый командой C, совпадал с цветом границ фигуры, задаваемым командой P, в противном случае попытка закрасить фигуру в любой цвет приведет к закрашиванию всей области экрана. Аналогичный результат будет и при попытке использования команды P без предварительной установки цвета командой C;

– закрашивание некоторой фигуры (замкнутой области) командой P должно осуществляться после установления курсора внутри данной области экрана. Если курсор находится на границе области (изображенной линии), то

команда P не выполняется. Если же курсор находится вне замкнутой области, то команда P закрашивает оставшуюся часть экрана вне области;

– если в результате выполнения команд движения курсор окажется за пределами экрана, сообщение об ошибке не выдается, а "лишние" части линий отсекаются. Однако в этом случае команда P, закрашивающая данную область, не выполнится, поскольку фигура окажется незамкнутой в пределах экрана;

– при повороте изображения на угол 90 или 270° размеры фигур автоматически масштабируются в отношении 4/3 к их размерам в углах 0 или 180°. Это связано с неодинаковостью размеров пикселя по горизонтали и вертикали.

Резюмируя изложенное в п. 7.2, приведем программу построения поперечного сечения вертикального ствола, иллюстрирующую работу графических операторов языка QBasic.

```
CLS
INPUT "Введите тип видеоадаптера 1-EGA, CGA; 2- VGA, SVGA)"; M
IF M = 1 THEN K1 = 1: K2 = 1: R = 1 ELSE K1 = 2: K2 = 2.4: R = 12
SCREEN R
CIRCLE (160 * K1, 97 * K2), 70 * K1
CIRCLE (160 * K1, 97 * K2), 66 * K1
FOR Y = 36 * K2 TO 156 * K2 STEP 14 * K2
    LINE (160 * K1, Y)-(160 * K1, Y + 8 * K2): REM ось Y
    LINE (160 * K1, Y + 10 * K2)-(160 * K1, Y + 12 * K2)
NEXT Y
FOR X = 86 * K1 TO 226 * K1 STEP 14 * K1
    LINE (X, 97 * K2)-(X + 8 * K1, 97 * K2): REM ось X
    LINE (X + 10 * K1, 97 * K2)-(X + 12 * K1, 97 * K2)
NEXT X
FOR X = 135 * K1 TO 185 * K1 STEP 50 * K1
    FOR Y = 63 * K2 TO 125 * K2 STEP 14 * K2
       LINE (X, Y)-(X, Y + 8 * K2)
       LINE (X, Y + 10 * K2)-(X, Y + 12 * K2)
       LINE (X, Y)-(X, Y + 8 * K2)
       LINE (X, Y + 10 * K2)-(X, Y + 12 * K2):REM оси клетей
NEXT Y, X
LINE (120 * K1, 65 * K2)-(150 * K1, 129 * K2), , B
LINE (170 * K1, 65 * K2)-(200 * K1, 129 * K2), , B
LINE (158 * K1, 37 * K2)-(162 * K1, 157 * K2), , B
RELS1$ = " D4 U2 L3 U1 L1 D2 R1 U1 "
RELS12$ = "D9 U5 L5 H2 L2 D5 R1 U5 R1 D5 E2 L1 U1 F1 R5"
IF M = 1 THEN
    DRAW "BM157,75 X" + VARPTR$(RELS1$)
    DRAW "BM157,115 X" + VARPTR$(RELS1$)
    DRAW "A2 BM163,79 X" + VARPTR$(RELS1$)
    DRAW "A2 BM163,119 X" + VARPTR$(RELS1$)
ELSE
    DRAW "BM315,180 X" + VARPTR$(RELS12$)
    DRAW "BM315,276 X" + VARPTR$(RELS12$)
    DRAW "A2 BM325,189 X" + VARPTR$(RELS12$)
    DRAW "A2 BM325,285 X" + VARPTR$(RELS12$)
END IF
```

Результат выполнения программы приведен на рис. 8.

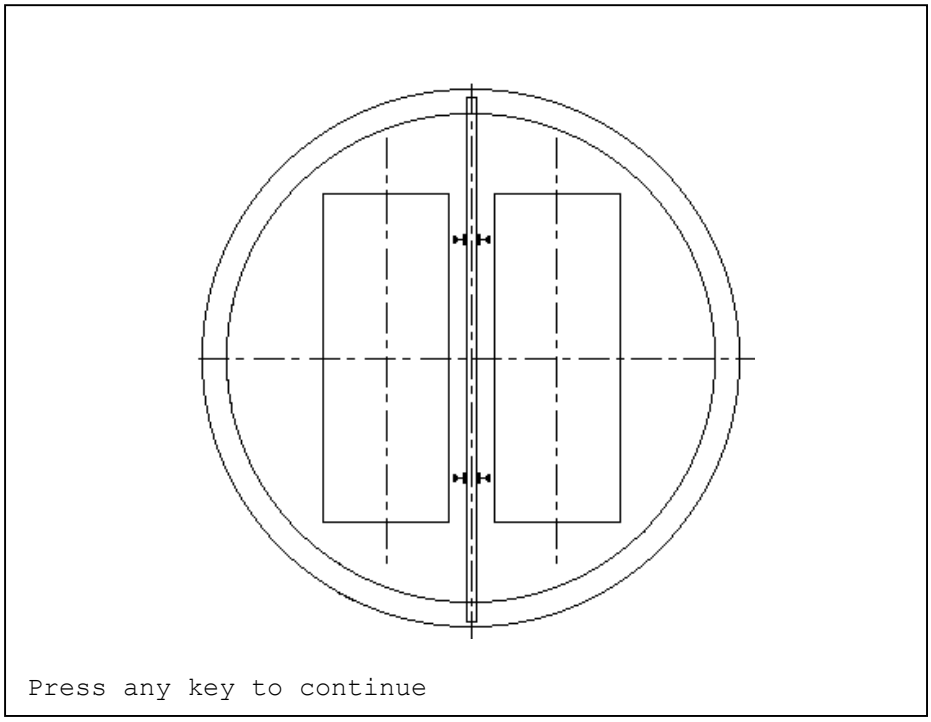


Рис. 8. Результат выполнения программы

Приведенная программа предусматривает возможность работы на мониторах с различными типами видеоадаптеров. В зависимости от заданного типа адаптера программой устанавливается лучший графический режим монитора, а координаты линий рассчитываются с учетом поправочных коэффициентов К1 и К2, задаваемых в зависимости от разрешающей способности монитора в том или ином режиме. Оси ствола и клетей рисуются в циклах, а рельсовые проводники – с использованием конструкции DRAW "X" + VARPTR$.

7.3. Построение графиков функций

Для построения графика функции с помощью вышеописанных операторов необходимо изменять в цикле аргумент (координату x выводимой точки) и рассчитывать по требуемому уравнению значение функции (координату y). Затем остается лишь ввести коэффициенты пропорциональности, производящие пересчет значений x и y в пиксели и "привязать" выводимые точки к координатным осям.

Приведем пример программы для построения графика функции $y=\sin^2 x$.

```
CLS
SCREEN 12
PI = 3.1416
PRINT "Введите желаемый формат окна для вывода графика (в пикселях):"
INPUT "  по горизонтали -"; G
INPUT "  по вертикали   -"; V
CLS
```

```
KX = G / 15
KY = V / 10: REM поправочные коэффициенты к координатам
PRINT TAB(32); " ГРАФИК ФУНКЦИИ "
LINE STEP(-G / 2, -V / 2)-STEP(G, V), , B:  REM окно
LINE STEP(-.1 * G, -V / 2)-STEP(-.8 * G, 0): REM ось X
LINE STEP(.4 * G, -.4 * V)-STEP(0, .8 * V): REM ось Y
FOR X = -2 * PI TO 2 * PI STEP .01
    Y = (SIN(X)) ^ 2
  X1 = 320 + KX * X
  Y1 = 240 - KY * Y
  IF Y1 > (240 - .4 * V) AND Y1 < (240 + .4 * V) THEN
   IF X1 > (320 - .4 * G) AND X1 < (320 + .4 * G) THEN PSET(X1, Y1)
  END IF
NEXT X
```

Результат выполнения программы приведен на рис. 9.

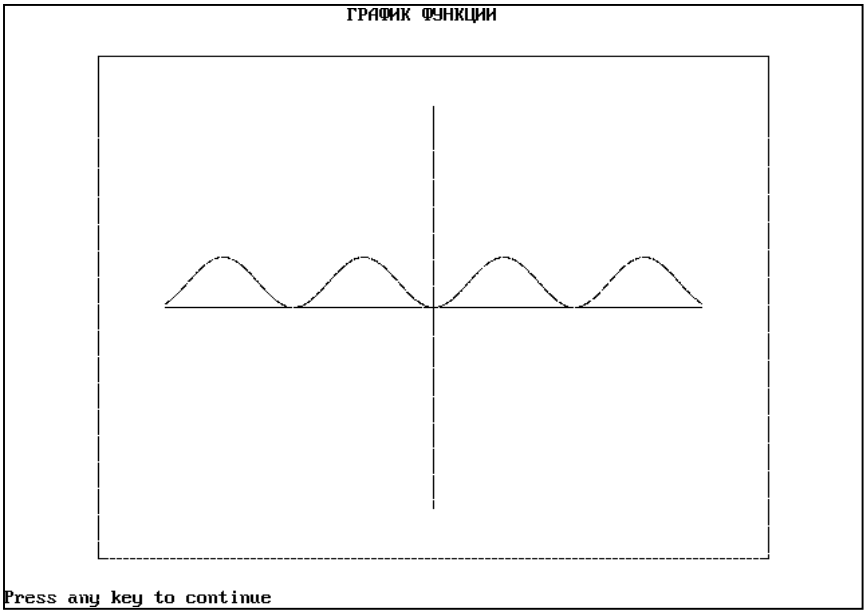


Рис. 9. Результат выполнения программы

Данная программа предусматривает вывод графика функции в окне, размеры которого задаются пользователем. Координатные оси и рамка привязаны к центру экрана, а линия выводимого графика ограничивается с четырех сторон размерами рамки.

Однако такое построение программы имеет ряд неудобств, основные из которых заключаются в следующем. Центральная точка экрана имеет координаты (320,240), что создает дополнительные трудности при размещении в ней начала координат , т.е. фактически точки с координатами (0,0). Кроме того, пределы изменения *x* и *y* не превышают всего нескольких единиц, в то время как координаты экрана составляют несколько сот пикселей. Это приводит к необходимости введения поправочных коэффициентов KX и KY, приводящих значения *x* и *y* к пикселям. Значение коэффициентов KX и KY в свою очередь зависит от выбранного размера окна.

Во избежание перечисленных трудностей, в данном случае целесообразно использовать оператор **WINDOW**, который позволяет любым образом изменить координатную сетку экрана. Данный оператор имеет следующий формат:

$$\text{WINDOW } (X_1,Y_1) - (X_2,Y_2)$$

где X_1,Y_1 – координаты левого нижнего угла экрана; X_2,Y_2 – координаты правого верхнего угла экрана.

Использование данного оператора очень удобно именно при построении графиков. Например, оператор WINDOW (-2,-2) – (2,2) создает на экране следующую координатную сетку (рис. 10). Сравните ее с координатной сеткой, приведенной на рис. 1, которая принимается по умолчанию.

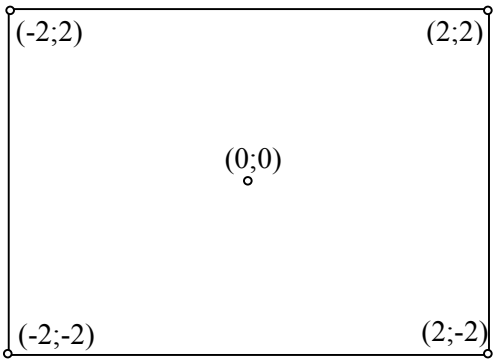


Рис. 10. Преобразование координатной сетки экрана оператором WINDOW

Такое расположение координатных осей позволяет значительно упростить программу построения графика функции. В этом случае данная программа может выглядеть следующим образом:

```
CLS
SCREEN 12
INPUT "Введите начальное и конечное значение X"; XN, XK
CLS
PRINT TAB(30); " ГРАФИК ФУНКЦИИ "
WINDOW (XN, -2)-(XK, 2)
LINE (XN, 0)-(XK, 0): REM ось X
LINE (0, -2)-(0, 2): REM ось Y
FOR X = XN TO XK STEP .01
   Y = (SIN(X))^2
   PSET (X, Y)
NEXT X
```

Здесь размеры окна по горизонтали и положение вертикальной оси задаются исходя из начального и конечного значений аргумента. Размер по вертикали ограничен исходя из возможных значений функции. В общем случае размер по вертикали целесообразно принимать несколько большим, чем максимальное значение функции на данном отрезке. Способ же нахождения максимального значения был рассмотрен нами ранее.

7.4. Лабораторная работа №5

Тема. Использование графических операторов в QBasic.

Цель работы. Изучить основные возможности компьютерной графики в QBasic освоить приемы построения графических объектов, ознакомиться с формами и размерами поперечных сечений горных выработок.

Задание. Согласно варианту задания составить программу построения поперечного сечения горной выработки с расположенным в ней оборудованием, используя основные графические операторы QBasic.

7.4.1. Варианты заданий

№ варианта	Наименование выработки	Форма сечения	Основные размеры	Располагаемое оборудование
1	Скиповой ствол	Круглая	$D_{св}$ = 6 м $t_{кр}$ = 300 мм	3 скипа, противовес
2	Бремсберг	Арочная	B_1 = 4100 мм $h_п$ = 1400 мм $t_{кр}$ = 150 мм	Ленточный конвейер, монорельсовая дорога
3	Откаточный штрек	Полигональная	задать самостоятельно	Ленточный конвейер
4	Ходок уклона	Трапециевидная	B_1 = 3700 мм B_2 = 3100 мм h = 2750 мм	_″_″_″_
5	Коренной штрек	Арочная	B_1 = 5200 мм $h_п$ = 1700 мм $t_{кр}$ = 200 мм	2 рельсовых пути на колею 900 мм
6	Скиповой ствол	Круглая	$D_{св}$ = 7 м $t_{кр}$ = 400 мм	4 скипа
7	Клетевой ствол	_″_″_″_	$D_{св}$ = 6 м $t_{кр}$ = 350 мм	2 клети, 2 противовеса
8	Клетевой ствол	_″_″_″_	$D_{св}$ = 8 м $t_{кр}$ = 300 мм	3 клети, противовес
9	Квершлаг	Арочная	B_1 = 4700 мм $h_п$ = 1300 мм $t_{кр}$ = 200 мм	Ленточный конвейер, рельсовый путь на колею 600 мм
10	Уклон	Трапециевидная	B_1 = 4200 мм B_2 = 3100 мм h = 2900 мм	Рельсовый путь на колею 900 мм
11	Вентиляционный штрек	Трапециевидная с наклонной кровлей	Задать самостоятельно	Рельсовый путь на колею 600 мм
12	Квершлаг	Прямоугольная	B_1 = 4500 мм h = 3400 мм	Ленточный конвейер, монорельсовая дорога
13	Штольня	Эллиптическая	Задать самостоятельно	Ленточный конвейер, рельсовый путь на колею 900 мм

Продолжение заданий

№ варианта	Наименование выработки	Форма сечения	Основные размеры	Располагаемое оборудование
14	Наклонный ствол	Сводчатая	То же	2 рельсовых пути на колею 900 мм
15	Штрек	Круглая	То же	Рельсовый путь на колею 900 мм

Примечание. Условные обозначения размеров выработок:

D – диаметр выработки в свету;

B_1, B_2 – ширина в свету соответственно нижнего и верхнего оснований;

h – высота выработки в свету;

h_n – высота прямолинейной части стоек;

$t_{кр}$ – толщина крепи.

При построении поперечных сечений вертикальных стволов расположение подъемных сосудов и армировки задать самостоятельно.

Пример выполнения лабораторной работы

Построить поперечное сечение штрека полуциркульной формы по следующим данным:

– ширина в свету нижнего основания $B_1 = 4700$ мм;

– высота прямолинейной части стоек $h_1 = 1500$ мм;

– толщина крепи $t_{кр} = 200$ мм;

– два рельсовых пути на колею $l = 900$ мм.

Присваиваем имена переменным и составляем таблицу идентификаторов:

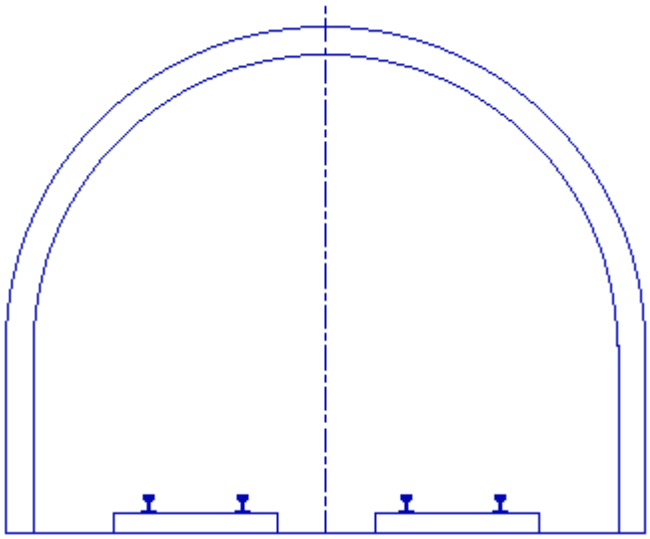
Переменная	Идентификатор	Примечание
B_1	B	Исходные размеры, мм, определяющие пропорции графического изображения
h_1	H_1	
$t_{кр}$	TK	
l	L	
–	H	Высота выработки, мм
–	RELS$	Символьная строка, задающая последовательность перемещения курсора для рисования поперечного сечения рельса

Программа на языке QBasic

```
CLS
SCREEN 12
COLOR 9
PI = 3.1415926#
B = 4700: H1 = 1500: TK = 200: L = 900
```

```
H = H1 + B / 2
WINDOW (-B, -H)-(B, H)
FOR I = -H1 TO B / 2 + TK STEP 330
    LINE (0, I)-(0, I + 180)
    LINE (0, I + 230)-(0, I + 280)
NEXT I
LINE (-B / 2, -H1)-STEP(B, 0)
LINE (-B / 2, -H1)-STEP(0, H1)
LINE (-B / 2 + TK, -H1)-STEP(0, H1)
LINE (B / 2 - TK, -H1)-STEP(0, H1)
LINE (B / 2, -H1)-STEP(0, H1)
CIRCLE (0, 0), B / 2, 9, 0, PI
CIRCLE (0, 0), B / 2 - TK, 9, 0, PI
RELS$ = "L3 D1 R7 U1 L3 U4 E2 U2 L5 D2 F2 D4 BU6 P9,9 BD6"
LINE (-B / 3, -H1 + 150)-STEP(1200, -150), , B
LINE (B / 3, -H1 + 150)-STEP(-1200, -150), , B
DRAW "BM230,322 X" + VARPTR$(RELS$)
DRAW "BM+47,0 X" + VARPTR$(RELS$)
DRAW "BM+82,0 X" + VARPTR$(RELS$)
DRAW "BM+47,0 X" + VARPTR$(RELS$)
```

Результат выполнения программы:



8. ПРИКЛАДНЫЕ ПРОГРАММЫ ДЛЯ РЕШЕНИЯ ЗАДАЧ ШАХТНОГО СТРОИТЕЛЬСТВА

8.1. Программа расчета параметров буровзрывных работ при проходке горизонтальных и наклонных выработок, проводимых смешанным забоем

Имя программы – BWR 4-PS
Автор программы – Вершинин Н.К.

8.1.1. Алгоритм расчета

Расчет параметров буровзрывных работ приемлем для выработок различной формы сечения площадью в проходке до 25 м^2 обычным способом при любом способе подрывки. В основе лежит табличный метод определения основных параметров необходимых для расчета.

Тип ВВ выбирается в зависимости от категории шахты по газу, наличия угля в забое выработки, работоспособности ВВ при соблюдении Правил безопасности. Диаметр (32 , 36 или 45 мм) и масса (200, 250, 300 г) патронов принимаются стандартными, согласно перечню выпускаемых промышленных ВВ. Диаметр шпуров принимается на 5-8 мм больше диаметра патрона.

Рекомендуемая глубина шпуров L выбирается из таблиц в зависимости от крепости пород, площади забоя, способа проведения выработки.

Глубина врубовых шпуров $L_{вр}$, м, рассчитывается по формуле:

$$L_{вр} = 1{,}15 \cdot L.$$

Глубина оконтуривающих шпуров $L_{ок}$, м,

$$L_{ок} = L.$$

Удельный расход ВВ q и число шпуров N также принимается по таблицам в зависимости от крепости пород, площади забоя, работоспособности ВВ, диаметра патрона, способа проведения выработки и др.

Расход ВВ на цикл Q, кг, рассчитывается отдельно для пород кровли, почвы и угля по формуле

$$Q = q \cdot S \cdot L \cdot \eta,$$

где S – площадь забоя (почва, кровля, уголь),м^2;

η – коэффициент использования шпура;

Средний заряд в шпуре $G_{ср}$, кг, определяется:

$$G_{ср} = Q / N.$$

Количество патронов в шпурах n рассчитывается в зависимости от массы патрона ВВ m, кг,

$$n = G_{ср} / m,$$

и округляется до ближайшего большего целого.

Фактический заряд в шпуре $G_ф$, кг:

$$G_ф = n \cdot m.$$

Линия наименьшего сопротивления (ЛНС) W, м, определяется исходя из площади забоя S, м^2, и числа шпуров N:

$$W = \sqrt{\frac{S}{N}} \ .$$

Длина забойки $L_з$, м, должна соответствовать Единым правилам безопасности при взрывных работах и рассчитываться по формуле

$$L_з = L - L_п \cdot n \ ,$$

где $L_п$ – длина патрона, м, (определяется в зависимости от плотности ВВ в патронах, диаметра и массы патрона).

Время замедления между взрывами определяется исходя из расчета оптимального времени замедления $T_з$, и принимается равным ближайшей большей стандартной серии (15 или 25 мс):

$$T_з = \frac{31{,}5}{\sqrt[4]{1{,}3 \cdot f}} \cdot W - 6 \cdot \sqrt[4]{1{,}3 \cdot f} + 9{,}6 \, ,$$

где f – коэффициент крепости пород.

Врубовые шпуры располагают в наиболее слабой породе (угле). Остальные шпуры располагают равномерно по всей площади забоя на расстоянии W друг от друга. Оконтуривающие шпуры располагают на линии, отстоящей на 5-10 см от контура выработки.

С помощью данной программы можно производить расчет паспорта БВР для выработок, пройденным по смешанному забою.

8.1.2. Исходные данные, подготавливаемые к расчету

Для расчета по программе BWR 4-PS необходимо подготовить следующие исходные данные.

Крепость пород кровли
Крепость пород почвы
Крепость угля
Площадь породного забоя кровли, м^2
Площадь породного забоя почвы, м^2
Площадь угольного забоя, м^2
Коэффициент использования шпура
Категория шахты по метану (I, II, III, сверхкатегорийная)
Диаметр патрона ВВ, мм
Масса патрона ВВ, г
Выемка горной массы, сплошная, раздельная

8.1.3. Результаты расчета

При выполнении программы на печать выводится следующие **результаты**:

категория шахты по газу;
крепость пород (кровли, почвы, угля);
площадь забоя (кровли, почвы, угля), м^2;
глубина шпуров(врубовых, отбойных, оконтуривающих), м;
КИШ;
число шпуров (кровли, почвы, угля);
ЛНС (кровли, почвы, угля), м;
расчетное время замедления, мс;
число патронов в шпуре (кровли, почвы, угля);
длина забойки шпура (кровли, почвы, угля), м;
заряд в шпуре (кровли, почвы, угля), кг;
принятое ВВ;
диаметр патрона, мм;
масса патрона, г;
длина патрона, см.

8.2. Расчет параметров буровзрывных работ при проходке горизонтальных и наклонных выработок, проводимых по однородным породам

Имя программы – BWR 4 – P
Автор программы – Вершинин Н.К.

8.2.1. Алгоритм расчета

Расчет параметров буровзрывных работ можно применить для выработок любой формы сечения площадью в проходке до 25 м^2 с применением контурного взрывания.

Длина шпуров рассчитывается исходя из глубины комплекта шпуров:

– врубовые $l_{вр} = 1{,}2 \cdot l_{шп}$;

– отбойные, оконтуривающие $l_{от} = l_{ок} = l_{шп}$,

где $l_{шп}$ – глубина комплекта шпуров, м.

Для оконтуривающих и внутренних шпуров выбираются типы ВВ в зависимости от категории шахты по газу при соблюдении Правил безопасности. Определяется длина линии расположения оконтуривающих шпуров и с учетом коэффициента крепости породы рассчитывается:

– расстояние между шпурами $a_{ок} = \dfrac{1}{0{,}11 \cdot f + 1{,}28}$,

– их число $N_{ок} = L_{ок} / \alpha_{ок}$,

– линия наименьшего сопротивления для оконтуривающих шпуров

$W_{ок} = \alpha_{ок}/m$,

где f – коэффициент крепости породы;

$L_{ок}$ – длина линии оконтуривающих шпуров, м;

m – коэффициент сближения шпуров.

Заряд в оконтуривающем шпуре $G_{ок}$ определяют исходя из погонного заряда $q_{ок}'$, кг/м:

$$G_{ок} = q_{ок}' \cdot l_{ок}.$$

С учетом ЛНС для оконтуривающих и внутренних шпуров определяют их площади работы.

Для внутренних шпуров рассчитывается удельный расход ВВ по таблицам или по формуле:

$$q = q_1 \cdot f_1 \cdot K_з \cdot e\ ,$$

где q_1 – нормируемый удельный расход ВВ, $q_1 = 0{,}1 \cdot f$;

f – коэффициент структуры породы;

$K_з$ – коэффициент зажима породы, $K_з = \sqrt{\dfrac{6{,}5}{S_{вн}}}$;

$S_{вн}$ – площадь работы внутренних шпуров, м^2;

e – работоспособность ВВ, см^3.

Число внутренних шпуров

$$N = 12{,}7 \cdot \frac{q \cdot S \cdot \eta}{\gamma \cdot d^2 \cdot \rho},$$

где η – коэффициент использования шпуров,

γ – коэффициент заполнения шпуров,

ρ – плотность патронирования ВВ, г/см^3;

d – диаметр патрона ВВ, см.

Расход ВВ на цикл

$$Q = q \cdot l_{шп} \cdot S_{вн} \cdot \eta.$$

Заряд во внутренних шпурах – отбойных и врубовых определяется по формулам:

$$G_{от} = Q / N_{вн};$$
$$G_{вр} = 1{,}15 \cdot G_{от}.$$

Число патронов в шпуре принимается целым. Длина забойки должна соответствовать Единым правилам безопасности при взрывных работах.

Время замедления принимается согласно стандартным сериям (15 или 25 мс), исходя из расчета оптимального времени замедления:

$$T_з = \frac{31{,}5}{\sqrt[4]{1{,}3 \cdot f}} \cdot W - 6 \cdot \sqrt[4]{1{,}3 \cdot f} + 9{,}6\,,$$

где W – ЛНС для внутренних шпуров, $W = n \cdot d_n / 2$**;**

n – коэффициент пропорциональности.

Общее сопротивление взрывной цепи складывается из сопротивлений электродетонаторов, соединительных и магистральных проводов.

8.2.2. Исходные данные, подготавливаемые к расчету

Для расчета по программе BWR 4-PS необходимо подготовить следующие исходные данные.

Крепость пород по шкале проф. Протодьяконова

Форма выработки (арочная, трапециевидная)

Размеры выработки в проходке, м:

– ширина (по верху, по низу)

– высота

Характеристика ВВ для оконтуривающих и внутренних шпуров:

– работоспособность, см^3;

– плотность патронирования, кг/м^3;

– масса патрона, кг

Глубина комплекта шпуров, м.

8.2.3. Результаты расчета

При выполнении программы на печать выводятся следующие **результаты:**

количество шпуров на цикл, шт;

коэффициент использования шпура;

подвигание забоя за цикл, м;

расход ВВ на цикл, кг;

объем бурения, шп·м;

общее сопротивление сети, Ом;

площадь ядра сечения, м^2;

ЛНС оконтуривающих шпуров, м;

ЛНС шпуров ядра сечения, м;

расстояние между оконтуривающими шпурами, м;

принятое время замедления, мс.

Кроме перечисленных данных в результате выполнения программы BWR 4-P на печать выводится таблица по следующей форме:

Назначение шпуров	Врубовые	Отбойные	Контурные
Длина шпура, м			
Заряд в шпуре, кг			
Число шпуров, шт			
Число патронов в шпуре, шт			

8.3. Программа расчета параметров буровзрывных работ при проходке вертикального ствола.

Имя программы – BWR 4-S
Автор программы – Вершинин Н.К.

8.3.1. Алгоритм расчета

Расчет параметров буровзрывных работ применяется для стволов круглого сечения диаметром от 4,5 до 9 м в свету с применением контурного взрывания.

Для ведения взрывных работ применяют малобризантные ВВ для оконтуривающих и высокобризантные – для внутренних шпуров в патронах диаметром 28-36 и 45 мм соответственно. Диаметр шпуров – 52 мм.

Алгоритм определения длины шпуров, их количества, величин зарядов в шпурах и общего расхода ВВ на цикл, принятый в программе BWR 4-S, аналогичен алгоритму, описанному в п. 8.2.1.

Шпуры располагают на концентрических окружностях по всему сечению ствола. Расстояния между шпурами приблизительно равно ЛНС. При углах падения пород более 40° внутренние шпуры располагают в шахматном порядке.

Линия наименьшего сопротивления $W_{ок}$ и расстояние между оконтуривающими шпурами $a_{ок}$ принимаются по таблицам, специально включенным в программу, а для внутренних шпуров рассчитываются по формулам:

$$W = n \cdot d_n / 2 \text{ ;}$$

$$a_{ок} = m \cdot W \text{ ,}$$

где n – коэффициент пропорциональности;

d_n – диаметр патрона ВВ, см;

m – коэффициент сближения шпуров.

Число оконтуривающих шпуров определяют исходя из длины линии оконтуривающих шпуров $B_{ок}$, м:

$$N_{ок} = B_{ок} / a_{ок} ,$$

Время замедления принимается согласно стандартным сериям (15 мс или 25 мс), исходя из расчета оптимального времени замедления:

$$t_{опт} = t_1 + t_2 + t_3 ,$$

где t_1 – время распространения ударной волны между группами шпуров, мс,

t_2 – время распространения микротрещин, мс;

t_3 – время раскрытия трещин, мс.

8.3.2. Исходные данные, подготавливаемые к расчету

Для расчета по программе BWR 4-S необходимо подготовить следующие исходные данные:

Крепость пород по шкале проф. Протодьяконова

Диаметр ствола в свету, м

Толщина крепи ствола, м
Глубина комплекта шпуров, м
Угол падения пород, °

8.3.3. Результаты расчета

При выполнении программы на печать в виде таблиц выводятся следующие результаты.

Характеристика и расход ВВ и данные о шпурах

Шпуры	Тип ВВ	Диаметр патрона, см	Масса патрона, кг	Масса ВВ в шпуре, кг	Число патронов ВВ
Внутренние					
Контурные					

Назначение шпуров	Врубовые	Отбойные	Контурные
Количество шпуров, шт			
Диаметр окружности, м			
Расстояние между шпурами, м			
Глубина шпуров, м			
Угол наклона, °			
ЛНС, м			
Масса ВВ в шпуре, кг			
Количество патронов, шт.			
Длина заряда, м			

Общее количество шпуров на цикл
Количество окружностей отбойных шпуров
Объем взрываемого массива, $м^3$
Объем бурения, шп·м

8.4. Программа расчета вентиляции горизонтальных и наклонных тупиковых выработок

Имя программы – Воздух - П
Автор программы – Красунцев Е.М.

8.4.1. Алгоритм расчета

Основной способ проветривания горизонтальных и наклонных тупиковых выработок – нагнетательный. Для выбора средств проветривания должны быть определены расходы воздуха: по выделению метана или углекислого газа; по газам, образующимся при взрывных работах в забое выработки; по числу людей, работающих в выработке; по средней минимальной скорости воздуха в выработке и минимальной скорости воздуха в призабой-

ном пространстве с учетом температуры. Выбор средств проветривания выработки производится по наименьшему из этих расходов.

Для выработки протяженностью до 300 м расчет выполняется сразу на максимальную длину. Для выработок большей протяженности расчеты выполняются на отдельные периоды для промежуточных значений 300, 600, 900 м и т.д., включая максимальную длину.

Программа предусматривает корректировку принятых параметров трубопровода, если расчетный напор H_B выходит за пределы допустимых значений.

Расход воздуха для проветривания тупиковой выработки по выделению метана или углекислого газа, м^3/с,

$$Q_{зп} = \frac{1{,}67 \cdot I_п \cdot K_н}{C - C_0},$$

где $I_п$ – метановыделение (выделение углекислого газа);

$K_н$ – коэффициент для угольного бассейна;

C – допустимая, согласно ПБ концентрация газа в исходящей струе, %;

C_0 – концентрация газа в струе воздуха, поступающего в выработку.

Расход воздуха по наибольшему количеству людей $n_{чел}$, одновременно работающих в выработке, м^3/с,

$$Q_{зп} = 0{,}1 \cdot n_{чел}\ .$$

Расход воздуха по минимальной скорости движения, м^3/с,

$$Q_{зп} = v_{п\ min} \cdot S\ .$$

Расход воздуха по минимальной скорости в призабойном пространстве в зависимости от температуры, м^3/с:

$$Q_{зп} = 0{,}333 \cdot v_{з\ min}\ .$$

Здесь S – площадь сечения выработки в свету; $v_{п\ min}$ – минимально допустимая скорость воздуха в тупиковой выработке, м/с; $v_{з\ min}$ – минимально допустимая скорость воздуха в призабойном пространстве в зависимости от температуры, м/с.

Расход воздуха для проветривания выработки по газам, образующимся при взрывных работах, м^3/с,

$$Q_5 = \frac{2{,}25}{T} \cdot \sqrt[3]{\frac{V_{ВВ} \cdot S_{св}^2 \cdot l_п^2 \cdot K_{обв}}{K_{ут}^2}},$$

где T – время проветривания выработки после взрывания, мин;

$V_{ВВ}$ – объем вредных газов, образующихся после взрывания, л,

$$V_{ВВ} = 100 \cdot B_{уг} + 40 \cdot B_{пор}\ ;$$

$B_{уг}$, $B_{пор}$ – масса одновременно взрываемых ВВ по углю и породе, соответственно, кг;

$l_п$ – расчетная длина тупиковой выработки или критическая длина, на которой происходит разжижение газов взрыва ниже предельной кон-

центрации (принимается к расчету минимальная из этих длин);

$K_{обв}$ – коэффициент обводненности выработки;

$K_{ут}$ – коэффициент утечек воздуха в вентиляционном трубопроводе,

$$K_{ут} = a + b \cdot l_n + c \cdot l_n^2 + k \cdot l_n^m \cdot Q_{зп} .$$

Здесь a, b, c, k, m – параметры уравнения, числовые значения которых задаются в программе в зависимости от типа трубопровода и его диаметра $d_{тр}$.

Аэродинамическое сопротивление гибкого трубопровода

$$R_{тр} = l_{нh} \cdot (\tau_{тр} + 20 \cdot d_{тр} \cdot n_1 + 10 \cdot d_{тр} \cdot n_2),$$

где $\tau_{тр}$ – удельное аэродинамическое сопротивление трубопровода,

n_1, n_2 – число поворотов трубопровода на 90 и 45°.

Подача вентилятора, м^3/с,

$$Q_в = Q_{зп} \cdot K_{ут.тр.} .$$

$K_{ут.тр.}$ вычисляется так же как и $K_{ут}$, но для полной длины трубопровода.

Давление вентилятора (депрессия гибкого трубопровода), даПа,

$$H_в = Q_в \cdot R_{тр} \cdot (0{,}59 / K_{ут.тр.} + 0{,}41)^2.$$

Выбор вентилятора производится путем наложения аэродинамических характеристик вентиляционного трубопровода и вентилятора. Если точка с координатами ($Q_в$, $H_в$), соответствующими расчетному режиму работы вентилятора лежит в экономичной зоне его работы (КПД $\eta \geq 0{,}6$), выбранный вентилятор может быть принят.

Для построения характеристик трубопроводов программа предусматривает вычисление координат трех точек при данной длине трубопровода.

8.4.2. Исходные данные, подготавливаемые к расчету

Для расчета по программе "Воздух-П" необходимо подготовить следующие исходные данные.

Породы забоя выработки (угольный пласт, пропластки, пустые породы)

Площадь поперечного сечения выработки в свету, м^2

Проектируемая длина выработки, м

Категория шахты по метану (I, II, III, сверхкатегорийная)

Концентрация метана в свежей струе, %

Допустимая концентрация метана в исходящей струе, %

Ожидаемое метановыделение в выработке, м^3/мин:

– с обнаженных поверхностей пластов

– из отбитого угля

– после взрывания по углю

Выделение углекислого газа, м^3/мин

Количество ВВ, взрываемого одновременно, кг:

– по углю

– по породе
Продолжительность проветривания после взрыва, мин
Коэффициент обводненности
Относительная влажность воздуха в забое, %
Количество людей в выработке, чел.
Параметры вентиляционного трубопровода:
– тип трубопровода
– диаметр, м
– длина звена, м
Максимально допустимое расстояние от конца трубопровода до забоя, м

8.4.3. Результаты расчета

При выполнении программы на печать выводятся следующие результаты:
Расчетная длина выработки, м
Расстояние от ВМП до устья выработки, м
Критическая длина выработки, м
Минимально допустимая скорость воздуха в выработке, м/с
Расчетный расход воздуха, $м^3$/с:
– по метановыделению
– по взрывным работам
– по минимальной скорости
– по количеству людей
– по тепловому фактору
Принятый расход воздуха в забое, $м^3$/с
Расчетная скорость воздуха, м/с
Максимально допустимая температура в забое, °С
Расход воздуха для всей выработки по метановыделению, $м^3$/с
Расчетные параметры вентиляционного трубопровода:
– тип трубопровода
– диаметр, м
– длина звена, м
Коэффициент утечек воздухопровода
Аэродинамическое сопротивление, кμ
Отставание труб от забоя, м
Расчетные параметры вентиляционной установки:
– подача, $м^3$/ с
– депрессия, даПа
Режим работы ВМП по характеристике:
– подача, $м^3$/ с
– депрессия, даПа
Количество воздуха, подаваемого к вентилятору, $м^3$/ с

8.5. Программа расчета вентиляции при сооружении вертикальных

стволов

Имя программы – Воздух - С
Автор программы – Красунцев Е.М.

8.5.1. Алгоритм расчета

Принцип расчета вентиляции при проведении ствола незначительно отличается от расчета вентиляции горизонтальных и наклонных тупиковых выработок. Отличие заключается в дополнительном определении расхода воздуха по разбавлению вредных веществ при сварочных работах. Для проветривания применяют жесткий трубопровод. На участках 40-50 м у забоя допускается применение гибких труб. Вентиляторы располагаются на поверхности не ближе 20 м от устья ствола при газовом режиме и не ближе 15 м от устья ствола при негазовом режиме.

8.5.2. Исходные данные, подготавливаемые к расчету

Для расчета по программе "Воздух-С" необходимо подготовить следующие исходные данные:

Наличие взрывных работ в стволе
Выделение углекислого газа, $м^3$/мин
Концентрация углекислого газа в атмосфере около ствола, %
Категория шахты по метану (I, II, III, сверхкатегорийная)
Наибольшее ожидаемое метановыделение в призабойное пространство, $м^3$ / мин
Основные параметры выработки:
– проектная глубина ствола, м
– площадь поперечного сечения ствола в свету, $м^2$
Расстояние от ВМП до устья ствола, м
Минимально допустимая скорость воздуха по тепловому фактору, м/с
Относительная влажность воздуха в стволе, %
Количество людей, работающих в стволе, чел.
Проведение сварочных работ:
– тип сварочных электродов
– расход электродов на 1 м сварочного шва, кг/м
– количество одновременно работающих сварочных постов, шт.
Приток воды в ствол, $м^3$/ч
Параметры буровзрывных работ:
– количество одновременно взрываемого ВВ, кг
– время проветривания после взрыва, мин
Параметры вентиляционного трубопровода:
– диаметр, м
– длина звена, м
– качество уплотнения стыков (хорошее, удовлетворительное)

– качество труб (новые, старые)
– фасонные части трубопровода и их количество
– максимально допустимое расстояние от конца трубопровода до забоя, м

8.5.3. Таблица результатов расчета

При выполнении программы на печать выводится следующие результаты:
Расчетная глубина ствола, м

Остальные результаты расчета см. в п. 8.4.3. применительно к расчету ствола.

Для построения рабочей характеристики трубопровода программа выдает таблицу значений расхода воздуха Q_i и депрессии H_i по трем точкам. Для каждой рассчитываемой глубины ствола строится характеристика трубопровода в координатах *Q-H*, совмещаемая с аэродинамической характеристикой вентилятора и, таким образом, по максимальному КПД выбирается оптимальный тип и режим работы вентилятора (ниже приведена форма таблицы).

L, м	Q_1	H_1	Q_2	H_2	Q_3	H_3

После выбора режимов работы вентилятора для каждой из рассчитываемых глубин ствола программа выдает итоговую таблицу результатов расчета режимов проветривания, форма которой приведена ниже:

L, м	$Q_{зп}$, м^3/с	$K_{ут}$	$R_{тр}$, кμ	$T_{зп}$, °С	$Q_в$, м^3/с	$H_в$, даПа	Определяющий фактор

8.6. Программа выбора схемы и расчета технико-экономических показателей армировки вертикальных стволов

Имя программы – ARMIR1
Автор программы – Прокопов А.Ю.

8.6.1. Алгоритм расчета

Программа ARMIR1 позволяет произвести выбор схемы армировки и оценить металлоемкость конструкции, трудоемкость и стоимость армирования по выбранной схеме.

Программа включает 24 схемы вертикальных стволов с соответствующими им характеристиками, из них 9 наиболее распространенных типо-

вых схем клетевых и 5 схем скиповых стволов, а также 7 альтернативных безрасстрельных схем клетевых и 3 схемы скиповых стволов.

В начале выполнения программы для выбора соответствующей схемы пользователю предлагается ознакомиться с типовым рядом сечений вертикальных стволов Южгипрошахта, а также с альтернативным рядом безрасстрельных схем, при этом на экран дисплея выводятся поочередно графические изображения всех поперечных сечений и приводятся характеристики каждой из запрограммированных схем, включающие :

– число, тип и габаритные размеры применяемых подъемных сосудов и противовесов;

– диаметр и глубину ствола, для которых применима данная схема;

– вид и типоразмер профиля расстрелов и проводников, принятые в типовом варианте (данные параметры могут быть изменены в ходе дальнейшего выполнения программы);

– расположение проводников относительно подъемных сосудов и противовесов.

После просмотра всех предложенных программой схем и характеристик, пользователь производит выбор одной из схем, при этом задает код, соответствующий выбранной схеме, по которому программа начинает расчет технико-экономических показателей.

Кроме указанного кода, исходными данными для этого расчета являются только форма и типоразмер (номер) профилей, используемых для расстрелов и проводников, а также глубина ствола и принятый шаг армировки.

При задании исходных данных пользователь может обращаться к содержащейся в программе справочной информации о применяемых для элементов армировки прокатных профилях и их основных параметрах, о возможном шаге армировки и др.

Алгоритм предусматривает автоматический ввод всех необходимых характеристик стандартных профилей, применяемых для изготовления элементов армировки, поэтому металлоемкость рассчитывается исходя из линейной плотности заданных типоразмеров профилей и выбранной схемы армировки.

Аналогично в программе заложены основные расценки и нормы времени на выполнение различных операций при армировании ствола. Программа "выбирает" необходимые значения из соответствующих массивов данных в зависимости от выбранной схемы, профилей проводников и расстрелов, а также способа крепления расстрелов, толщины крепи и крепости вмещающих пород. Общая стоимость (и трудоемкость) армирования яруса определяется как сумма стоимостей (трудоемкостей) отдельных операций.

8.6.2. Исходные данные, подготавливаемые к расчету

Для расчета по программе ARMIR1 необходимо подготовить следующие исходные данные:

Тип ствола (клетевой, скиповой)
Тип армировки (типовая, безрасстрельная)
Тип и типоразмер профиля расстрела (двутавр, коробчатый, другой)
Тип и типоразмер профиля проводника (рельс, коробчатый, другой)
Шаг армировки, м
Дополнительные характеристики
для типовых схем:
– глубина заделки расстрела, м
для безрасстрельных схем:
– глубина шпуров под анкера, м
– диаметр штанг анкеров, мм
– толщина крепи, м
– крепость вмещающих пород по шкале проф. Протодьяконова

8.6.3. Результаты расчета

В результате выполнения программы ARMIR1 на печать выдаются следующие данные и таблица по нижеприведенной форме.

Схема
Профиль проводника
Профиль расстрела
Шаг армировки
Глубина заделки расстрелов в крепь (для типовых схем)

Наименование параметров	Ед. изм.	Показатели	
		На ярус	На 1 м
Металлоемкость	кг		
Трудоемкость	чел-ч		
Стоимость монтажа	руб.		

8.7. Программа расчета параметров графика организации работ

Имя программы – "График"
Автор программы – Дмитриенко В.А.

8.7.1. Алгоритм расчета

Данная программа предназначена для расчета основных параметров организации работ проходческого цикла.

Трудоемкость работ проходческого цикла определяется согласно "Единым нормам времени и расценок". Трудоемкость работ каждой операции находят как произведение объема работ W на установленную норму времени $H_{вр}$:

$$T_i = W \cdot H_{вр} .$$

Принятая норма определяется с учетом поправочного коэффициента K_i от норм:

$$H_{вр} = H_{вр}^{'} \cdot K_i .$$

Состав звена проходчиков устанавливается исходя из общей трудоемкости работ $T_ц$, чел-смен, которую определяют на цикл или на 1 м:

$$T_ц = \Sigma T_ц / t_ц ,$$

где $t_ц$ – продолжительность цикла, ч.

Учитывая нормы размещения проходчиков в забое (1 человек на 2÷3 м^2 площади забоя), возможности перевыполнения норм выработки до 20%, принимается состав звена на цикл $n_я^ц$. Тогда коэффициент перевыполнения норм выработки $K_{п.н.} = T_ц / n_я^ц$.

При установлении явочного состава проходчиков $n_я^ц$ на один цикл следует соблюдать основное правило цикличной организации труда: продолжительность цикла должна быть кратна продолжительности смены. При проведении горизонтальных и наклонных выработок небольшого сечения продолжительность цикла принимается равной как правило 6 или 12 ч, при сооружении вертикальных стволов – 18 или 24 ч.

Сравнение двух или более вариантов организации работ целесообразно производить по комплексной норме времени и комплексной расценке.

8.7.2. Исходные данные, подготавливаемые к расчету

Для расчета по программе "График" необходимо подготовить следующие исходные данные:

Количество нормируемых процессов, шт.

Способ сооружения выработки

Данные по каждому нормируемому процессу:

– объем работ, ед. изм.

– норма времени, чел-ч

– расценка, руб/м

– поправочный коэффициент

Продолжительность цикла, ч

8.7.3. Результаты расчета

В результате выполнения программы "График" на печать выдается таблица результатов по нижеприведенной форме и следующие данные:

№ п/п	Объем работ, ед. изм.	Норма времени, чел- ч	Расценка, руб.	Поправочный. коэфф.	Трудозатраты, чел-ч	Заработная. плата, руб.	Длительность процесса, ч

	Σ				Σ	Σ	

Расчетная скорость проходки, м/ мес

Принятое число проходчиков сменного звена, чел.

Продолжительность взрывных работ, ч

Коэффициент перевыполнения норм выработки

Коэффициент БВР

Комплексная норма времени, чел-ч/м

Комплексная расценка, руб/м

8.8. Программа расчета локальной сметы

Имя программы – "Смета"
Автор программы – Дмитриенко В.А.

8.8.1. Алгоритм расчета

Основанием для составления сметы являются привязанные к местным условиям единичные расценки и объемы работ.

В локальной смете, кроме общей стоимости, рассчитывается и величина нормативной условно-чистой продукции (НУЧП). Для этого по каждому виду работ на основе скорректированных единичных расценок в смете указывается стоимость единицы измерения (общая, в том числе заработной платы и эксплуатации машин). Эти затраты определяются на весь объем работ. Суммируя затраты, получают величину прямых забойных затрат. По итогам забойных затрат определяются общешахтные расходы по установленным для данной стройки нормам. Затем определяется величина прямых забойных затрат (сумма забойных затрат и общешахтных расходов).

Размер накладных расходов определяется по соответствующим нормам к сумме прямых затрат (забойных затрат и общешахтных расходов) в общей стоимости и к сумме забойных затрат в НУЧП.

Плановые накопления определяют по сумме прямых забойных затрат и накладных расходов в общей стоимости и по сумме забойных затрат в НУЧП.

Из итога сметной стоимости и НУЧП исключается сумма снижения за счет применения снижающих коэффициентов.

Далее в смету включаются затраты пропорционально стоимости работ по объекту, входящие в сметную стоимость товарной продукции:

– средства на временные здания и сооружения;

– сметная стоимость прочих работ и затрат (удорожание работ, производимых в зимнее время, затраты на повышение зарплаты среднеоплачиваемых категорий работников, затраты на уборку строительного мусора и др.);

– резерв средств на непредвиденные работы и затраты.

8.8.2. Исходные данные, подготавливаемые к расчету

Для расчета по программе "Смета" необходимо подготовить следующие исходные данные.

Количество нормируемых процессов, шт

Исходные данные на каждый объект:

– объем работ на весь объект, ед. изм.

– прямые затраты на единицу работ, руб.

– основная заработная плата, руб.

– эксплуатация машин и механизмов, руб.

– заработная плата рабочих, обслуживающих машины

– затраты труда на единицу работ

Коэффициент общешахтных расходов

Коэффициент накладных расходов

8.8.3. Таблица результатов расчета

Программа "Смета" выдает результаты расчета по нижеприведенной форме:

ЛОКАЛЬНАЯ СМЕТА

А. ПО ПРОЦЕССАМ

№	Объем работ	На единицу работ					На объект				
		Прямые затраты, руб	Основная зарпл., руб	Эксплуат. машин		Затраты труда, чел.-ч	Общая стоимость, руб.				Затраты труда без обслуж. машин
				Всего	Зрп		Всего	Основн зарпл., руб.	Эксплуат. машин		
									Всего	Зрп	

Б. ПО СТАТЬЯМ ЗАТРАТ И РАСХОДОВ

ИТОГО ПРЯМЫЕ ЗАБОЙНЫЕ ЗАТРАТЫ

В Т.Ч. ОСНОВНАЯ ЗАРАБОТНАЯ ПЛАТА, РУБ. НОРМАТИВНАЯ ТРУДОЕМКОСТЬ, ЧЕЛ-Ч
ОБЩЕШАХТНЫЕ РАСХОДЫ (78%) ИТОГО ПРЯМЫЕ РАСХОДЫ
НАКЛАДНЫЕ РАСХОДЫ (28,3%) ИТОГО СМЕТНАЯ СЕБЕСТОИМОСТЬ
ПЛАНОВЫЕ НАКОПЛЕНИЯ (8%) ИТОГО СМЕТНАЯ СТОИМОСТЬ
ВРЕМЕННЫЕ ЗДАНИЯ И СООРУЖЕНИЯ (9%) ЗИМНЕЕ УДОРОЖАНИЕ (2,1%) НЕПРЕДВИДЕННЫЕ РАБОТЫ И ЗАТРАТЫ (1,5%)
СМЕТНАЯ СТОИМОСТЬ СТРОИТЕЛЬНОЙ ПРОДУКЦИИ, РУБ.

Далее по тексту идут лабораторные работы, основной целью которых является ознакомление студентов с алгоритмом работы вышеописанных прикладных программ на практике. Настоящие программы широко используются студентами старших курсов при курсовом и диплом проектировании, поэтому предлагаемый блок лабораторных работ призван подготовить студентов к дальнейшей самостоятельной работе с программами. Каждый студент выполняет индивидуальный вариант задания по всем предлагаемым работам.

8.9. Лабораторная работа №6

Тема. Использование прикладных программ BWR4-P и BWR4-PS при расчете паспорта буровзрывных работ для проведения горизонтальных и наклонных горных выработок.

Цель работы. Изучить порядок расчета паспорта буровзрывных работ для проведения горизонтальных и наклонных горных выработок по породному или смешанному забою. Освоить программы BWR4-P и BWR4-PS по расчету параметров буровзрывных работ.

Задание. Согласно варианту задания рассчитать параметры паспорта буровзрывных работ для проведения горизонтальной или наклонной выработки, проводимой по породному, угольному или смешанному забою.

8.9.1. Варианты заданий

№ варианта	Наименование выработки	Форма сечения	Основные размеры, м	Крепость пород	Мощность пласта, м	Площадь угольного забоя, $м^2$	Категория по метану

№ варианта	Наименование выработки	Форма сечения	Основные размеры, м	Крепость пород	Мощность пласта, м	Площадь угольного забоя, м^2	Категория по метану
1	Откаточный штрек	Арочная	$L_1 = 5{,}2$ $H_1 = 2{,}7$	8	1,5	8,4	Негазовая
2	Бремсберг	Арочная	$L_1 = 3{,}2$ $H_1 = 2{,}2$	5	1,2	3,84	I
3	Вентиляционный штрек	Прямоугольная	$L_1 = 3{,}2$ $H_1 = 2{,}2$	12	1,8	5,8	I
4	Квершлаг	Трапециевидная	$L_1 = 3{,}7$ $L_2 = 3{,}1$ $H_1 = 2{,}75$	6	–	–	II
5	Полевой штрек	Арочная	$L_1 = 4{,}1$ $H_1 = 2{,}9$	4	–	–	III
6	Ходок уклона	Прямоугольная	$L_1 = 3{,}2$ $H_1 = 2{,}1$	–	2,1	6,72	Сверхкатегорийная
7	Штольня	Арочная	$L_1 = 4{,}9$ $H_1 = 2{,}8$	5	–	–	Негазовая
8	Уклон	Трапециевидная	$L_1 = 3{,}2$ $L_2 = 2{,}5$ $H_1 = 2{,}2$	4	0,8	2,4	II
9	Ходок бремсберга	Арочная	$L_1 = 2{,}7$ $H_1 = 2{,}2$	4	0,6	1,7	I
10	Наклонный ствол	Трапециевидная	$L_1 = 4{,}5$ $L_2 = 4{,}2$ $H_1 = 2{,}95$	9	1,1	4,62	II
11	Вентиляционная сбойка	Трапециевидная	$L_1 = 3{,}2$ $L_2 = 2{,}9$ $H_1 = 2{,}6$	11	–	–	III
12	Порожняковая ветвь околоствольного двора	Арочная	$L_1 = 4{,}2$ $H_1 = 3{,}2$	8	–	–	II
13	Бортовой ходок	Трапециевидная	$L_1 = 3{,}2$ $L_2 = 2{,}8$ $H_1 = 2{,}2$	4	1,0	3,1	III
14	Вентиляционный штрек	Арочная	$L_1 = 3{,}9$ $H_1 = 2{,}6$	7	0,8	2,3	Сверхкатегорийная
15	Квершлаг	Прямоугольная	$L_1 = 4{,}2$ $H_1 = 2{,}8$	10	–	–	II

Примечание. Условные обозначения размеров выработок:

D – диаметр выработки в свету;

L_1, L_2 – ширина соответственно нижнего и верхнего оснований;

H_1 – высота выработки в свету.

Тип взрывчатого вещества выбрать самостоятельно в зависимости от категории шахты по метану. Характеристики патронов задать согласно выбранному ВВ.

8.10. Лабораторная работа №7

Тема. Использование прикладной программы BWR4-S при расчете паспорта буровзрывных работ для проведения вертикальных стволов.

Цель работы. Изучить порядок расчета паспорта буровзрывных работ для проведения вертикального ствола. Освоить программу BWR4-S по расчету параметров буровзрывных работ.

Задание. Согласно варианту задания рассчитать параметры паспорта буровзрывных работ для проведения вертикального ствола.

8.10.1. Варианты заданий

№ варианта	Диаметр ствола в свету, м	Толщина крепи, м	Глубина комплекта шпуров, м	Крепость пород	Угол падения пород, °
1	5	0,25	4	6	10
2	5,5	0,40	4,2	7	12
3	6	0,30	4,3	5	22
4	6	0,35	4	8	6
5	6	0,50	1,3	3	11
6	6,5	0,30	4	9	0
7	7	0,30	4	7	2
8	7	0,15	4	12	6
9	7,5	0,20	4	10	13
10	8	0,30	3	6	29
11	8	0,50	1,5	5	5
12	8,5	0,30	3	6	13
13	9	0,30	3,2	6	18
14	9	0,25	4	7	33
15	9	0,35	3	8	7

8.11. Лабораторная работа №8

Тема. Использование прикладных программ "Воздух-П" и "Воздух-С" при расчете вентиляции тупиковых выработок.

Цель работы. Изучить порядок расчета вентиляции тупиковых горных выработок. Освоить программы "Воздух-П" и "Воздух-С" по расчету параметров вентиляции.

Задание. Согласно варианту задания рассчитать параметры вентиляции тупиковой выработки.

8.11.1 Варианты заданий

№ варианта	Наименование выработки	Длина выработки, м	Площадь поперечного сечения в свету, $м^2$	Количество людей, работающих в выработке	Относительная влажность воздуха, %	Степень обводненности выработки (водоприток, м3/ч)	Количество ВВ, кг,		Длина звена вентиляционных. труб, м
							по углю	по породе	
1	2	3	4	5	6	7	8	9	10
1	Вентиляционный ствол	270	28,3	8	85	7	–	92	3,2
2	Вспомогательный ствол	563	38,5	10	90	5	–	134	4
3	Главный ствол	840	28,3	8	75	1,5	–	102	4
4	Наклонный ствол	1200	17	7	90	Обводненная	18	46	12
5	Штольня	245	15,8	7	75	Сухая	–	58	12
6	Квершлаг	170	12,7	6	90	Частично обводненная	–	44	20
7	Квершлаг	375	15,2	7	60	Сухая	–	62	20
8	Уклон	1100	11,3	5	70	Сухая	24	38	20
9	Вентиляционная сбойка	240	9,5	5	80	Частично обводненная	–	36	12
10	Вентиляционный ствол	930	33,2	11	95	8	–	129	3,2
11	Ходок уклона	980	8,2	5	90	Обводненная	12	25	12
12	Ходок бремсберга	1260	7,9	6	95	Частично обводненная	14	20	20
13	Бремсберг	860	12,5	7	80	Водяные завесы	22	30	12
14	Воздухоподающий ствол	220	23,7	9	80	2	–	120	4
15	Гезенк	115	19,6	8	75	1	–	84	4

Примечание. Расчет произвести для условий Российского Донбасса. Время проветривания после взрывных работ задать самостоятельно согласно требованиям Правил Безопасности. При расчете вентиляции вертикального ствола принять хорошее качество уплотнения стыков вентиляционного трубопровода и использование новых вентиляционных труб.

Для всех вариантов задать следующие исходные данные:

Способ проходки выработки – буровзрывной.

Выемка угля и породы – совместная.

Расстояние от ВМП до устья выработки:

15 м – для горизонтальных и наклонных выработок;

20 м – для вертикальных стволов.

Минимально допустимая скорость по тепловому фактору – 0,15 м/с.

8.12. Лабораторная работа №9

Тема. Использование прикладной программы ARMIR1 для расчета технико-экономических показателей схем армировки вертикального ствола.

Цель работы. Изучить типовые и безрасстрельные схемы армировки клетевых и скиповых стволов и их характеристики. Изучить типы и типоразмеры профилей проводников и расстрелов. Освоить программу ARMIR1 расчета технико-экономических показателей схем армировки.

Задание. Согласно варианту задания выбрать схему армировки вертикального ствола и рассчитать ее технико-экономические показатели.

8.12.1. Варианты заданий

№ варианта	Глубина ствола, м	Диаметр ствола в свету, м	Толщина крепи, мм	Подъемные сосуды	Тип армировки	Глубина заделки расстрелов, мм	Параметры анкеров		Крепость вмещающих пород
							Длина, м	Диаметр, мм	
1	620	6	300	2 клети, клеть аварийно-ремонтного подъема	Типовая	300	–	–	7
2	890	6	500	2 клети, 2 противовеса	Типовая	400	–	–	4
3	135	7	350	2 клети, клеть аварийно-ремонтного подъема	Типовая	300	–	–	5
4	903	7	450	2 клети, 2 противовеса	Безрасстрельная	–	32	0,9	5
5	298	8	300	3 клети, противовес	Типовая	300	–	–	8
6	1230	6	300	3 скипа, противовес	Типовая	300	–	–	8
7	700	6	400	3 скипа, противовес	Безрасстрельная	–	36	1	4
8	508	8	300	3 клети, противовес	Безрасстрельная	–	32	0,8	6
9	622	7	300	4 скипа	Типовая	300	–	–	9
10	950	8	350	3 клети, противовес	Типовая	300	–	–	5
11	260	6	150	3 скипа, противовес	Безрасстрельная	–	28	0,8	11
12	385	6	200	2 клети, 2 противовеса	Безрасстрельная	–	30	1,2	8
13	630	7	300	4 скипа	Безрасстрельная	–	36	1,2	6
14	1048	7	400	4 скипа	Типовая	400	–	–	4
15	303	6	100	3 скипа, противовес	Безрасстрельная	–	32	0,8	12

Примечание. Шаг армировки h принять самостоятельно, при этом обязательно учесть профиль проводника, предусмотренный выбранной схемой ар-

мировки. Для коробчатых проводников стандартными являются следующие значения *h*: 3; 4; 5 и 6 м; для рельсовых – 3,126; 4,168 и 6,252 м.

ЛИТЕРАТУРА

1. Зельднер Г.А. QuickBASIC для носорога. – М.: ABF, 1994. – 480 с.
2. Лапшин Е. Графика для IBM PC. – М.: СОЛОН, 1995. – 228 с.
3. Вяльцев М.М. Технология строительства горных предприятий в примерах и задачах: Учеб. пособие для вузов. – М.: Недра, 1989. – 240 с.
4. Специальные способы строительства в примерах и задачах: Учеб. пособие/ М.М. Вяльцев, Ю.А. Полозов, Ю.Н. Спичак; Новочерк. политехн. ин-т. – Новочеркасск, 1990. – 87 с.
5. Технология строительства наклонных выработок: Учеб. пособие/ И.А. Мартыненко, П.С. Сыркин, М.С. Данилкин и др.: Новочерк. гос. техн. ун-т. – Новочеркасск, 1994. – 112 с.

ПРИЛОЖЕНИЕ 1

ЗАРЕЗЕРВИРОВАННЫЕ СЛОВА ЯЗЫКА QBASIC

В этом приложении приведена таблица, содержащая список зарезервированных слов языка QBasic. Эти слова не могут быть использованы в качестве меток или имен переменных. Словарь включает ключевые слова языка, а также служебные инструкции, которые обрабатываются средой QBasic и компилятором. В таблице приведена краткая расшифровка ключевого слова языка QBasic и дается приблизительный перевод. Ключевые слова, значение которых рассматривается в настоящем пособии, выделены жирным шрифтом.

Ключевые слова и их расшифровка	Перевод
ABS (ABSOLUTE)	абсолютное значение
ACCESS	доступ
ALIAS	замена
AND	и
ANY	любой
APPEND	добавление в конец (файла)
AS	как
ASC (ASCII code)	код из таблицы ASCII
ATN	арктангенс
BASE	точка отсчета
BEEP	сигнал
BINARY	двоичный (файл)
BLOAD (Binary LOAD)	загрузка образа (памяти)
BSAVE (Binary SAVE)	запись образа (памяти)
BYVAL (BY VALue)	(передача) по значению
CALL	вызов (BASIC-процедуры)
CALLS	вызов (не BASIC-процедуры)
CASE	случай
CDBL (Conversion number to DouBLe)	преобразование числа к типу удвоенной точности
CDECL (C-language DECLare)	объявить процедуру, использующую передачу параметров по правилам языка С
CHAIN	передать управление (файлу)
CHDIR (CHange DIRectory)	сменить директорию
CHR$ (CHaracteR)	символ
CINT (Conversion number to INTeger)	преобразование числа к целому типу
CIRCLE	окружность
CLEAR	обнулять
CLNG (Conversion to LoNG)	преобразовать к длинному целому типу
CLOSE	закрыть (файл)

Ключевые слова и их расшифровка	Перевод
CLS (CLear Screen)	очистить экран
COLOR	цвет
COM (COMmunicate port)	параллельный порт
COMMAND$	командная строка
COMMON	общая (переменная/ группа переменных)
CONST (CONSTant)	константа
COS	косинус
CSNG (Conversion to SINGle)	преобразовать к длинному целому типу
CSRLIN (CurSoR LINE)	строка, на которой находится курсор
CVD (ConVert string to Double)	представить строку как число удвоенной точности
CVDMBF (ConVert Double in Microsoft Binary Format)	представить строку в формате Microsoft Binary как число удвоенной точности
CVI (ConVert string to Integer)	представить строку как целое число
CVL (ConVert string to Long)	представить строку как длинное целое число
CVS (ConVert string to Single)	представить строку как число обычной точности
CVSMBF (ConVert Single in Microsoft Binary Format)	представить строку в формате Microsoft Binary как число обычной точности
DATA	данные
DATE$	текущая дата (ММ-ДД-ГГГГ)
DECLARE	объявить (процедуру/ функцию)
DEF (DEfine Fn function)	определить функцию FN
DEFDBL (DEFine DouBle)	определить тип данных удвоенной точности
DEFINT (DEFine INTeger)	определить целый тип данных
DEFLNG (DEFine LoNG)	определить длинный целый тип данных
DEFSNG (DEFine SiNGle)	определить тип данных обычной точности
DEFSTR (DEFine STRing)	определить символьный тип данных
DIM (DIMension)	резервирование места (для массива/ переменной)
DO	делать
DOUBLE	двойной точности (переменная)
DRAW	рисовать
ELSE	иначе
ELSEIF (ELSE IF)	иначе, если…
END	конец (программы/ процедуры/ функции или управляющей конструкции)
ENDIF (END IF)	конец условия IF
ENVIRON (ENVIRONment dos change)	изменить переменную окружения DOS
ENVIRON$ (ENVIRONment dos get)	получить значение переменной DOS
EOF (End Of File)	конец файла
EQV (EQuiValence)	эквивалентность
ERASE	стереть содержимое или уничтожить массив

Ключевые слова и их расшифровка	Перевод
ERDEV (Error on DEVice code)	код внешнего устройства, вызвавшего ошибку
ERDEV$ (Error on DEVice name)	название внешнего устройства, вызвавшего ошибку
ERL (ERror Line)	номер строки, которая вызвала ошибку
ERR (ERRor code)	код ошибки
ERROR	ошибка
EXIT	выход (из процедуры/ функции, управляющей конструкции)
EXP	экспонента
FIELD	поле (файла прямого доступа)
FILEATTR (FILE ATTRibute)	атрибут файла
FILES	файлы
FIX (FIXed)	округление
FOR	для
FRE (FREe memory)	свободная область памяти
FREEFILE (FREE FILE number)	свободный номер файла
FUNCTION	функция
GET	взять
GOSUB (GO SUBroutine)	перейти к подпрограмме
GOTO (GO TO)	перейти к
HEX$ (HEXadecimal string)	шестнадцатеричная строка
IF	если
IMP (IMPlication)	импликация
INKEY$ (INput KEY)	ввести символ
INP (INput from Port)	прочитать байт из порта ввода/ вывода
INPUT	ввести (переменную)
INPUT$ (INPUT string)	ввести строку символов
INSTR (IN STRing position)	позиция в строке
INT (INTeger)	целое (число)
INTEGER	целое (тип данных)
IOCTL (Input/Output ConTroL data string to device driver)	послать управляющую строку драйверу устройства
IOCTL$ (Input/Output ConTroL data string from device driver)	принять управляющую строку из драйвера устройства
IS	если
KEY	клавиша
KILL	уничтожить (файл)
LBOUND (Lower BOUND)	нижняя граница массива
LCASE$ (Lower-CASE string)	строка в нижнем регистре
LEFT$	слева
LEN	длина
LET	пусть будет

Ключевые слова и их расшифровка	Перевод
LINE	линия
LIST	список (функциональных клавиш)
LOC (Location Of Current Position)	место текущей позиции (в файле)
LOCAL	местный
LOCATE	разместить
LOCK	заблокировать (файл)
LOF (Len Of File)	длина файла
LOG	логарифм
LONG	длинное целое (тип данных)
LOOP	пока
LPOS (Line Printer's head position)	позиция печатающей головки принтера
LPRINT (Line printer PRINT)	печатать на принтер
LSET (Left SET)	присвоить слева (для файлов прямого доступа)
LTRIM$ (Left TRIM)	подровнять (убрать пробелы) слева
MID$ (MIDdle)	середина
MKD$ (MaKe Double to string)	превратить число удвоенной точности в строку
MKDIR (MaKe DIRectory)	сделать директорию
MKDMBF (MaKe Double to string in Microsoft Binary Format)	превратить число удвоенной точности в строку в формате Microsoft Binary
MKI$ (MaKe Integer to string)	превратить целое число в строку
MKL$ (MaKe Long to string)	превратить длинное целое число в строку
MKS$ (MaKe Single to string)	превратить число обычной точности в строку
MKSMBF$ (MaKe Single to string in Microsoft Binary Format)	превратить число обычной точности в строку в формате Microsoft Binary
MOD	модуль
NAME	имя
NEXT	следующий
NOT	нет
OCT$ (OCTal)	восьмеричный
OFF	выключить (отслеживание событий или строку функциональных клавиш)
ON	включить (отслеживание событий или строку функциональных клавиш)
OPEN	открыть (файл или устройство)
OPTION	начало (точки отсчета)
OR	или
OUT	считать файл из порта вода/ вывода
OUTPUT	вывести
PAINT	закрасить
PALETTE	палитра
PCOPY (Page COPY)	скопировать экранную страницу
PEEK	прочесть байт по машинному адресу

Ключевые слова и их расшифровка	Перевод
PEN	световое перо
PLAY	играть (ноты)
PMAP (Point MAP)	преобразование физических координат экрана в логические и наоборот
POINT	точка
POKE	записать байт по машинному адресу
POS (POSition)	позиция
PRESET (Point RESET)	стереть точку
PRINT	напечатать
PSET (Point SET)	вывести точку
PUT	поместить
RANDOM	прямой (доступ к файлу)
RANDOMIZE	случайный
READ	считать
REDIM (REDIMension)	повторно определить размерность
REM (REMark)	комментарий
RESET	закрыть (все открытые файлы)
RESTORE	восстановить
RESUME	продолжить
RETURN	возвращение
RIGHT$	справа
RMDIR (ReMove DIRectory)	удалить директорию
RND (RaNDom)	случайным образом
RSET (Right SET)	присвоить слева (для файлов прямого доступа)
RTRIM$ (Right TRIM)	подровнять (убрать пробелы) справа
RUN	запустить
SADD (String ADDress)	адрес строки (в памяти)
SCREEN	экран
SEEK	установить позицию (в файле)
SEG (SEGment)	сегмент (памяти)
SELECT	выбрать
SETMEM (SET MEMory)	установить размер динамической области памяти
SGN (SiGNum)	знак
SHARED	разделяемый
SHELL	временный выход
SIGNAL	сигнал
SIN	синус
SINGLE	обычной точности (тип данных)
SLEEP	задержка
SOUND	звук
SPACE$ (SPACE string)	строка пробелов

Ключевые слова и их расшифровка	Перевод
SPC (Skip sPaCes)	пропустить пробелы
SQR	квадратный корень
STATIC	локальная (переменная)
STEP	шаг
STICK (Specified joySTICK)	управление джойстиком
STOP	стоп
STR$ (STRing)	строка
STRIG (Specified jojstick TRIGger)	нажатие кнопки джойстика
STRING	символьное (тип данных)
STRING$	строка символов
SUB	процедура
SWAP	обмен
SYSTEM	система (выход в DOS)
TAB	табулятор
TAN	тангенс
THEN	тогда
TIME$	время
TO	до
TROFF (TRace OFF)	выключить трассировку
TRON (TRace ON)	включить трассировку
TYPE	тип
UBOUND (Upper BOUND)	верхняя граница (массива)
UCASE$ (Upper-CASE sting)	строка в верхнем регистре
UNLOCK	разблокировать (файл)
UNTIL	до тех пор, пока
USING	используются
VAL (VALue)	значение
VARPTR	вычисление адреса переменной
VARPTR$	строковое представление адреса переменной
VARSEG	вычисление сегмента адреса переменной
VIEW	обзор
WAIT	ждать
WEND	конец цикла
WHILE	пока
WIDTH	ширина
WINDOW	окно
WRITE	вывести
XOR	неэквивалентность

ПРИЛОЖЕНИЕ 2

КОДЫ ОШИБОК

В QBasic существуют два основных типа ошибок: ошибки *периода компиляции* и *периода выполнения*. Первые – это в основном синтаксические ошибки, обнаруженные компилятором, вторые – это ошибки, выявленные соответствующими средствами их обнаружения, помещенными компилятором в объектный код программы.

Большинство ошибок периода компиляции это синтаксические ошибки: пропуск символов, описки в операторах, недостающие скобки и т. д. Если компилятор находит в исходной программе недопустимые конструкции, которые он не может разрешить или понять, то QBasic автоматически переходит в режим редактирования, причем курсор указывает на место, где обнаружена ошибка. Чтобы удалить сообщение об ошибке необходимо нажать любую клавишу, а затем исправить ошибку и повторно запустить программу на выполнение.

Ошибки периода выполнения происходят при исполнении программы. Это могут быть, например, ошибки файловой системы (диск переполнен или защищен от записи), неверные вызовы функций (использование графических функций без графического адаптера), попытка извлечения квадратного корня из отрицательного числа, ошибки памяти (как правило, переполнение), и многие другие.

Ниже приведены коды и сообщения о наиболее часто встречающихся ошибках (с переводом), а также сделаны некоторые пояснения и указаны основные причины возникновения таких ошибок.

Ошибки периода выполнения

2 ***Syntax error*** (синтаксическая ошибка)
Синтаксическая ошибка периода выполнения возникает при попытке загрузить символьные данные в числовую переменную с помощью оператора READ. Другие синтаксические ошибки обнаруживаются компилятором.

3 **RETURN without GOSUB** (RETURN без GOSUB).
Встретился оператор RETURN без предварительного выполнения оператора GOSUB (вне вызванной подпрограммы); т.е. система не знает, откуда делать возврат.

4 **Out of data** (отсутствие данных).
Число аргументов в операторе READ больше числа аргументов в соответствующем операторе DATA.

5 **Illegal fuction call** (неверный вызов функции).
Это обобщенная ошибка, связанная с передачей неправильного аргумента некоторому оператору или функции. Вот некоторые из многих причин, вызывающих данную ошибку:
– слишком большое значение кода цвета или экранного режима;

– использование графического оператора без графического адаптера или без установки соответствующего режима оператором SCREEN;
– попытка выполнить недопустимую математическую операцию, например, извлечение квадратного корня из отрицательного числа.

6 **Overflow** (переполнение).
Переполнение возникает в результате вычислений, когда полученное число слишком велико, чтобы его можно было представить указанным числовым типом

7 **Out of memory** (отсутствие памяти).
Это сообщение появляется во многих случаях, например, при определении массива слишком большого размера.

9 **Subscript out of range** (индекс вне диапазона).
Значение индекса массива превосходит максимальную величину, указанную при описании массива.

11 **Division by zero** (деление на нуль).
Деление на нуль или возведение нуля в отрицательную степень.

13 **Type mismatch** (несоответствие типа).
Использование символьного значения вместо числового или наоборот. Это может произойти в операторах PRINT USING или DRAW.

15 **String too long** (слишком длинная строка).
Строка, полученная при вычислении символьного выражения, превышает 32767 байт.

25 **Device fault** (отказ устройства).
Аппаратная ошибка; например, неисправность интерфейса принтера или адаптера связи.

27 **Out of paper** (отсутствие бумаги).
Контроллер принтера фиксирует отсутствие в нем бумаги. Он может быть просто выключен или неисправен.

52 **Bad file number** (неверный номер файла).
Номер файла, заданный в операторе работы с файлом, не был определен в операторе OPEN или этот номер превышает максимально допустимое значение.

53 **File not found** (файл не найден).
Файл с заданным именем на указанном устройстве не найден.

55 **File already open** (файл уже открыт).
Попытка открыть ранее открытый файл или удалить открытый файл.

61 **Disk full** (диск заполнен).
Нет достаточного места на указанном или текущем диске при выполнении файловой операции. Необходимо обеспечить требуемый объем свободного пространства на диске и перезапустить программу.

62 **Input past end** (ввод после конца).
Попытка считать из файла больше данных, чем он содержит.

64 **Bad file name** (неверное имя файла).
Попытка присвоить недопустимое имя файлу, (например, в имени файла присутствуют недопустимые символы.)

71 **Disk not ready** (диск не готов).
Не закрыт дисковод или не вставлен диск.

72 **Disk media error** (ошибка на диске).
Обнаружены плохие сектора (физические повреждения) на гибком или жестком диске.

76 **Path not found** (путь не найден).
Указанный путь не найден (например, в команде OPEN и т.д.).

Ошибки периода компиляции

407 **Program too large** (программа слишком велика).
Программа содержит более 65530 операторов.

410 **"," expected** (ожидается ",").
По синтаксису оператора требуется запятая (,).

411 **";" expected** (ожидается ";").
По синтаксису оператора требуется точка с запятой (;).

412 **"(" expected** (ожидается "(").
Пропущена правая скобка (().

413 **")" expected** (ожидается ")").
Пропущена левая скобка ()).

414 **"=" expected** (ожидается "=").
Пропущен знак равенства (=).

415 **"-" expected** (ожидается "-").
Пропущен знак дефиса (-).

416 **Statement expected** (ожидается оператор).
Требуется оператор QBasic

417 **Label/line number expected** (ожидается метка или номер строки).
В операторе IF, GOTO, GOSUB или ON отсутствует метка или номер строки.

432 **AS expected** (ожидается AS).
Пропущено зарезервированное слово AS в операторе OPEN.

433 **DEF FN expected** (ожидается DEF FN).
Встречен оператор END FN или EXIT FN вне описания функции. Описание функции должно начинаться с оператора DEF FN.

434 **IF expected** (ожидается IF).

Для операторов END IF или EXIT IF нет соответствующего оператора IF.

435 DO LOOP expected (ожидается цикл DO)
Для операторов LOOP или EXIT LOOP нет соответствующего оператора DO, открывающего тело цикла.

436 SELECT expected (ожидается SELECT).
В операторе SELECT CASE пропущено зарезервированное слово SELECT, либо для операторов END SELECT или EXIT SELECT нет соответствующего оператора SELECT CASE. Эта ошибка может быть вызвана также использованием зарезервированного слова CASE в качестве имени переменной в программе.

437 CASE expected (ожидается CASE).
В операторе SELECT CASE пропущено зарезервированное слово CASE. Эта ошибка может быть также вызвана использованием зарезервированного слова SELECT в качестве имени переменной в программе.

438 FOR LOOP expected (ожидается цикл FOR).
Для оператора EXIT FOR нет соответствующего оператора FOR, открывающего тело цикла.

439 SUB expected (ожидается SUB).
Операторы END SUB или EXIT SUB находятся вне подпрограммы. Подпрограмма должна начинаться оператором SUB.

440 END DEF expected (ожидается END DEF).
Отсутствует оператор END DEF, завершающий описание функции.

441 END IF expected (ожидается END IF).
Отсутствует оператор END IF, завершающий блок IF.

442 LOOP/WEND expected (ожидается LOOP или WEND).
Отсутствуют операторы LOOP или WEND, завершающие циклы DO или WHILE.

443 END SELECT expected (ожидается END SELECT).
Отсутствует оператор END SELECT в конструкции SELECT CASE.

444 END SUB expected (ожидается END SUB).
Отсутствует оператор END SUB, завершающий описание подпрограммы.

445 NEXT expected (ожидается NEXT).
Отсутствует оператор NEXT, завершающий цикл FOR.

446 THEN expected (ожидается THEN).
В операторе IF нет соответствующей части THEN.

447 TO expected (ожидается TO).
Отсутствует TO в конструкции FOR.

448 GOSUB expected (ожидается GOSUB).
В операторе ON нет соответствующей части SUB.

449 **GOTO expected** (ожидается GOTO).
В операторе ON нет соответствующей части GOTO.

456 **Undefined label/line reference** (ссылка на неопределенную метку или строку).
Неопределенная метка или номер строки в операторах IF, GOTO, GOSUB или ON. Необходимо исправить метку или номер строки или определите его.

458 **Duplicate label/line number** (двойная метка или номер строки).
Повторное определение метки или номера строки. Необходимо найти одинаковые метки или номера строк в основной программе и подпрограммах и исправить их.

460 **Duplicate function definition** (двойное определение функции).
Описаны две функции оператором DEF FN с одинаковыми именами.

467 **Invalid line number** (неправильный номер строки).
Номера строк должны лежать в диапазоне от 0 до 65535.

471 **Unknown identifier/syntax error** (неопределенный идентификатор или синтаксическая ошибка).
Ошибка в строке – компилятор не смог точно определить ее причину.

АЛФАВИТНЫЙ УКАЗАТЕЛЬ

ОГЛАВЛЕНИЕ

Учебное издание

Альберт Юрьевич Прокопов
Иван Андреевич Мартыненко
Сергей Георгиевич Страданченко
Николай Викторович Титов
Есмак Михайлович Красунцев
Николай Константинович Вершинин

Шахтное и подземное строительство.
Решение практических задач на ЭВМ.

Редакторы: Н.А. Юшко
Ж.В. Паршина

ЛР № 020417 12.02.97 г. Подписано в печать 25.05.99 г. Формат 60×84 1/16
Бумага офсетная. Печать оперативная. Усл. п.л. 10,0. Уч.-изд. л. 10,5. Усл. кр.-отт. 10,1.
Тираж 150. С 156. Заказ №

Южно-Российский государственный технический университет
Редакционно-издательский отдел ЮРГТУ
346428, Новочеркасск, ул. Просвещения, 132.
Шахтинский институт ЮРГТУ
346500, Шахты, пл. Ленина, 1