Теория для Занятия №18-19.

Оператор цикла WHILE (ПОКА)

Применяется для организации циклов с неизвестным заранее числом повторений, логика которых предполагает выполнение цикла пока истинно некоторое условие.

Синтаксис оператора WHILE:

WHILE Условие **DO** Оператор;

Конструкция WHILE…DO переводится как “пока…делать”. Оператор (простой или составной), стоящий после служебного слова DO и называемый телом цикла, будет выполнятся циклически, пока значение “Условия” равно TRUE (истина). Само условие цикла может быть логической константой, переменной или логическим выражением.

Условие выполнения тела цикла WHILE проверяется до начала каждой итерации. Поэтому если условие сразу не выполняется, то тело цикла игнорируется и будет выполнятся оператор, стоящий сразу за телом цикла.

Оператором в теле цикла может быть другой циклический оператор, т.е. циклы могут быть вложенными.

При написании циклов с предусловием следует помнить о двух вещах. Во-первых, чтобы цикл имел шанс когда-нибудь завершится, содержимое его тела должно обязательно влиять на условие цикла. Во-вторых, условие должно состоять из корректных выражений и значений, определенных еще до первого выполнения тела цикла.

Пример цикла WHILE: Вычисление факториала 10! = 1·2·3·4·5·6·7·8·9·10

```
VAR
  Factorial, N : Integer;
BEGIN
  Factorial := 1; {стартовое значение факториала =0! }
  N := 1;        {стартовое значение для условия цикла }
  WHILE N<=10 DO              {заголовок цикла, условие }
   begin                    {начало тела цикла }
     Factorial := Factorial*N; {вычисление факториала N! }
     N := N + 1              {N должно меняться в цикле}
   end;                     {конец тела цикла }
  WriteLn('10!= ',Factorial);  {вывод результата расчета }
END.
```

Обратите внимание на присваивание N:=1 пред циклом. Без него значение N может быть любым, и условие может оказаться некорректным, не говоря уже о самом значении факториала. Значение N меняется внутри цикла. При этом гораздо безопаснее так писать тело цикла, чтобы оператор, влияющий на условие, был бы последним. Это гарантирует от нежелательных переборов. Если, скажем, в

рассмотренном выше примере, поставить строку N:=N+1; перед вычислением факториала, то результатом программы будет значение 11!. Исправить оплошность можно, заменив стартовое значение N на 0, а условие – на N<10. Но от этого программа вряд ли станет нагляднее.

Следует так же обратить внимание на то, что в некоторых случаях, когда значение логического выражения станет равно **False**, цикл никогда не завершится!!!

Пример простейшего бесконечного цикла: **While True Do** <оператор>;

Практическая часть

1. Дано натуральное число N. Определить количество цифр в данном числе, их сумму, количество четных цифр и наибольшую цифру данного числа.
2. Составить программу проверки «гипотезы Сиракуз». Она гласит: если взять любое натуральное число N. Если оно четно, то разделим его на 2, а если не четно, то умножим на 3 и прибавим 1 и будем повторять эти действия, то в конце независимо от выбранного числа мы получим 1. (с помощью программы оцените среднюю длину цепочек для 1<N<21, найдите закономерность в полученных цепочках и подумайте, как её использовать для больших чисел???)
3. Напишите программу поиска минимального числа, большего 300 и которое нацело делится на 19.
4. Напишите программу, которая определяет, является ли заданное натуральное число степенью 3.