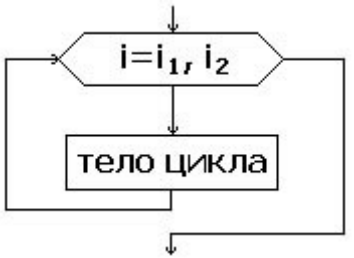
Программирование на языке Паскаль лекция 3

Циклические алгоритмы

1. Цикл с известным количеством повторений.

На языке Паскаль повторение некоторой последовательности действий известное число раз выполняет оператор `for`. Подсчёт количества выполняемых действий осуществляется при помощи специальной переменной — счётчика. Цикл `for` может быть представлен в двух формах.



Первая форма последовательно наращивает счётчик:

```
for <переменная порядкового типа>:=<начальное значение>
      to <конечное значение> do <операторы>
```

Вторая форма последовательно уменьшает счётчик:

```
for <переменная порядкового типа>:=<начальное значение>
      downto <конечное значение> do <операторы>
```

Пример программы — вывод на экран квадратов чисел от 1 до 10:

```
Program Test1;
var N: Integer;
begin
   for N:=1 to 10 do
      writeln (sqr(N));
   readln;
end.
```

Пример программы — вывод на экран кубов чисел от 11 до 5:

```
Program Test2;
var N: Integer;
begin
   for N:=11 downto 5 do
      writeln (N*N*N:5);
   readln;
end.
```

Задание №7.

1. Набрать оба примера программы, сохранить в личной папке под именем `seventh_v1.pas` и `seventh_v2.pas`.

2. Написать программу, определяющую сумму чисел от N1 до N2, введённых с клавиатуры (N2> N1) и сохранить в личной папке под именем `seventh_v3.pas`.

3. Написать программу, определяющую среднее арифметическое чисел от N1 до N2, введённых с клавиатуры (N2> N1) и сохранить в личной папке под именем `seventh_v4.pas`.

4. Задача возведения в квадрат без операции умножения.
Квадрат любого натурального числа N равен сумме N первых нечётных чисел:
$1^2=1$
$2^2=1+3$
$3^2=1+3+5$
$4^3=1+3+5+7$
$5^2=1+3+5+7+9$
. . . . . .
Основываясь на данном свойстве, составить программу, позволяющую напечатать квадраты натуральных чисел от 1 до N (N введено с клавиатуры) и сохранить в личной папке под именем `seventh_v5.pas`.

Циклы с условием

Цикл со счётчиком `for` отлично выполняет свои функции, когда число повторений тела цикла известно к моменту его начала. Но часто приходится решать задачи, когда число повторений цикла неизвестно и определяется лишь постепенно, после некоторого количества повторений тела цикла. В этом случае применяют другую разновидность цикла — цикл с условием.

В языке Паскаль циклов с условием предусмотрено два: условие цикла может проверяться перед телом цикла или после него.

2. Цикл с предусловием.

Условие проверяется перед выполнением цикла. Цикл будет повторяться до тех пор, пока проверка этого условия будет давать результат «истина» (true), то есть пока условие выполняется. Если условие сразу оказывается ложным, цикл не будет выполнен ни разу.

while <логическое условие> **do** <оператор-тело цикла>

Так же как при использовании цикла `for` и оператора `if`, после служебного слова `do` предполагается только один оператор. Если в теле цикла нужно выполнить несколько операторов, оно оформляется как блок `begin ... end`.

Пример программы — возведение числа в указанную целую степень:

```
program Step;
var P, A: Real;   // P – результат очередного шага
                  // A – основание степени
    N, i: integer;   // N – показатель степени
                     // i – счётчик числа шагов
begin
   writeln ('Введите основание степени:');
   readln(A);
   writeln ('Введите показатель степени:');
   readln(N);
i:=0;             // 0-й шаг
P:=1;             // 2^0=1
while i<abs(N) do    // показатель может быть отрицательным,
      // поэтому используем его абсолютную величину. Если N=0,
      // то в тело цикла не попадаем ни разу, т.к. 0-й шаг уже сделан
begin
   i:=i+1;  // i теперь равно номеру текущего шага
   P:=P*A;  // получаем результат i-го шага, т.е. A^i
end;
// в переменной P получен результат для положительного N
if N<0 then P:=1/P;
writeln('Результат=', P:6:3);
readln;
end.
```

Задание №8.

1. Набрать пример программы, сохранить в личной папке под именем `eighth_v1.pas`.

2. Написать программу, вычисления n! (факториал числа N), используя цикл с предусловием, сохранить в личной папке под именем `eighth_v2.pas`.
Подсказка: Факториал числа n - это произведение чисел от 1 до n, т.е. n!=1*2*3*...*n. По определению факториал нуля равен 1.

3. Написать программу, считающую произведение двух чисел A и B, введённых с клавиатуры, используя только операцию сложения. Сохранить в личной папке под именем `eighth_v3.pas`.

3. Цикл с постусловием.

Вторая разновидность цикла проверяет условие после выполнения тела цикла. Поэтому правильно будет назвать это условие условием окончания цикла. Цикл такого вида называется циклом с постусловием.

Цикл будет повторяться до тех пор, пока проверка этого условия будет давать результат «ложь» (false), то есть пока условие не выполнено. Даже если условие сразу окажется истинным, цикл выполниться хотя бы один раз.

```
repeat
        <тело_цикла>    {операторы begin ... end не требуются!}
until <логическое условие>
```

Если в цикле `while` проверялось условие продолжения цикла, то в цикла `repeat ... until` — условие окончания.

Использование оператора `repeat ... until` оправдано тогда, когда нужны повторяющиеся действия, от выполнения которых зависит дальнейшее продолжение цикла.

Пример программы, в которой требуется вводить с клавиатуры числа и подсчитывать их сумму. Сумму необходимо подсчитывать до первого введённого отрицательного числа:

```
program Summer1;
var sum, a: Real; // sum – для накопления суммы
                  // a – для очередного числа
begin
   sum:=0;
   a:=0;
   repeat
      sum:=sum+a;    // добавляем введённое число к сумме
      writeln ('Введите число:'); // ввод очередного числа
      readln(a);
   until a<0;  // проверка введённого числа на отрицательность
writeln('Сумма чисел =', sum:6:3);
readln;
end.
```

Хитрость этого примера состоит в том, чтобы выполнить оператор {`sum:=sum+a`} до проверки условия окончания цикла и не нарушить правильности алгоритма. Ведь если поставить {`sum:=sum+a`} после ввода переменной `a`, то введённое отрицательное число добавится к сумме, чего быть не должно. А если поставить его перед вводом переменной `a`, то что же тогда прибавится к сумме во время первого шага цикла? Ведь переменная `a` ещё не введена! Изначально присвоив переменной `a` нулевое значение, мы решаем эту проблему.

Пример программы, решающей туже задачу, но с использованием оператора цикла `while`:

```
program Summer2;
var sum, a: Real; // sum – для накопления суммы
                  // a – для очередного числа
begin
   sum:=0;
   writeln ('Введите число:'); // ввод очередного числа
   readln(a);
   while a>=0 do
      begin
         sum:=sum+a;   // добавляем введённое число к сумме
         writeln ('Введите число:'); // ввод очередного числа
         readln(a);
      end;
writeln('Сумма чисел =', sum:6:3);
readln;
end.
```

Необходимость задать начальное значение переменной a вынуждает нас повторить операторы ввода переменной `a` дважды — до цикла и внутри него. С этой точки зрения использование `while` оказывается менее удобным.

Задание №8, продолжение.

1. Написать программу, считающую произведение двух чисел A и B, введённых с клавиатуры, используя только операцию сложения. Использовать цикл с постусловием. Сохранить в личной папке под именем `eighth_v4.pas`.
2. Написать программу, которая подсчитывает произведение целых чисел, введённых с клавиатуры. Произведение подсчитывается до тех пор, пока вводятся числа в интервале от -10 до +10. Использовать цикл с постусловием. Сохранить в личной папке под именем `eighth_v5.pas`.
3. Выполнить ту же задачу, но с использованием цикла с предусловием. Сохранить в личной папке под именем `eighth_v6.pas`.
4. Написать программу «Угадай-ка»:

С использованием датчика случайных чисел в программе загадывается число в диапазоне 0...100. На отгадывание числа даётся 10 попыток. Играющий вводит каждый раз очередное число. После каждого ответа программа выводит на экран одно из сообщений - «больше», «меньше» или «угадано», в зависимости от числа, введённого пользователем. Цикл завершается при выполнении одного из условий: либо число попыток достигло 10, либо дан правильный ответ.

Подсказки:

1. Каждый раз в цикле наращивается переменная `k`, содержащая счётчик попыток.

2. Для отслеживания правильного ответа введите логическую переменную `flag`, которой первоначально следует присвоить значение `False`. Если ответ верен, присвойте этой логической переменной значение `True`.

3. Цикл завершается, если значение счётчика попыток `k` равно 10 или если логическая переменная имеет значение `True` (использовать операцию логическое «или» - `OR`).

4. При выходе из цикла надо сообщить, угадано ли число или же выход и цикла произошёл по совершении 10 попыток. Для этого надо проверить значение логической переменной `flag` «истина» или «ложь»).

Сохранить в личной папке под именем `eighth_v5.pas`.