Оглавление

5. Варианты заданий

Каждый вариант задания представляет собой пару: входной язык и машинный, или выходной, язык (см. табл. 5.1). В качестве входного языка предлагается один из языков программирования высокого уровня, в который должно быть обязательно включено следующее подмножество языковых конструкций и операторов:

- идентификаторы;
- числовые константы целого типа и вещественного типа, представленные с фиксированной и плавающей точкой;
- символьные (строковые) константы;
- переменные с индексами (массивы и элементы массивов);
- комментарии (строчные и блочные);
- имена функций пользователя;
- арифметические операции;
- операции сравнения (меньше, больше, равно, не равно, меньше или равно, больше или равно);
- операторы описания данных (идентификаторов и массивов);
- операторы описания процедур и функций (если предусмотрены в языке);
- операторы условного и безусловного перехода;
- метки (если предусмотрены в языке).

Описание синтаксиса предлагаемых входных языков приведено в разделе 6. Допускается выбор студентом иного диалекта входного языка (например, VisualBasic вместо TurboBasic). В этом случае студент должен отразить это в отчете дать свое описание синтаксиса языка.

В качестве машинного языка предлагаются различные языки высокого уровня и язык Ассемблера (см. табл. 5.1).

Таблица 5.1

№ варианта	Входной язык	Выходной язык
1	Паскаль	Си
2	Бейсик	Си
3	Perl	Си
4	Фортран	Си
5	JavaScript	Си
6	Паскаль	Perl
7	Бейсик	Perl
8	Си	Perl
9	Фортран	Perl
10	JavaScript	Perl
11	Паскаль	Фортран
12	Бейсик	Фортран
13	Perl	Фортран
14	Си	Фортран
15	JavaScript	Фортран
16	Паскаль	Бейсик
17	Си	Бейсик
18	Perl	Бейсик
19	Фортран	Бейсик
20	JavaScript	Бейсик
21	Паскаль	JavaScript
22	Бейсик	JavaScript
23	Perl	JavaScript
24	Фортран	JavaScript
25	Си	JavaScript
26	Паскаль	Ассемблер
27	Си	Ассемблер
28	Бейсик	Ассемблер
29	Perl	Ассемблер
30	Фортран	Ассемблер

Таблица 5.1

№ варианта	Входной язык	Выходной язык
31	JavaScript	Ассемблер
32	Си	Паскаль
33	Бейсик	Паскаль
34	Perl	Паскаль
35	Фортран	Паскаль
36	JavaScript	Паскаль

Приступая к выполнению лабораторных работ, студент должен внимательно ознакомиться со своим вариантом задания, сформировать согласованное с преподавателем подмножество конструкций выбранной версии входного языка и используемые инструментальные средства разработки.

6. Описание языков программирования, выбранных в качестве входных языков

6.1. Язык программирования Паскаль

(TurboPascal 7.0)

Идентификаторы

Произвольная последовательность букв и цифр, начинающаяся с буквы. Может включать символы подчеркивания.

Числовые константы целого типа

Произвольная последовательность цифр без знака.

Числовые константы вещественного типа, представленные с фиксированной точкой

Последовательность цифр, включающая одну десятичную точку вида

```
123.45
.25
25.
```

Числовые константы вещественного типа, представленные с плавающей точкой

Последовательность, включающая цифры, десятичную точку (необязательную), символ «e» или «E», а также знак «+» или «-» вида (необязательный):

```
1.23e-25
1.23E-25
1.23e+25
1.23E+25
1.23e2
1.23E2
.25e-5
.25E+5
1E-78
1e67
```

Символьные (строковые) константы

Последовательность символов, заключенная в апострофы, расположенная в пределах одной строки, вида:

```
'acb 12_& ?tu'
```

Переменные с индексами (массивы и элементы массивов)

Идентификатор, после которого в квадратных скобках через запятую перечислены выражения-индексы, вида:

```
Abc[12, I, i-6]
C[1+i]
```

Комментарии (строчные и блочные)

Только блочные – последовательность символов, заключенная в фигурные скобки, возможно содержащая несколько строк:

```
{ Это комментарий,
Который содержит 2 строки}
```

Обращения к процедурам и функциям пользователя

Идентификатор, после которого в круглых скобках следует последовательность выражений-аргументов, разделенных запятыми. Отсутствие аргументов не допускается:

```
F(12, 4, i)
f(av-6)
```

Арифметические операции

Сложение	+
Вычитание	-
Умножение	*
Деление	/
Возведение в степень	^

Операции сравнения

Меньше	<
Больше	>
Равно	=
Не равно	<>
Меньше или равно	<=
Больше или равно	>=

Оператор присваивания

Имеет вид «:=». Слева стоит идентификатор или элемент массива, а справа – выражение. Заканчивается символом «;», например:

```
a:=b+c;
b[2,i-9]:=12;
```

Операторы блока

```
Begin – начало блока
…
End; – конец блока
```

Оператор описания программы

Программа начинается оператором Program с указанием имени программы. Затем могут идти описания данных, процедур и функций, а затем тело программы, заключенное в операторы блока, оканчивающееся точкой.

```
Program <идентификатор>;
…
Begin
…
End.
```

Операторы описания данных (идентификаторов и массивов)

Начинается оператором Var и может содержать несколько строк описаний, состоящих из перечисления идентификаторов через запятую и после двоеточия ключевое слово типа.

```
Var
    A,b: real;
    C: integer;
```

Типы переменных: integer (целый), real (вещественный), string (строковый)

Для массивов после двоеточия указывается ключевое слово массива «array of», в квадратных скобках через запятую перечисляются границы изменения каждого из индексов, разделенные символами «..», и затем тип элементов:

```
Var
a,b,c : array of [1..3, 10..20] of integer;
```

Операторы описания процедур и функций

Процедуры имеют заголовок вида

```
procedure <идентификатор> (<список формальных параметров>);
```

и тело – список операторов, заключенный в операторы блока

```
begin … end;
```

Между заголовком и телом может присутствовать оператор описания данных Var. Например:

```
procedure abc (r: real);
var
r1,r2:real;
begin
y:=sinr(r1)/cos(r2)*tan(r);
end;
```

Функции имеют заголовок вида:

function <идентификатор> (<список формальных параметров>): <тип возвращаемого значения>;

В остальном структура функций аналогична структуре процедур. Исключение составляет обязательное присутствие в теле функции хотя бы одного оператора return <значение>;

Оператор безусловного перехода и метки

goto <метка> ;

Метка - идентификатор, расположенный в теле программы в начале строки, после которого стоит знак «:»:

```
a: str:='ujhti';
```

Оператор условного перехода

Начинается с ключевого слова «if», имеет полный и неполный формат:

If <условие> then <оператор_1> else <оператор_2>;

If <условие> then <оператор_1>;

В качестве операторов могут использоваться блоки операторов.

6.2. Язык программирования Java Script

(ECMA-262 - Netscape)

Идентификаторы

Произвольная последовательность букв и цифр, начинающаяся с буквы. Может включать символы подчеркивания.

Числовые константы целого типа

Произвольная последовательность цифр без знака.

Числовые константы вещественного типа, представленные с фиксированной точкой

Последовательность цифр, включающая одну десятичную точку вида

123.45

.25

25.

Числовые константы вещественного типа, представленные с плавающей точкой

Последовательность, включающая цифры, десятичную точку (необязательную), символ «e» или «E», а также знак «+» или «-» вида (необязательный):

```
1.23e-25
1.23E-25
1.23e+25
1.23E+25
1.23e2
1.23E2
1E-78
.25e-5
.25E+5
1e67
```

Символьные (строковые) константы

Набор символов, возможно пустой, заключенный в апострофы или кавычки:

```
"Это строковая константа"
'Это тоже строковая константа'
```

Переменные с индексами (массивы и элементы массивов)

Идентификатор, после которого в квадратных скобках стоит выражение-индекс, вида:

```
Abc[12]
C[1+i]
```

В языке используются только одномерные массивы.

Комментарии (строчные и блочные)

Только строчные – последовательность символов от знаков «//» до конца строки,

```
X=45 //Это комментарий
```

Обращения к функциям пользователя

Идентификатор, после которого в круглых скобках следует последовательность выражений-аргументов, разделенных запятыми. Отсутствие аргументов не допускается:

```
F(12, 4, i)
f(av-6)
```

Арифметические операции

Сложение	+
Вычитание	-
Умножение	*
Деление	/
Возведение в степень	^

Операции сравнения

Меньше	<
Больше	>

Равно	==
Не равно	!=
Меньше или равно	<=
Больше или равно	>=

Оператор присваивания

Имеет вид «=». Слева стоит идентификатор или элемент массива, а справа – выражение. Заканчивается символом «;», например:

```
a=b+c;
b[2]=12;
```

Операторы блока

```
{ – начало блока
…
} – конец блока
```

Операторы описания данных (идентификаторов и массивов)

Начинается оператором Var и может содержать список идентификаторов через запятую. Типы отсутствуют.

```
var <имя переменной>;
var x,y,z;
```

Для одномерных массивов после идентификатора указывается ключевое слово массива « = new array» и в круглых скобках указывается количество элементов:

```
var <идентификатор> = new array(количество элементов)
var c = new array(100)
```

Многомерные массивы в языке JavaScript отсутствуют.

Операторы описания функций

Процедуры имеют заголовок вида

```
function <идентификатор> (<список формальных параметров>);
```

и тело – список операторов, заключенный в операторы блока

```
{ … }
```

Например:

```
function sum(a,b)
{
var y;
y=a+b;
return y;
}
```

Оператор безусловного перехода и метки

```
goto <метка> ;
```

Метка - идентификатор, расположенный в теле программы в начале строки, после которого стоит знак «:»:

```
a: str:='ujhti';
```

Оператор условного перехода

Начинается с ключевого слова «if», имеет полный и неполный формат:

```
if (логическое выражение)
{
операторы
}
```

или

```
if (логическое выражение)
{
Операторы_1
}
else
{
Операторы_2
}
```

6.3. Язык программирования Basic

(Microsoft Turbo Basic)

Идентификаторы

Произвольная последовательность букв и цифр, начинающаяся с буквы. Последний символ имени определяет тип идентификатора:

```
a$ – символьный
a% – целый
a& – длинный целый
a! – вещественный обычной точности
a# – вещественный двойной точности
```

Числовые константы целого типа

Произвольная последовательность цифр без знака.

Числовые константы вещественного типа, представленные с фиксированной точкой

Последовательность цифр, включающая одну десятичную точку вида

```
123.45
.25
25.
```

Числовые константы вещественного типа, представленные с плавающей точкой

Последовательность, включающая цифры, десятичную точку (необязательную), символ «e» или «E», а также знак «+» или «-» вида (необязательный):

```
1.23e-25
1.23E-25
1.23e+25
1.23E+25
1.23e2
1.23E2
```

```
1E-78
.25e-5
.25E+5
1e67
```

Символьные (строковые) константы

Последовательность символов, заключенная в кавычки, расположенная в пределах одной строки, вида:

“acb 12_& ?tu”

Переменные с индексами (массивы и элементы массивов)

Идентификатор, после которого в круглых скобках через запятую перечислены выражения-индексы, вида:

```
Abc%(12, I, i-6)
C$(1+i)
```

Комментарии (строчные и блочные)

Только строчные – строка, начинающаяся с оператора «REM».

```
REM Это комментарий
I%=12
```

Обращения к функциям пользователя

Идентификатор, после которого в круглых скобках следует последовательность выражений-аргументов, разделенных запятыми. Отсутствие аргументов не допускается:

```
M=F(12, 4, i)
f(av-6)
```

Арифметические операции

Сложение	+
Вычитание	-
Умножение	*
Деление	/
Возведение в степень	**

Операции сравнения

Меньше	<
Больше	>
Равно	=
Не равно	<>
Меньше или равно	<=
Больше или равно	>=

Оператор присваивания

Имеет вид «=». Слева стоит идентификатор или элемент массива, а справа – выражение. Заканчивается символом «;», например:

```
a%=b%+c%
b$(2,i%-9)=”12”
```

Оператор останова

Прерывает выполнение программы:

```
STOP
```

Оператор окончания программы/процедуры

Завершает текст модуля:

```
END
```

Операторы описания массивов

Описание массивов осуществляется с помощью оператора DIM с указанием размеров. Например, оператор

```
DIM a(10), b(10:20, 25:45)
```

описывает одномерный массив a, элементы которого имеют индексы от 0 до 10, и двухмерный массив b, элементы которого имеют индексы: первый от 10 до 20, второй от 25 до 45.

Если нижняя граница индексов в описании не указана, то она считается равной 0.

В описании массива вместо константы может использоваться переменная. Например,

```
DIM a(n)
```

Значение n должно быть предварительно определено.

Оператор безусловного перехода и метки

```
goto <метка> ;
```

Метка - идентификатор, расположенный в теле программы в начале строки, после которого стоит знак «:»:

```
a: str$=”ujhti”;
```

Операторы описания процедур и функций

Подпрограмма - помеченная меткой последовательность операторов, оканчивающаяся оператором RETURN. Выполняется, когда достигнут оператор GOSUB.

```
....
GOSUB aa
....
END
aa:
 <операторы>
RETURN
```

Оператор RETURN осуществляет возврат к оператору, непосредственно следующему за GOSUB.

Оператор условного перехода

Начинается с ключевого слова «if», имеет полный и неполный формат:

```
IF <условие> THEN <оператор1> [ELSE <оператор2>]
```

Например:

```
IF a < b THEN t=15 ELSE t=17
```

Если после THEN или после ELSE располагается целая группа операторов, то можно использовать IF-блок, который имеет следующую структуру:

```
IF <условие> THEN
      <операторы1>
ELSE
```

```
        <операторы2>
    END IF
```

При этом ELSE и операторы за ним могут отсутствовать, т.е. возможна конструкция

```
    IF  <условие> THEN
        <операторы>
    END IF
```

6.4. Язык программирования C++

(ISO/IEC 14882)

Идентификаторы

Произвольная последовательность букв и цифр, начинающаяся с буквы. Может включать символы подчеркивания и начинаться с них.

Числовые константы целого типа

Произвольная последовательность цифр без знака.

Числовые константы вещественного типа, представленные с фиксированной точкой

Последовательность цифр, включающая одну десятичную точку вида

```
123.45
.25
25.
```

Числовые константы вещественного типа, представленные с плавающей точкой

Последовательность, включающая цифры, десятичную точку (необязательную), символ «e» или «E», а также знак «+» или «-» вида (необязательный):

```
1.23e-25
1.23E-25
1.23e+25
1.23E+25
.25e-5
.25E+5
1.23e2
1.23E2
```

Символьные (строковые) константы

Символьная константа – один символ, заключенный в апострофы:

```
'a'
```

Строковая константа - последовательность символов, заключенная в кавычки, расположенная в пределах одной строки, вида:

```
"acb 12_& ?tu"
```

Переменные с индексами (массивы и элементы массивов)

Идентификатор, после которого в квадратных скобках перечислены выражения-индексы, вида:

```
Abc[12][I][i-6]
C[1+i]
```

Комментарии (строчные и блочные)

Блочный комментарий – последовательность символов, начинающаяся с «/*» и оканчивающаяся «*/», возможно содержащая несколько строк:

```
/* Это комментарий,
который содержит 2 строки*/
```

Строчные – от символов «//» до конца строки.

```
i=i+1; // это инкремент
```

Обращения к функциям пользователя

Идентификатор, после которого в круглых скобках следует последовательность выражений-аргументов, разделенных запятыми. Скобки могут быть пустыми в случае отсутствия аргументов:

```
F(12, 4, i)
f(av-6)
g()
```

Арифметические операции

Сложение	+
Вычитание	-
Умножение	*
Деление	/

Операции сравнения

Меньше	<
Больше	>

Равно	==
Не равно	!=
Меньше или равно	<=
Больше или равно	>=

Оператор присваивания

Имеет вид «=». Слева стоит идентификатор или элемент массива, а справа – выражение. Заканчивается символом «;», например:

```
a=b+c;
b[2][i-9]=12;
```

Операторы блока

```
{ – начало блока
…
} – конец блока
```

Структура программы

Программа начинается операторами описания данных. Затем могут идти описания данных и функций, а затем основная функция программы void main () и ее тело, заключенное в операторы блока, оканчивающееся точкой.

```
Описания
void main()
{
…
}
```

Операторы описания данных (идентификаторов и массивов)

Начинается с ключевого слова типа и содержит перечисление идентификаторов через запятую. Оканчивается знаком «;»

```
<тип> <список элементов>;
```

Типы переменных: int (целый), float (вещественный), char (символьный).

Элементом списка может быть массив, для которого указывается идентификатор и размерности:

```
int a,b,c;
float d[3][4], c[78];
```

Операторы описания функций

Функции имеют заголовок вида

```
<тип> <идентификатор> (<список формальных парамет-
ров>);
```

и тело – список операторов, заключенный в операторы блока

```
{ … };
```

Например:

```
int abc (float r)
{
float r1,r2;
y:=sinr(r1)/cos(r2)*tan(r);
}
```

В теле функции может присутствовать оператор

```
return (<значение>);
```

Оператор безусловного перехода и метки

```
goto <метка> ;
```

Метка - идентификатор, расположенный в теле программы в начале строки, после которого стоит знак «:»:

```
a: str='ujhti';
```

Оператор условного перехода

Начинается с ключевого слова «if», имеет полный и неполный формат:

```
if (логическое выражение) оператор_1; else опера-
тор_2;
if (логическое выражение) оператор_1;
```

Вместо отдельных операторов могут использоваться блоки операторов:

```
if (логическое выражение)
{операторы_1}
else
```

{*операторы_2*}

6.5. Язык программирования Perl

(5.003 for FreeBSD 2.1.0.)

Идентификаторы

Произвольная последовательность букв и цифр, начинающаяся со специального символа или буквы. Может включать символы подчеркивания и начинаться с них.

Специальный начальный символ определяет тип идентификатора:

- $ - идентификатор обозначает обычную переменную,
- @ - идентификатор является именем массива (структуры).

Отсутствие символа означает, что идентификатор является именем процедуры.

Числовые константы целого типа

Произвольная последовательность цифр без знака.

Числовые константы вещественного типа, представленные с фиксированной точкой

Последовательность цифр, включающая одну десятичную точку вида

123.45

.25

25.

Числовые константы вещественного типа, представленные с плавающей точкой

Последовательность, включающая цифры, десятичную точку (необязательную), символ «e» или «E», а также знак «+» или «-» вида (необязательный):

```
1.23e-25
1.23E-25
1.23e+25
1.23E+25
.25e-5
```

```
.25E+5
1.23e2
1.23E2
```

Символьные (строковые) константы

Строковая константа - последовательность символов, заключенная в апострофы или кавычки, расположенная в пределах одной строки, вида:

```
"acb 12_& ?tu"
'abc'
```

Переменные с индексами (массивы и элементы массивов)

Идентификатор, после которого в квадратных скобках перечислены выражения-индексы, вида:

```
@Abc[12]
@C[1+i]
```

Комментарии (строчные и блочные)

Блочные комментарии отсутствуют.

Строчные – от символа «#» до конца строки.

```
$i=$i+1; # это инкремент
```

Обращения к подпрограммам

Идентификатор, следующий после знака «&», после которого в круглых скобках следует последовательность выражений-аргументов, разделенных запятыми. При отсутствии аргументов скобки не ставятся:

```
&F(12, 4, $i);
&f($av-6);
&g;
```

Арифметические операции

Сложение	+
Вычитание	-
Умножение	*
Деление	/
Возведение в степень	**

Операции сравнения

Меньше	<
Больше	>
Равно	==
Не равно	!=
Меньше или равно	<=
Больше или равно	>=

Оператор присваивания

Имеет вид «=». Слева стоит идентификатор или элемент массива, а справа – выражение. Заканчивается символом «;», например:

```
$a=$b+$c;
@b[$i-9]=12;
```

Операторы блока

```
{ – начало блока
…
} – конец блока
```

Структура программы

Программа представляет собой произвольную последовательность операторов и подпрограмм.

Операторы описания данных (идентификаторов и массивов)

Операторы описания данных в языке отсутствуют.

Операторы описания подпрограмм

Подпрограммы имеют заголовок вида

```
sub <идентификатор>
```

и тело – список операторов, заключенный в операторы блока

```
{ … }
```

Например:

```
sub show_value
    {
      print 'The value    id ', $_[0];
```

```
        }
    &show_value(1001);
```

В теле подпрограммы может присутствовать оператор

```
return <значение>;
```

Оператор безусловного перехода и метки

```
goto <метка> ;
```

Метка - идентификатор, расположенный в теле программы в начале строки, после которого стоит знак «:»:

```
a: $str='ujhti';
```

Оператор условного перехода

Начинается с ключевого слова «if», имеет полный и неполный формат:

```
if (<логическое выражение>)
        {
            <оператор_1>;
        }
else
        {
            <оператор_2>;
        }
```

или

```
if (<логическое выражение>)
        {
            <оператор>;
        }
```

6.6. Язык программирования Fortran

(Fortran IV):

Идентификаторы

Произвольная последовательность прописных букв и цифр, начинающаяся с буквы.

Числовые константы целого типа

Произвольная последовательность цифр без знака.

Числовые константы вещественного типа, представленные с фиксированной точкой

Последовательность цифр, включающая одну десятичную точку вида

123.45

.25

25.

Числовые константы вещественного типа, представленные с плавающей точкой

Последовательность, включающая цифры, десятичную точку (необязательную), символ «E», а также знак «+» или «-» вида (необязательный):

1.23E-25

1.23E+25

1.23E2

.25E-6

Символьные (строковые) константы

Символьная константа – один символ, заключенный в апострофы:

```
'a'
```

Строковая константа - последовательность символов, заключенная в кавычки, расположенная в пределах одной строки, вида:

```
"acb 12_& ?tu"
```

Переменные с индексами (массивы и элементы массивов)

Идентификатор, после которого в круглых скобках через запятую перечислены выражения-индексы, вида:

```
Abc(12,I,I-6)
C(1+i)
F()
```

Структура программы

Структура программы является строково-ориентированной. Так, первый символ строки служит для маркировки текста как комментария (символом С), с первого по пятый символ располагается область меток, а с седьмого по семьдесят второй располагается собственно текст оператора или комментария. Остальные символы строки транслятором игнорируются. Если текст оператора не вписывается в отведенное пространство (с 7 по 72 символ), в шестой колонке следующей строки ставится признак продолжения, и затем оператор продолжается на ней.

Располагать два или более оператора в одной строке нельзя.

Программа имеет заголовок вида

```
PROGRAM <ИМЯ ПРОГРАММЫ>
```

Затем следуют операторы описания данных, за ними – исполняемые операторы основной программы, оканчивающиеся операторами

```
STOP
END
```

Комментарии (строчные и блочные)

Блочные комментарии отсутствуют.

Строчные – строка, начинающаяся с символа «С» в первой позиции.

```
C это инкремент
```

Обращения к функциям пользователя

Идентификатор, после которого в круглых скобках следует последовательность выражений-аргументов, разделенных запятыми. Скобки могут быть пустыми в случае отсутствия аргументов:

```
F(12, 4, I)
A3(AV-6)
G()
```

Вызов подпрограмм пользователя

Осуществляется оператором CALL, в котором указывается имя подпрограммы, после которого в круглых скобках следует последовательность выражений-аргументов, разделенных запятыми. Скобки могут быть пустыми в случае отсутствия аргументов:

```
CALL F(12, 4, I)
CALL A3(AV-6)
CALL G()
```

Арифметические операции

Сложение	+
Вычитание	-
Умножение	*
Деление	/
Возведение в степень	**

Операции сравнения

Больше	.GT.
Меньше	.LT.
Больше равно	.GE.
Меньше равно	.LE.
Не равно	.NE.
Равно	.EQ.

Оператор присваивания

Имеет вид «=». Слева стоит идентификатор или элемент массива, а справа – выражение, например:

```
A=B+C
B(2,I-9)=12
```

После основной программы располагаются одна или несколько подпрограмм.

Операторы описания данных (идентификаторов и массивов)

Описание переменных с ключевого слова типа и содержит перечисление идентификаторов через запятую

```
<тип> <список элементов>
```

Типы переменных:

INTEGER (целый),

REAL (вещественный),

CHARACTER (символьный).

Например,

```
REAL A,B
```

Для описания массивов используется оператор DIMENSION, в котором указывается его имя и список размерностей в круглых скобках через запятую:

```
DIMENSION <имя массива> (размерность)
```

Например:

```
DIMENSION MASSIVE(A1,…,An)
```

Операторы описания функций

Функции имеют заголовок вида

```
<тип> FUNCTION <идентификатор>
PARAMETER <список формальных параметров>
```

и тело – список операторов, начинающийся операторами описания данных и оканчивающийся операторами

```
RETURN
END
```

Операторы описания подпрограмм

Подпрограммы имеют заголовок вида

```
SUBROUTINE <идентификатор>
PARAMETER <список формальных параметров>
```

и тело – список операторов, начинающийся операторами описания данных и оканчивающийся операторами

```
RETURN
END
```

Оператор безусловного перехода и метки

```
GO TO <метка>
```

Метка - число, расположенное в области с 1 по 5 символ строки:

```
123   I=25
```

…

```
      GOTO 123
```

Оператор условного перехода

Начинается с ключевого слова «IF», имеет только неполный формат:

```
IF (логическое выражение) оператор
```