Лабораторная работа № 3

Программирование процессов циклической структуры

Цель работы: изучение основных приемов проектирования и отладки программ циклической структуры на Паскале.

Процессы циклической структуры

В **циклических алгоритмах** выполнение некоторых операторов (групп операторов) осуществляется многократно с одними и теми же или модифицированными данными.

Циклический алгоритм должен содержать следующие основные блоки:

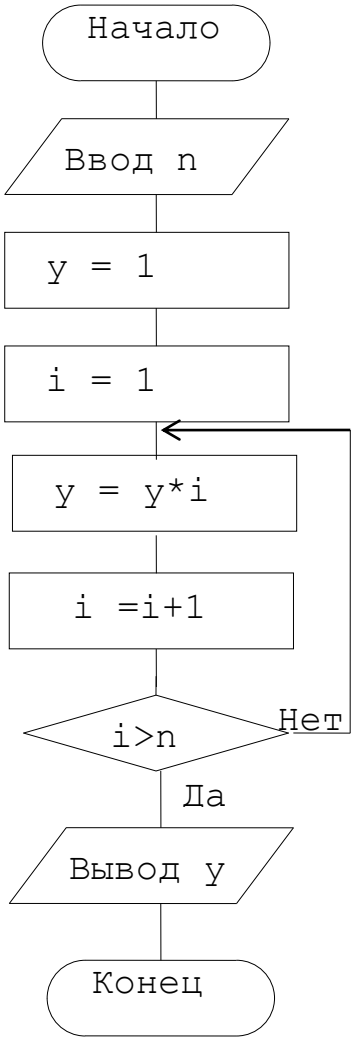
- определение начальных значений параметров (инициализация);
- проверка условия продолжения или окончания вычислений;
- вычисление операций при заданных параметрах цикла (тело цикла);
- изменение значений параметров цикла.

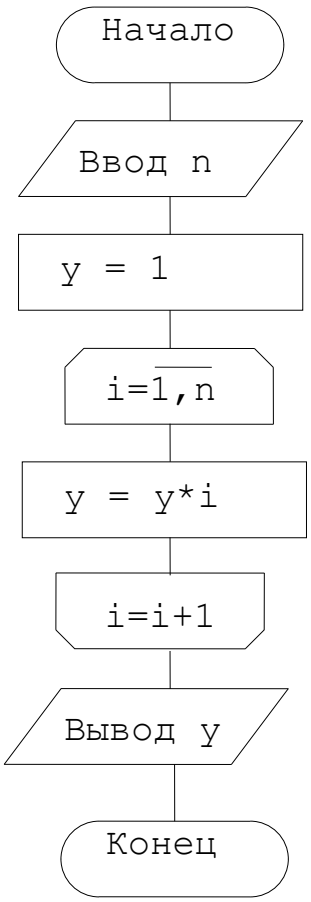
Для реализации циклических структур в схемах алгоритма используется символ "границы цикла", но в некоторых случаях удобнее использовать символ "решение".

Символ **граница цикла** состоит из двух частей и отображает начало и конец цикла. Обе части символа имеют один и тот же идентификатор. Условия для инициализации, приращения, завершения цикла помещаются внутри символа в начале или конце в зависимости от расположения операции, проверяющей условие завершения цикла.

На рисунке 3.1 приведены схемы алгоритма вычисления факториала (y=n!), использующие символ "решение" (рисунок 3.1,а) и символ "границы цикла" (рисунок 3.1,б).

Циклический алгоритм, представленный на рисунке 3.1,а, содержит все необходимые блоки: определение начальных значений параметров (i = 1); вычисление операций при заданных параметрах цикла (y = y*i); изменение значений параметров цикла (i =i+1); проверка условия продолжения или окончания вычислений (i>n).





а) б)

Рисунок 3.1

Операторы цикла языка Паскаль

В Паскале есть три оператора цикла.

Оператор for определяет цикл с управляющей переменной (параметром) и обеспечивает выполнение оператора в теле цикла заданное число раз. В операторе указывается переменная любого перечислимого типа (управляющая переменная) и определяется последовательность значений, которые должна пробежать эта переменная в ходе выполнения цикла.

Формат

for <переменная>:=<выражение1> to <выражение2> do <оператор>;

или

for <переменная>:=<выражение1> downto <выражение2> do <оператор>;

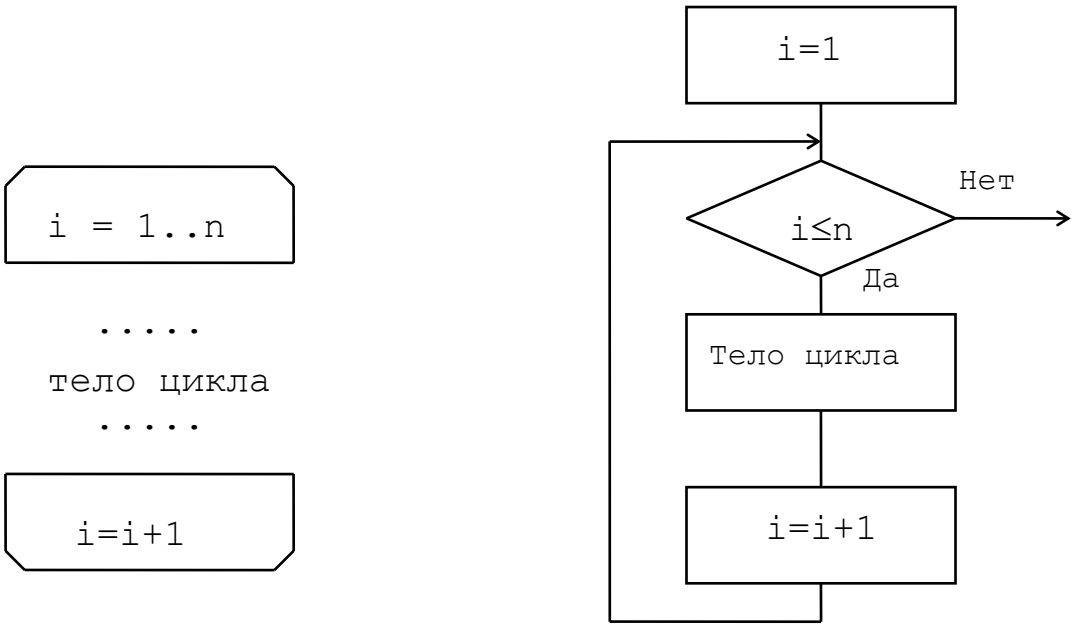
<Выражение1> и <выражение2> определяют, соответственно, начальное и конечное значение управляющей переменной и должно быть

того же типа, что и управляющая переменная. Изменение значений управляющей переменной осуществляется в порядке возрастания (to) или убывания (downto) их порядковых номеров в соответствующем типе данных.

Пример: for i:=2 to 10 do x:= x+i;

Управляющая переменная должна быть перечислимого типа. Из стандартных типов - это целый, символьный и логический типы. Шаг цикла for всегда постоянный и равен интервалу между двумя ближайшими значениями типа параметра цикла (для целого типа это 1). Изменение параметра цикла и контроль выхода из цикла осуществляется автоматически.

Варианты схем алгоритма для реализации циклических структур с помощью оператора for (при изменении параметра от 1 до n) приведены на рисунке 3.2.



а) б)

Рисунок 3.2

Рассмотрим порядок выполнения действий в операторе

for p:=kn to kv do <оператор>;

Выполнение оператора for начинается с присвоения параметру цикла начального значения (p:=kn). Затем производится проверка условия p <= kv. Если условие истинно, то выполняется оператор в цикле, а затем параметру цикла присваивается следующее значение и вновь производится проверка условия p <= kv. Если условие p <= kv не справедливо, то <оператор> в цикле не выполняется, а управление передается оператору, следующему за оператором цикла. Если используется параметр целого типа, то это означает его увеличение на единицу при каждом новом выполнении

расположенного в цикле оператора (задать шаг отличный от 1 или -1 в этом случае нельзя!).

Не рекомендуется изменять параметр цикла внутри цикла.

Войти в цикл можно только через его начало, а выйти либо при исчерпании значений параметра цикла, либо по метке, расположенной вне данного цикла.

Следует помнить, что при выходе из цикла по исчерпании значений параметра цикла значение этого параметра не определено. Хотя обычно параметр остается равным конечному значению, не следует опираться на это в своих программах. Исключение - выход из цикла переходом по goto. В этом случае значение переменной (параметра цикла) остается таким же, каким было на момент выполнения оператора goto.

Если в цикле должно быть выполнено несколько операторов, то следует использовать составной оператор.

Ниже приведена программа на языке Паскаль, реализующая вычисление факториала y=n!.

```
Program P301;
var
        n, i, y: integer;
begin
        readln(n);
        y:=1;
        for i:=1 to n do
                y:= y*i;
        writeln('y=', y);
end.
```

Оператор while определяет цикл с предварительной проверкой условия.

Варианты схем алгоритма для реализации циклических структур с помощью оператора while приведены на рисунке 3.3.

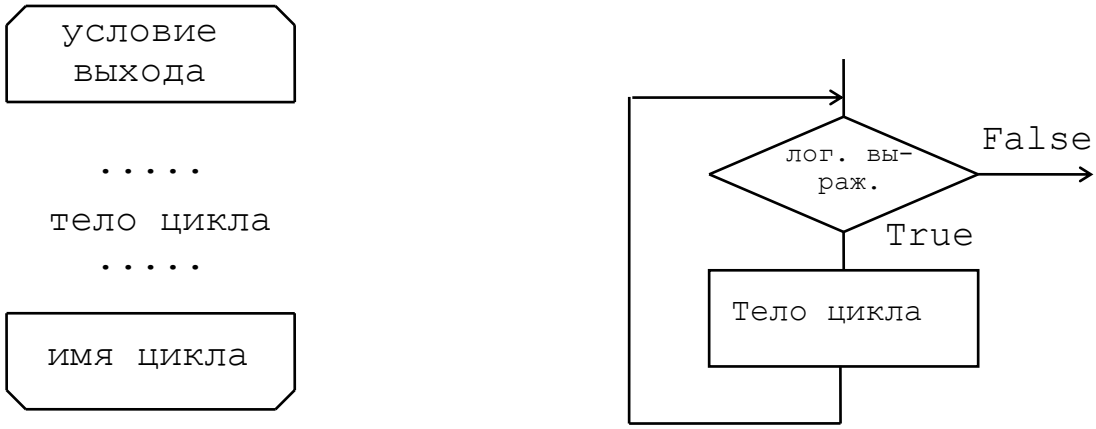


Рисунок 3.3

Формат оператора:

while <логическое выражение> do <оператор>;

Выполнение оператора while начинается с проверки логического выражения. Если оно имеет значение true, то <оператор> будет выполнен и произойдет переход к проверке логического выражения. Если логическое выражение имеет значение false, то выполнение цикла будет завершено.

Изменение операндов в логическом выражении для обеспечения выхода из цикла должно быть предусмотрено в самом цикле.

Пример. Подсчитать количество нулей, которыми заканчивается число n.

```
Program P302;
var
        i, d, n: integer;
begin
        write ('Введите натуральное число: '); readln(n);
        i:=0; d:=10;
        while n mod d=0 do
                begin
                i:=i+1;
                d:=d*10
                end;
        writeln('Число нулей: ', i);
end.
```

Если логическое выражение сразу же равно False - оператор в цикле не выполнится ни разу.

Если в цикле необходимо выполнить несколько операторов, то необходимо использовать составной оператор.

Оператор repeat определяет цикл с последующей проверкой условия.

Формат:

repeat

<список операторов>

until <логическое выражение>;

<Список операторов> может содержать больше одного оператора. Указанные операторы будут выполняться в цикле до тех пор, пока <логическое выражение> не примет значение True.

Изменение операндов в логическом выражении для обеспечения выхода из цикла должно быть предусмотрено в самом цикле.

Варианты схем алгоритма для реализации циклических структур с помощью оператора repeat приведены на рисунке 3.4.

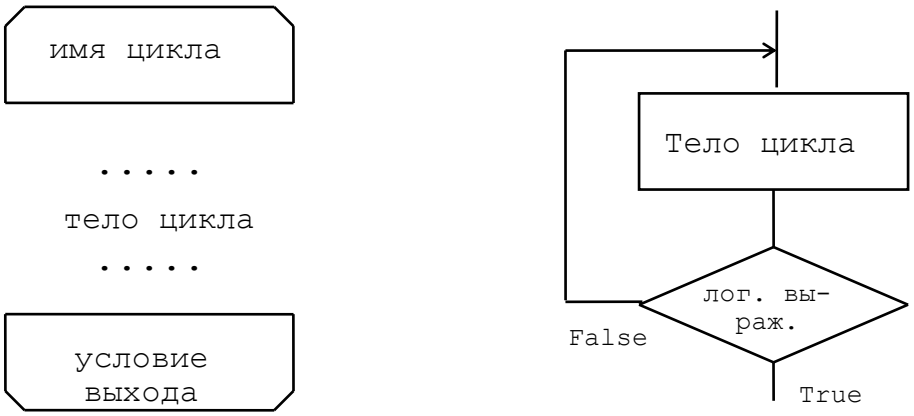


Рисунок 3.4

Пример. Вычислить сумму последовательности чисел, введенных с клавиатуры. Признак окончания ввода - равенство введенного числа нулю.

```
Program P303;
var
        s,a:integer;
begin
        s:=0;
        repeat
                readln(a);
                s:=s+a;
        until a=0;
        writeln('Сумма: ',s:6);
end.
```

Тело цикла с постусловием всегда будет выполнен хотя бы один раз.

Входить в цикл можно только через заголовок цикла!

Из циклов while и repeat можно выйти не только по выполнении условия, но и с помощью оператора goto.

В циклах repeat, while и for можно использовать две стандартные процедуры - Break и Continue. Процедура Break позволяет досрочно выйти из цикла, не дожидаясь выполнения условия выхода. Процедура Continue позволяет начать новую итерацию цикла, даже если предыдущая не завершена.

Если телом цикла является циклическая структура, то такие циклы называют вложенными или сложными. Цикл, содержащий в себе другой цикл, называют внешним. Цикл, содержащийся в теле другого цикла, называют внутренним.

Внутренний и внешний циклы могут быть любыми из трех рассмотренных видов: циклами с параметром, циклами с предусловием, циклами с постусловием. Правила организации как внешнего, так и внутреннего циклов такие же, как и для простого цикла каждого из этих видов. Однако при построении вложенных циклов необходимо соблюдать следующее дополнительное условие: все операторы внутреннего цикла должны полностью лежать в теле внешнего цикла.

Параметры циклов разных уровней изменяются не одновременно. Вначале все свои значения изменит параметр цикла наивысшего уровня вложенности при фиксированных (начальных) значениях параметров циклов с меньшим уровнем. Затем изменяется на один шаг значение параметра цикла следующего уровня и снова полностью выполняется самый внутренний цикл и т.д. до тех пор, пока параметры циклов всех уровней не примут все требуемые значения.

Нельзя в двух циклах использовать один и тот же идентификатор в качестве параметра.

Пример выполнения задания лабораторной работы.

В последовательности чисел, вводимых с клавиатуры, подсчитать количество нулей. Признак завершения ввода: два одинаковых числа подряд.

Схема алгоритма для решения поставленной задачи приведена на рисунке 3.5.

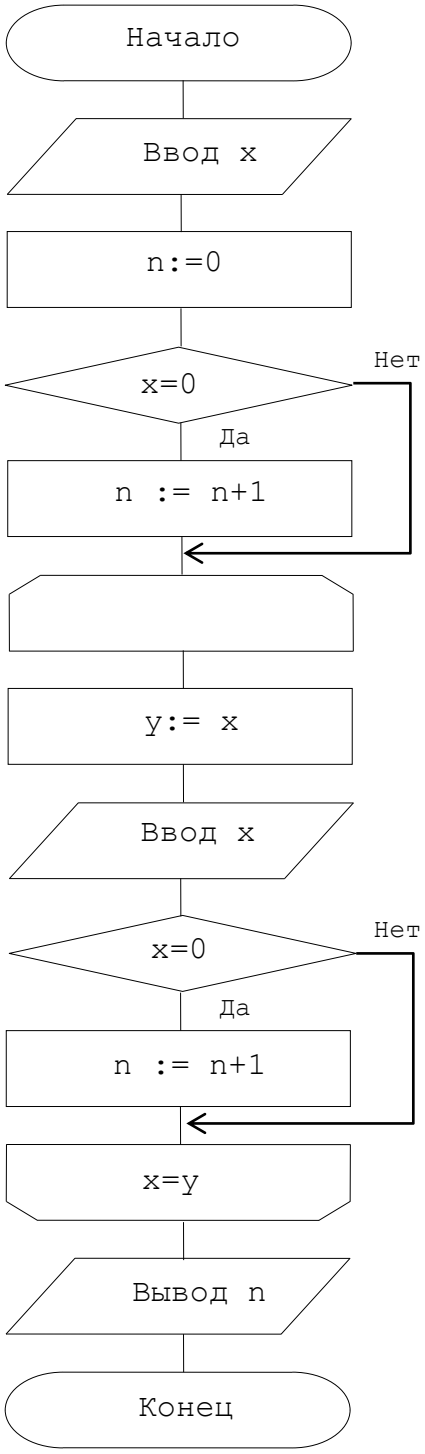


Рисунок 3.5 – Схема алгоритма

Текст программы:

```
Program ABC;
  var x, y, n : integer;
begin
  readln(x);
  n:=0;
  if x=0 then n:=n+1;
  repeat
    y:=x;
    readln(x);
    if x=0 then n:=n+1;
  until x=y;
  writeln('n=', n);
end.
```

Варианты заданий

Вариант 3.1
В последовательности чисел, вводимых с клавиатуры, подсчитать количество пар одинаковых чисел, следующих подряд. Признак завершения ввода: сумма чисел последовательности стала меньше -10.

Вариант 3.2
В последовательности чисел, вводимых с клавиатуры, подсчитать количество пар следующих подряд равных чисел. Признак завершения ввода: два нуля подряд.

Вариант 3.3
В последовательности чисел, вводимых с клавиатуры, определить количество положительных чисел. Признак завершения ввода: сумма чисел превысила 25.

Вариант 3.4
В последовательности чисел, вводимых с клавиатуры, подсчитать количество четных чисел. Признак завершения ввода: число 8 встретилось два раза.

Вариант 3.5
В последовательности чисел, вводимых с клавиатуры, подсчитать количество повторений двух одинаковых чисел. Признак завершения ввода: введение числа равного первому.

Вариант 3.6
В последовательности чисел, вводимых с клавиатуры, определить количество чисел между двумя числами 6. Признак завершения ввода: второе число 6.

Вариант 3.7
В последовательности чисел, вводимых с клавиатуры, подсчитать количество четных чисел. Признак завершения ввода: два числа подряд больше 20.

Вариант 3.8
В последовательности чисел, вводимых с клавиатуры, определить количество четных чисел. Признак завершения ввода: сумма чисел последовательности стала больше 100.

Вариант 3.9
Найти, есть ли в последовательности чисел, вводимых с клавиатуры, два одинаковых отрицательных числа, следующих подряд. Признак завершения ввода: первое число встретилось три раза.

Вариант 3.10
В последовательности чисел, вводимых с клавиатуры, подсчитать количество пар одинаковых чисел, следующих подряд. Признак завершения ввода: второй раз встретилось число 10.

Вариант 3.11
В последовательности чисел, вводимых с клавиатуры, определить количество чисел больше 10. Признак завершения ввода: произведение чисел последовательности оказалось больше их суммы.

Вариант 3.12
Определить, в какой из двух последовательностей чисел, вводимых с клавиатуры, больше положительных чисел. Признак завершения ввода: совпадение элементов в последовательностях.

Вариант 3.13
В последовательности чисел, вводимых с клавиатуры, подсчитать сумму чисел между двумя числами 8. Признак завершения ввода: второе число 8.

Вариант 3.14
Определить количество совпадений в двух последовательностях чисел, вводимых с клавиатуры. Признак завершения ввода: ноль в любой из последовательностей.

Вариант 3.15
В последовательности символов, вводимых с клавиатуры, подсчитать количество пар символов '*'. Признак завершения ввода: два символа '+' подряд.

Вариант 3.16
В последовательности символов, вводимых с клавиатуры, подсчитать количество *. Признак завершения ввода: ввод цифры.

Вариант 3.17
В последовательности символов, вводимых с клавиатуры, подсчитать количество цифр. Признак завершения ввода: две любые прописные латинские буквы подряд.

Вариант 3.18
Дано целое положительное число. Определить количество нулей в числе.

Вариант 3.19
Дано целое положительное число. Определить сумму его цифр.

Вариант 3.20
Дано целое число. Используется ли при его записи цифра 4?

Вариант 3.21
Даны два целых числа. В каком из них больше цифр 5?

Вариант 3.22
Даны два целых числа. Есть ли у них одинаковые (по значению) разряды.

Вариант 3.23
Вычислить для произвольного n

$$y = \sum_{i=1}^{n} \sum_{j=1}^{i} (i + j)^i$$

(Для вычисления сумм и степеней использовать операторы цикла)

Вариант 3.24
Вычислить для произвольного n

$$y = \sum_{i=1}^{n} \sum_{j=1}^{i} (i + j^i)$$

(Для вычисления сумм и степеней использовать операторы цикла)

Вариант 3.25
Вычислить для произвольного n

$$y = \sum_{i=1}^{n} \sum_{j=1}^{n} (i^j + 1)^i$$

(Для вычисления сумм и степеней использовать операторы цикла)

Вариант 3.26
Вычислить для произвольного n

$$y = \sum_{i=1}^{n} \sum_{j=1}^{i} (i^j + j)^i$$

(Для вычисления сумм и степеней использовать операторы цикла)

Примеры контрольных заданий

1. В последовательности чисел, вводимых с клавиатуры, подсчитать количество нулей. Признак завершения ввода: два одинаковых числа подряд.
2. Дано целое положительное число. Определить количество нулей в числе.
3. Вычислить для произвольного n

$$y = \sum_{i=1}^{n} \sum_{j=1}^{i} (i + j)^i$$

(Для вычисления сумм и степеней использовать оператор For)