

Цикл предусматривает многократное выполнение некоторых операторов, входящих в тело цикла.

В языке Pascal имеются три оператора цикла:

- **For** (цикл на заданное число повторений);
- **While** (цикл **ПОКА** — с предусловием);
- **Repeat** (цикл **ДО** — с постусловием).

Если **число повторений известно**, то лучше воспользоваться оператором цикла **с параметром**.

Цикл на заданное число повторений с *возрастающим* или *убывающим* значением параметра.

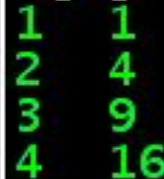
```
For {парам} := {нач_зн} To  
           {кон_зн} Do  
  {оператор} ;
```

Замечания:

Тело цикла

- параметр — порядковый тип;
- в цикле можно использовать операторные скобки;
- в теле цикла нельзя менять параметр цикла;
- параметр цикла увеличивается на единицу;
- начальное значение больше конечного, иначе тело цикла игнорируется;
- для уменьшения параметра, **to** заменяется на **downto**.

```
Program My_program;  
Uses CRT;  
Var f:Integer;  
Begin  
  ClrScr;  
  For f := 1 to 4 do begin  
    Write (f, ' ');  
    WriteLn (f*f)  
  End;  
  ReadKey;  
End.
```



```
1 1  
2 4  
3 9  
4 16
```

1. Вывод таблицы умножения в столбец.

Внешний цикл J

Начинает работу.
Выполняется 10 раз.

Внутренний цикл I

Выполняет 10 проходов
за 1 проход внешнего цикла.
Выполняется 100 раз.

```
{таблица умножения}
uses crt;
var i,j:integer;
begin
  clrscr;
  for j:=1 to 10 do
  begin
    writeln;
    for i:=1 to 10 do
      writeln(j, ' * ', i, ' = ');
    end;
  readkey;
  end.
```

```
7 * 2 = 14
7 * 3 = 21
7 * 4 = 28
7 * 5 = 35
7 * 6 = 42
7 * 7 = 49
7 * 8 = 56
7 * 9 = 63
7 * 10 = 70
8 * 1 = 8
8 * 2 = 16
8 * 3 = 24
8 * 4 = 32
8 * 5 = 40
8 * 6 = 48
8 * 7 = 56
8 * 8 = 64
8 * 9 = 72
8 * 10 = 80
```

Все операторы внутреннего цикла должны располагаться в теле внешнего.

Передача управления происходит от внутреннего цикла к внешнему!!!

1. **Выполните программу** вывода на экран в три столбца список чисел от 1 до N, их квадратов и кубов. Число N вводится с клавиатуры. Например, для N = 5 на экране должно быть:

1	1	1
2	4	8
3	9	27
4	16	64
5	25	125

Для проверки корректности работы программы при различных входных данных проводят её тестирование, которое заключается в подборе самых разнообразных входных данных, чтобы получить все возможные (и невозможные) варианты работы программы и «выловить» неучтённые ошибки.

2. **Выполните программу** вывода строчных букв латинского алфавита в прямом и обратном порядке.

Используемый материал:

Оператор цикла **For**:

```
For <парам> := <нач_зн> To <кон_зн> Do <оператор>;
```

- параметр – целый тип (обычно, **Integer**);
- в цикле можно использовать операторные скобки;
- параметр цикла увеличивается на единицу.



Цикл **While** сначала проверяет **условие**, и только если оно истинно, выполняет тело цикла.

```
While {условие} do  
  {оператор};
```

- В теле кода, написанном ниже, цикл не выполнится ни разу:

```
x:=1;  
While x>1 do  
  x:=x-1;
```

- Можно получить бесконечный цикл. Например:

```
x:=1  
While x>0 do  
  x:=x+1;
```

Программа вывода на экран суммы чисел от *a* до *b*.

Цикл работает, пока изменяющаяся переменная *f* не станет больше значения *b*.

Попробуй изменить алгоритм.

```
Program My_program;
Uses CRT;
Var a,b,s,f:Integer;
Begin
  ClrScr;

  Write ('Введите начальное число: ');
  ReadLn (a);
  Write ('Введите конечное число: ');
  ReadLn (b);

  S:=0;

  F := a;
  While f<=b do Begin
    S := S + f;
    F := F + 1;
  End;

  Write ('Сумма чисел от ', a, ' до ', b, ' равна ', S);

  ReadKey;
End.
```

Можно ли обойтись без переменной F?

```
[■] Output
Введите начальное число: 2
Введите конечное число: 10
Сумма чисел от 2 до 10 равна 54
```

1. **Выполните программу**, которая определяет максимальное из введенных чисел с клавиатуры (ввод чисел заканчивается числом 0). Ниже представлен рекомендуемый вид экрана:

```
Введите числа. Для завершения ввода
введите 0.
89
15
0
Максимальное число 89.
```

Используемый материал:

Оператор цикла **While**:

```
While <условие> do <оператор>;
```

Цикл **While** сначала проверяет условие, и только если оно истинно, выполняет основное тело цикла.

