Оператор цикла с предусловием while..do

Оператор, реализующий этот алгоритм, в языке Free Pascal имеет вид:

while выражение do оператор ;

Выражение должно быть логической константой, переменной или логическим выражением.

Работает оператор while следующим образом. Вычисляется значение выражения. Если оно истинно (True), выполняется оператор. Выполнение цикла заканчивается, если условие ложно, после этого управление передается оператору, следующему за телом цикла. Выражение вычисляется перед каждой итерацией цикла.

Если при первой проверке выражение ложно (False), цикл не выполнится ни разу.

Если в цикле надо выполнить более одного оператора, необходимо использовать составной оператор :

```
while условие do
begin
оператор_1 ;
оператор_2 ;
. . .
оператор_n ;
end ;
```

Рассмотрим пример. Пусть необходимо вывести на экран значения функции y = x^n

Применим цикл с предусловием :

```
var x,n,y  : real ;
i: integer;
begin
{считывание значений x и n}
writeln('введите число и показатель степени');
readln(x,n);
{Присваивание параметру цикла стартового значения}
i: = 1 ;
{присваивание переменной, необходимой для хранения результата начального значения}
y:=1;
{Цикл с предусловием}
while i<=n do {Пока параметр цикла не превышает
```

Если необходимо возвести 2 в степень 5, программа будет выполняться следующим образом:

1. начальные значения: x=2, n=5, i=1, y=1. При первой итерации цикла проверяем условие i<=n -> 1<=5, условие истинно, следовательно выполняем цикл. получаем следующие значения: y=y*x -> y=1*2=2; i=i+1 -> i=1+1=2;

2. при втором прохождении цикла проверяем условие: i<=n -> 2<=5 условие снова истинно выполняем дальше. В результате второго прохождения цикла получим следующие значения: y=y*x -> y=2*2=4; i=i+1 -> i=2+1=3;

3. снова проверка условия: 3<=5. Условие истинно, цикл продолжается получаем следующие значения: y=y*x -> y=4*2=8; i=i+1 -> i=3+1=4;

4. проверяем условие: 4<=5. Условие истинно, цикл продолжается получаем следующие значения: y=y*x -> y=8*2=16; i=i+1 -> i=4+1=5;

5. проверяем условие 5<=5. Условие истинно, продолжаем выполнять цикл, получаем результат: y=y*x -> y=16*2=32; i=i+1 -> i=5+1=6;

6. проверяем условие 6<=5. Условие ложно прекращаем цикл. Переходим к оператору, следующему после цикла.

7. выводим результат (последнее полученное значение y): x^n= 32

Оператор цикла с постусловием repeat..until

Оператор, реализующий цикл с постусловием, в языке Free Pascal имеет вид:

repeat оператор until выражение;

Если в цикле с предусловием проверка условия осуществляется до тела цикла, то в цикле с постусловием условие проверяется после тела цикла. Сначала выполняются операторы, являющиеся телом цикла, после чего проверятся условие, если последнее ложно, то цикл повторяется. Выполнение цикла прекратится, если условие станет истинным.

Работает цикл следующим образом. В начале выполняется оператор, представляющий собой тело цикла. Затем вычисляется значение выражения. Если оно ложно (False), оператор тела цикла выполняется ещё раз. В противном случае цикл завершается, и управление передается оператору, следующему за циклом. Таким образом, нетрудно заметить, что цикл с постусловием всегда будет выполнен хотя бы один раз, в отличие от цикла с предусловием, который может не выполниться ни разу.

Пример:
Посчитать количество цифр заданного натурального числа n.

```
program nat;
uses crt;
var n:real;
k:integer; //счетчик цифр
begin
writeln('введите натуральное число ');
readln(n);
k:=0;
repeat
n:=n div 10;
k:=k+1;
until n=0;
writeln('Количество цифр числа = ',k);
end.
```

В результате мы будем получать следующие значения:

Шаг цикла	n	m	k
Исходные значения	12345	12345	0
1	12345	1234	1
2	12345	123	2
3	12345	12	3
4	12345	1	4
5	12345	0	5

Оператор цикла for..do

Оператор цикла for..do используется в тех случаях, когда известно количество итераций цикла:

for параметр_цикла:=начальное_значение to конечное_значение do оператор ;

for параметр_цикла:=конечное_значение downto начальное_значение do оператор ;

где оператор — любой оператор языка, параметр_цикла — имя переменной целочисленного типа, начальное_значение и конечное_значение должны быть того же типа, что и параметр_цикла.

Шаг изменения цикла for всегда постоянен и равен интервалу между двумя ближайшими значениями типа параметра цикла (при целочисленном значении параметра цикла шаг равен 1).

В случае если тело цикла состоит более чем из одного оператора, необходимо использовать составной оператор:

```
for параметр_цикла:=начальное_значение to
конечное_значение do
begin
оператор_1 ;
оператор_2 ;
. . .
оператор_N ;
end ;
```

алгоритм работы цикла for..do.

1) Параметру_цикла присваивается начальное_значение.

2) Если значение параметра_цикла превосходит конечное_значение, то цикл завершает свою работу. В противном случае выполняется п. 3.

3) Выполняется оператор.

4) Значение параметра_цикла изменяется на соответствующий шаг и осуществляется переход к п. 2, и т. д. Понятно, что этот алгоритм представляет собой цикл с предусловием.

Фрагмент подпрограммы, приведённый далее, демонстрирует применение цикла for:

```
var i : integer ; c : char ;
begin
{Вывод на экран чисел от 1 до 10.}
for i :=1 to 10 do
writeln ( i ) ;
{Вывод на экран чисел от 10 до -10.}
for i :=10 downto −10 do
writeln ( i ) ;
{Вывод на экран символов от a до r.}
for c := ’ a ’ to ’ r ’ do
writeln ( c ) ;
end ;
```

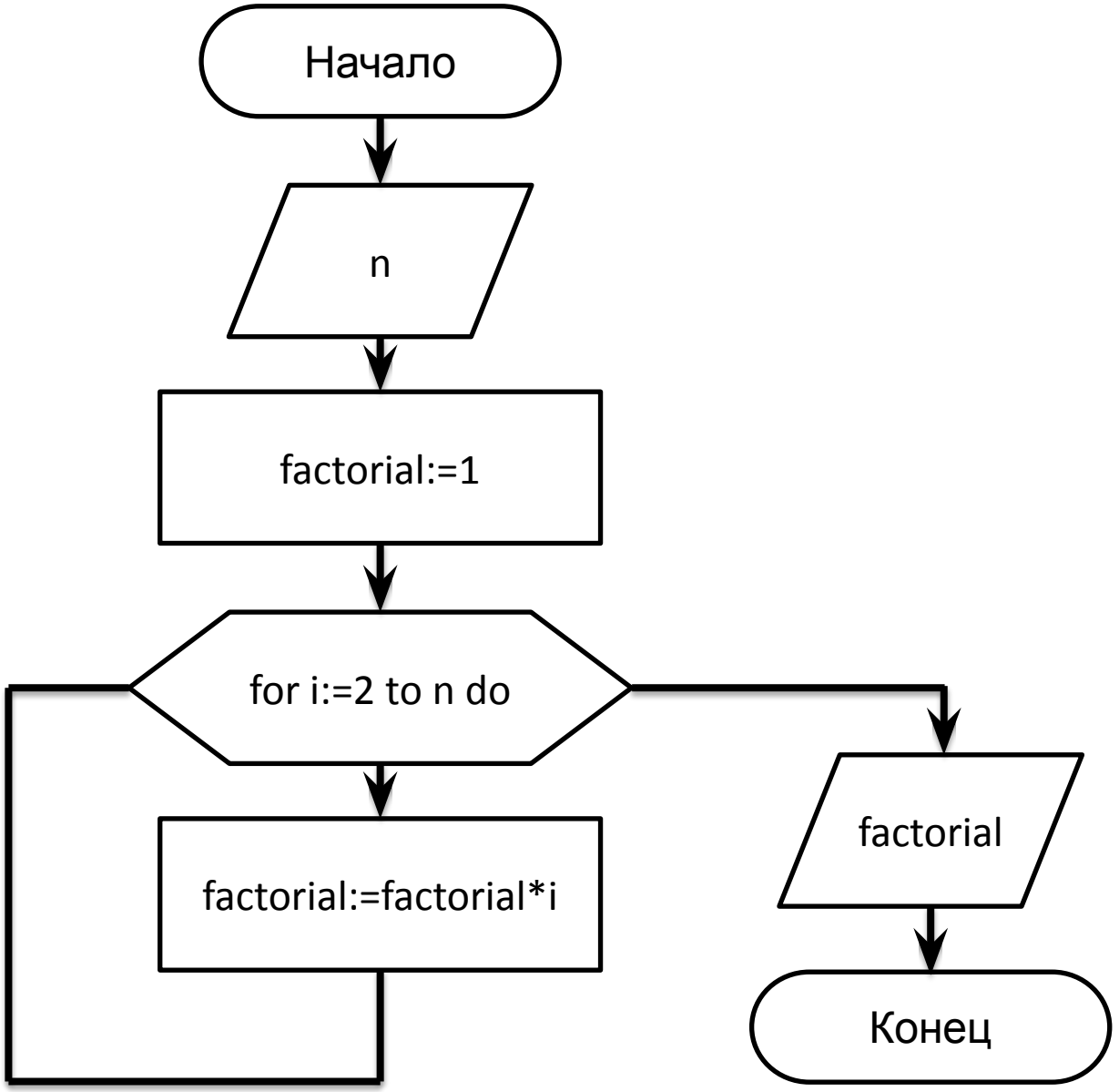
пример: вычислить факториал числа.

Входные данные: N — целое число, факториал которого необходимо вычислить.

Выходные данные: factorial — значение факториала числа N, произведение чисел от 1 до N, целое число.

Промежуточные переменные: i — параметр цикла, целочисленная переменная, последовательно принимающая значения 2, 3, 4 и так далее до N.

Блок-схема


Начало
n
factorial:=1
for i:=2 to n do
factorial:=factorial*i
factorial
Конец

Итак, вводится число N. Переменной factorial, предназначенной для хранения значения произведения последовательности чисел, присваивается начальное значение, равное единице. Затем организуется цикл, параметром которого выступает переменная i. Если значение параметра цикла меньше или равно N, то выполняется оператор тела цикла, в котором из участка памяти с именем factorial считывается предыдущее значение произведения, умножается на текущее значение параметра цикла, а результат снова помещается в участок памяти с именем factorial. Когда параметр i становится больше N, цикл заканчивается, и выводится значение переменой factorial, которая была вычислена в теле цикла.

текст программы вычисления факториала на языке Free Pascal

```
var
factorial, n, i : integer ;
begin
write ( ’ n= ’ ) ; readln ( n ) ;
factorial : = 1 ;
for i :=2 to n do
factorial := factorial * i ;
writeln (factorial) ;
end .
```

при вычислении факториала числа 5 получим:

Шаг цикла	n	i	factorial
Начальные значения	5	2	1
1	5	2	2
2	5	3	6
3	5	4	24
4	5	5	120