

Программирование циклических алгоритмов на языке Паскаль

Лекция 4

План

1. Понятие цикла
2. Оператор цикла For
3. Цикл While
4. Цикл Repeat

Литература



Литература

1. Касторнов А.Ф., Евстратова Г.А. Язык программирования Паскаль : учебное пособие для вузов. - Череповец : ГОУ ВПО ЧГУ, 2010. - 117 с. - Библиогр.: С.114.
2. Электронный учебник по языку программирования Паскаль
[/http://pascal.guti.ru](http://pascal.guti.ru)



Понятие цикла

- Алгоритмы решения многих задач являются циклическими, в которых для достижения результата определенная последовательность действий выполняется несколько раз.
- Например, программа контроля знаний выводит вопрос, принимает ответ, добавляет отметку за ответ к сумме баллов, затем повторяет эти действия до тех пор, пока испытуемый не ответит на все вопросы.
- Или, например, для поиска нужной фамилии в списке следует проверить первую фамилию списка на совпадение с искомой, затем вторую, третью и т.д. до тех пор, пока не будет найдена нужная фамилия или не будет достигнут конец списка.



Понятие цикла

- Алгоритм, в котором есть группа операторов, выполняемая несколько раз, называется **циклическим**. Группа повторяемых операторов называется **телом цикла**.
- В Паскале циклы могут быть реализованы при помощи операторов циклов For, While и Repeat.

Оператор цикла For

- Оператор цикла **For** используется в том случае, если тело цикла надо выполнить несколько раз, причем число повторов заранее известно.



1-я форма записи оператора цикла For

1-я форма записи оператора **For** в общем виде выглядит следующим образом:

```
For Счетчик:=Начальное_значение to Конечное_значение do Оператор;
```

Где

- **For, to, do** – служебные слова.
 - **Счетчик** – это переменная порядкового типа (обычно типа **Integer**), которая определяет число повторов цикла.
 - Число повторов считается по формуле:
Конечное_значение–Начальное_значение+1.
 - **Конечное_значение** должно быть больше или равно **Начальному_значению**.
-



1-я форма записи оператора цикла For

- Если тело цикла состоит из нескольких операторов, то 1-я форма записи оператора **For** выглядит так:

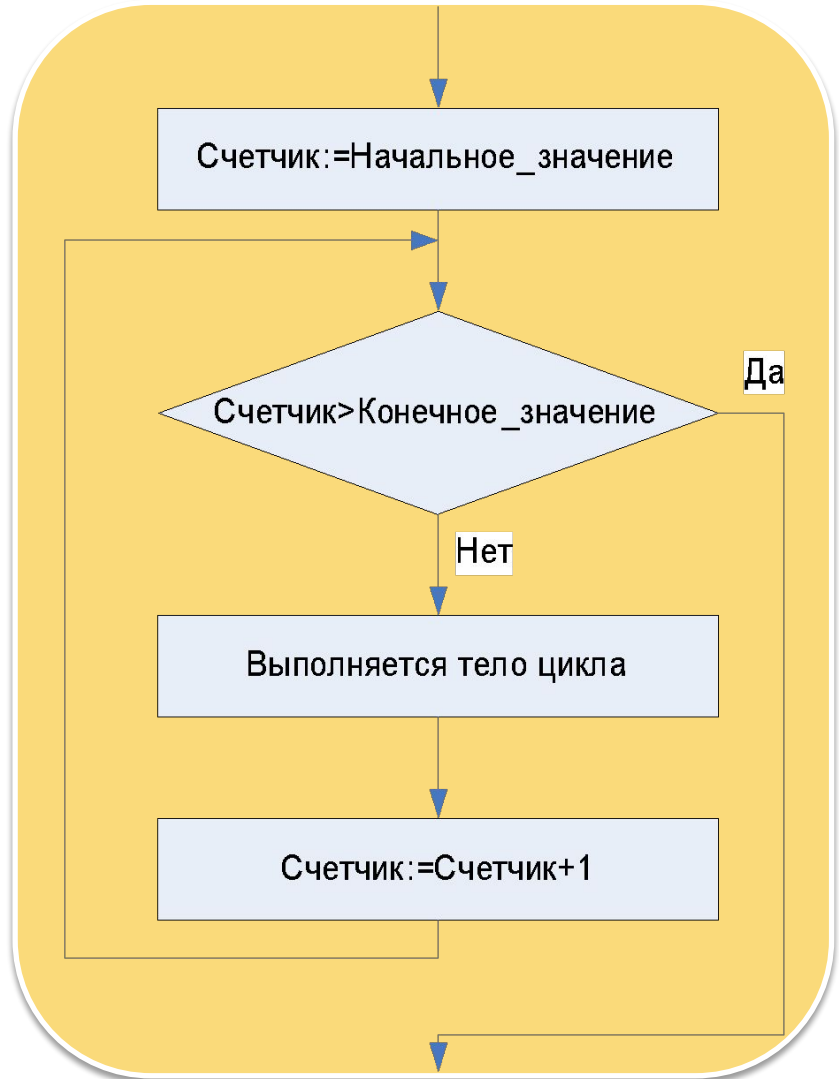
```
For Счетчик:=Начальное_значение to Конечное_значение do  
  Begin  
    {Тело цикла}  
  End;
```



1-я форма записи оператора цикла For

Рассмотрим алгоритм работы цикла **For** в первой форме записи.

1. Счетчику присваивается **Начальное_значение**.
2. Проверяется условие: Значение счетчика больше **Конечного_значения**?
3. Если условие истинно (Да), выполнение цикла заканчивается.
4. Если условие ложно (Нет), то выполняется тело цикла, затем значение счетчика увеличивается на единицу и снова выполняется проверка условия, т.е. п.2.



2-я форма записи оператора цикла For

2-я форма записи оператора **For** в общем виде выглядит следующим образом:

```
For Счетчик:=Начальное_значение downto Конечное_значение  
do Оператор;
```

Где:

- **For, downto, do** – служебные слова.
 - **Счетчик** – это переменная порядкового типа (обычно типа **Integer**), которая определяет число повторов цикла.
 - Число повторов считается по формуле:
Начальное_значение–Конечное_значение+1.
 - **Начальное_значение** должно быть больше или равно **Конечному_значению.**
-



2-я форма записи оператора цикла For

- Если тело цикла состоит из нескольких операторов, то 2-я форма записи оператора **For** выглядит так:

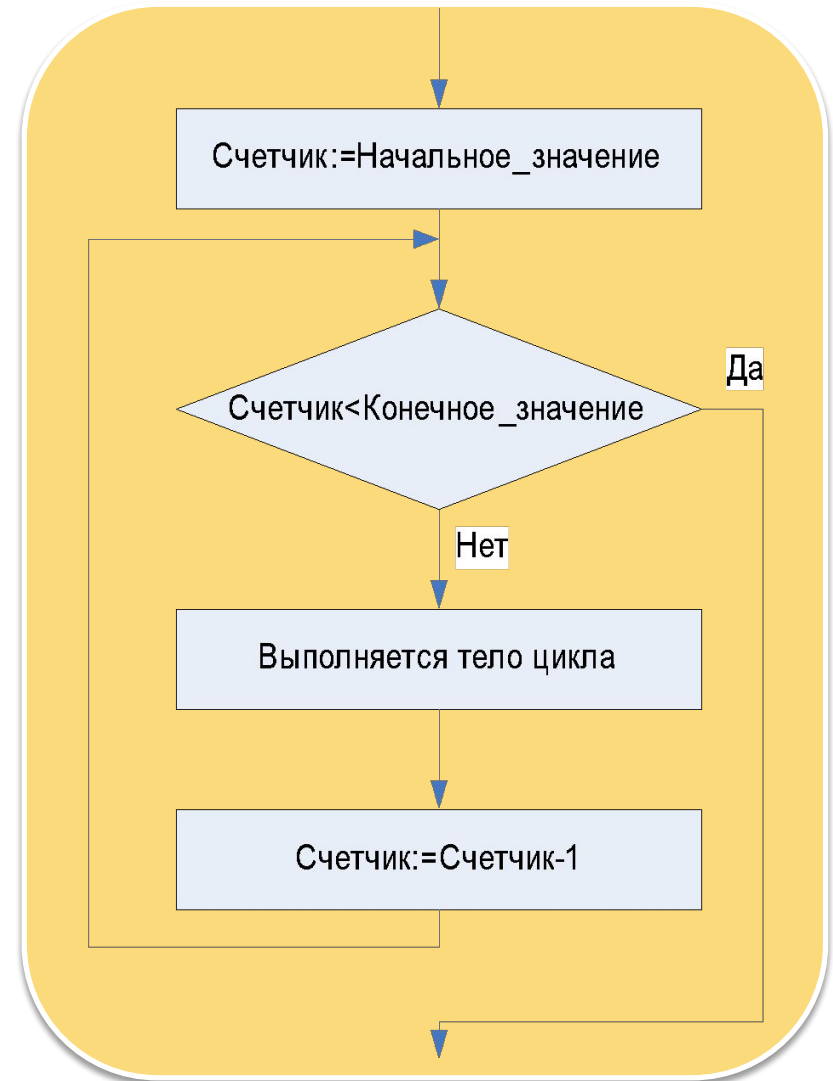
```
For Счетчик:=Начальное_значение downto  
    Конечное_значение do  
  
    Begin  
        //Тело цикла  
  
    End;
```



2-я форма записи оператора цикла For

Рассмотрим алгоритм работы цикла **For** во второй форме записи:

1. Счетчику присваивается **Начальное_значение**.
2. Проверяется условие: Значение счетчика меньше **Конечного_значения**?
3. Если условие истинно (Да), выполнение цикла заканчивается.
4. Если условие ложно (Нет), то выполняется тело цикла, затем значение счетчика уменьшается на единицу и снова выполняется проверка условия, т.е. п.2.



Оператор цикла For

Пример: Программа формирует строку звездочек. Количество звездочек в строке определяется пользователем.

program Ex1;

var i, n: integer; {i – счетчик, n – необходимое количество звездочек}

s: string; {s – формируемая строка звездочек}

begin

Writeln ('Введите количество звездочек'); {запрашивается количество звездочек}

Readln (n); {пользователь вводит количество звездочек n}

s:=''; {формирование строки звездочек начинается с пустой строки}

{Строка формируется по циклу For. Начальное_значение счетчика – 1,
Конечное_значение – необходимое количество звездочек n. }

for i := 1 **to** n **do**

s:=s+'*'; {на каждом шаге цикла к строке приклеивается одна звездочка}

Writeln (s); {выводится строка}

Readln;

end.

Цикл While

- Цикл **While** используется в том случае, если число повторений тела цикла во время разработки программы неизвестно и может быть определено только во время ее работы.
- В общем виде оператор **While** записывается следующим образом:

While Условие **do** Оператор;

Где

- **While, do** – служебные слова.
- **Условие** – выражение логического типа, определяющее продолжение цикла.



Цикл While

- Если тело цикла состоит из нескольких операторов, то цикл **While** записывается следующим образом:

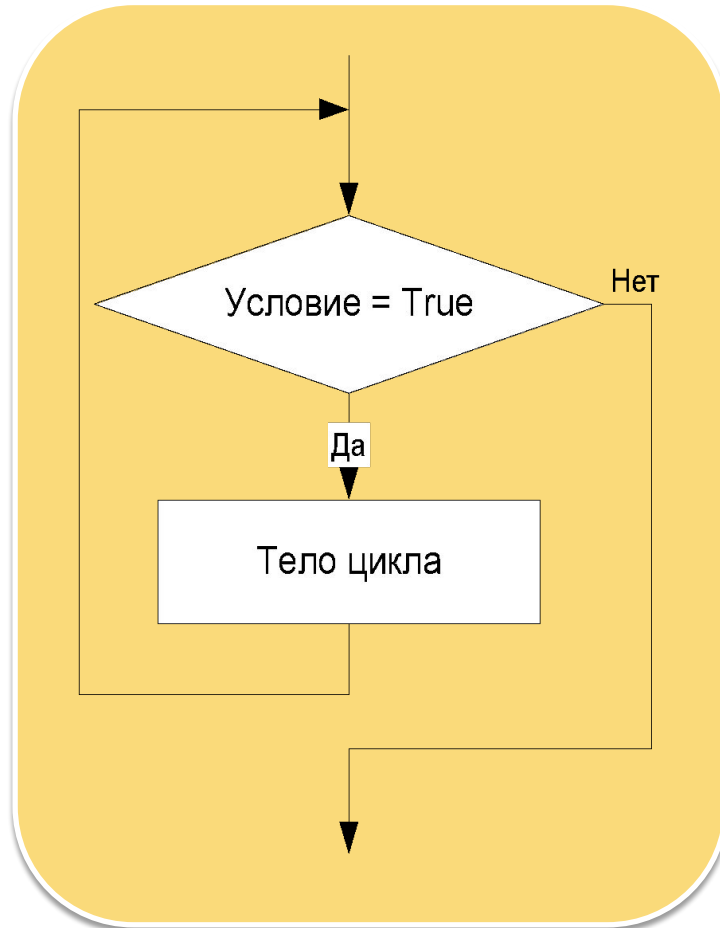
```
While Условие do  
  Begin  
    //Тело цикла  
  End;
```



Цикл While

Рассмотрим алгоритм работы цикла **While**:

1. Проверяется условие.
2. Если условие истинно, то выполняется тело цикла. После чего снова проверяется условие.
3. Если условие ложно, то цикл завершается.



Цикл While

- Таким образом, **While** – цикл с предусловием или **цикл «Пока»** (тело цикла выполняется пока истинно условие).
- Если при первом проходе цикла условие окажется ложным, то тело цикла не будет выполнено ни разу.
- Если условие никогда не станет ложным, то цикл будет повторяться бесконечно, т.е. произойдет зацикливание.



Цикл While

Program Ex2;

var Account: Real; {размер счета}

Month: Integer; {количество месяцев,
прошедших с момента открытия счета}

begin

Account:=1000; {на счет положили 1000 рублей}

Month:=0; {счет только что открыт}

while Account<=1500 **do** {пока размер счета не превышает 1 500 рублей}

begin

Account:=Account*1.02; {увеличили размер счета на 2% от предыдущего
значения}

Month:=Month+1; {увеличили количество месяцев}

end;

Writeln(Month); {вывели результат}

Readln;

end.

Пример: Гражданин открыл счёт в банке, вложив 1 000 рублей. Каждый месяц размер вклада увеличивается на 2% от имеющейся суммы. Определите, через какое количество месяцев размер вклада превысит 1 500 рублей.

Цикл Repeat

- Цикл **Repeat**, как и цикл **While**, используется в программе в том случае, если необходимо выполнить тело цикла несколько раз, но число повторений заранее неизвестно.
- В общем виде цикл **Repeat** записывается следующим образом:

```
Repeat  
    //Тело цикла  
Until Условие;
```

Где

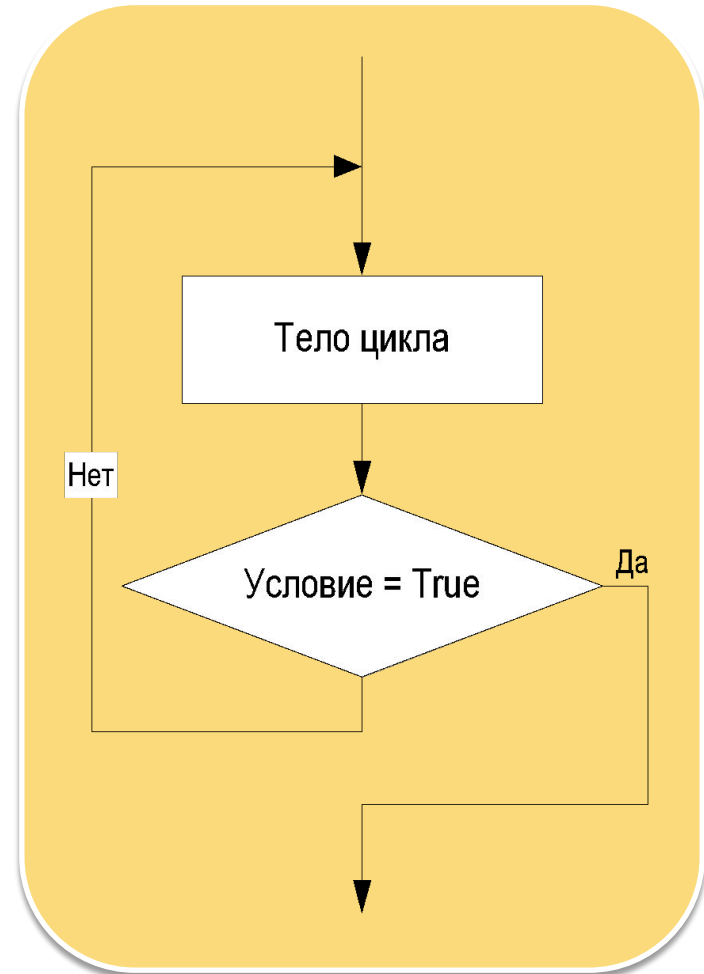
- **Repeat, Until** – служебные слова.
 - **Условие** – выражение логического типа, определяющее окончание цикла.
-



Цикл Repeat

Рассмотрим алгоритм работы цикла **Repeat**:

1. Выполняется находящееся между зарезервированными словами **Repeat** и **Until** тело цикла.
2. Проверяется условие.
3. Если условие истинно, цикл завершается.
4. Если условие ложно, снова выполняется тело цикла.



Цикл Repeat

- Таким образом, **Repeat** – цикл с постусловием или **цикл «До»** (тело цикла выполняется до истинности условия).
- Следовательно, тело цикла выполняется хотя бы один раз.
- Если условие никогда не станет истинным, то цикл станет бесконечным.



Цикл Repeat

Program Ex3;

var

Time: integer; {время деления}

Cells: integer; {количество клеток}

begin

Time:=0; {клетка еще не начала деление}

Cells:=1; {клетка одна}

Repeat

 Time:=Time+3; {через следующие три часа}

 Cells:=Cells*2; {количество клеток увеличилось в 2 раза}

Until Cells>24; {до истинности условия «количество клеток больше 24»}

Writeln (Time); {вывод результата}

Readln;

end.

Пример: Одноклеточная амёба каждые 3 часа делится на 2 клетки. Определите, через какое количество часов число клеток превысит 24.