



ЦИКЛЫ

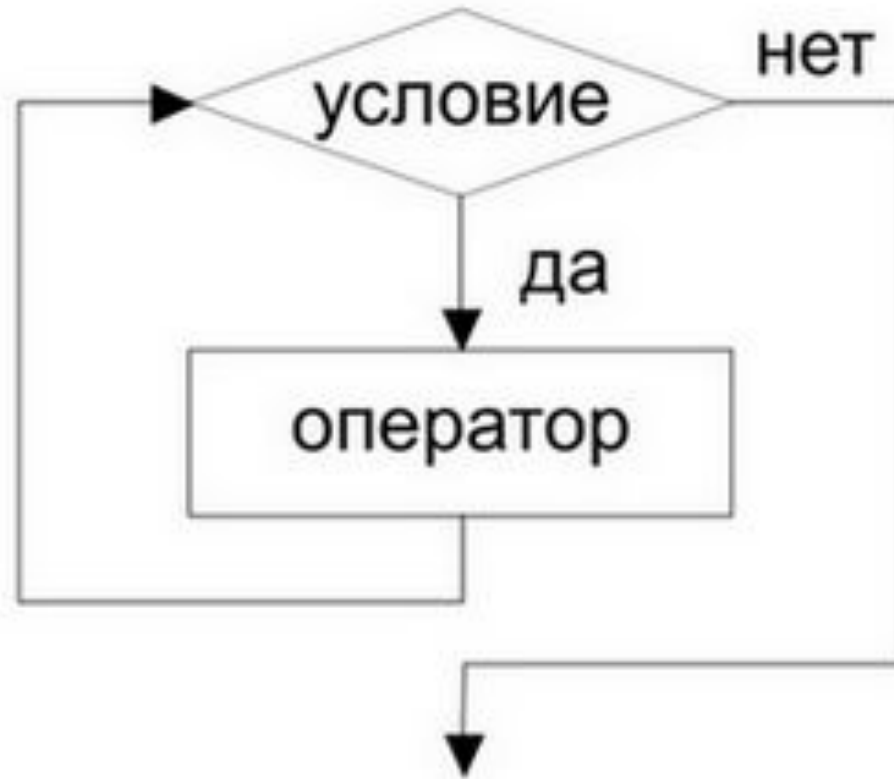
Что такое цикл

- **Цикл** — специальный оператор языка программирования, с помощью которого то или иное действие можно выполнить **нужное количество раз**, в зависимости от некоего условия.

Что такое цикл

- Каждое повторение цикла называется:
- **ШАГ ЦИКЛА** или **ИТЕРАЦИЯ**

Цикл с предварительным условием

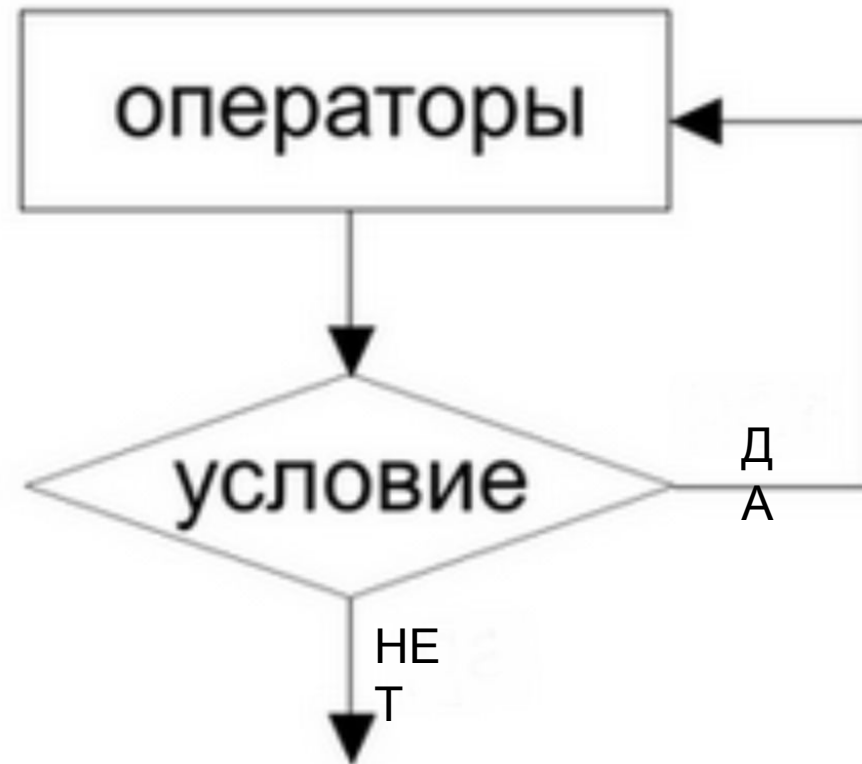


Цикл **WHILE** () цикл с предусловием

- Общий синтаксис:

```
while(утверждение)
{
    действия для повторения;
    //это тело цикла.
}
```

Цикл с последующим условием



ЦИКЛ **DO .. WHILE ()** ЦИКЛ С ПОСТУСЛОВИЕМ

do

{

 действие;

 //это тело цикла.

}

while(условие);

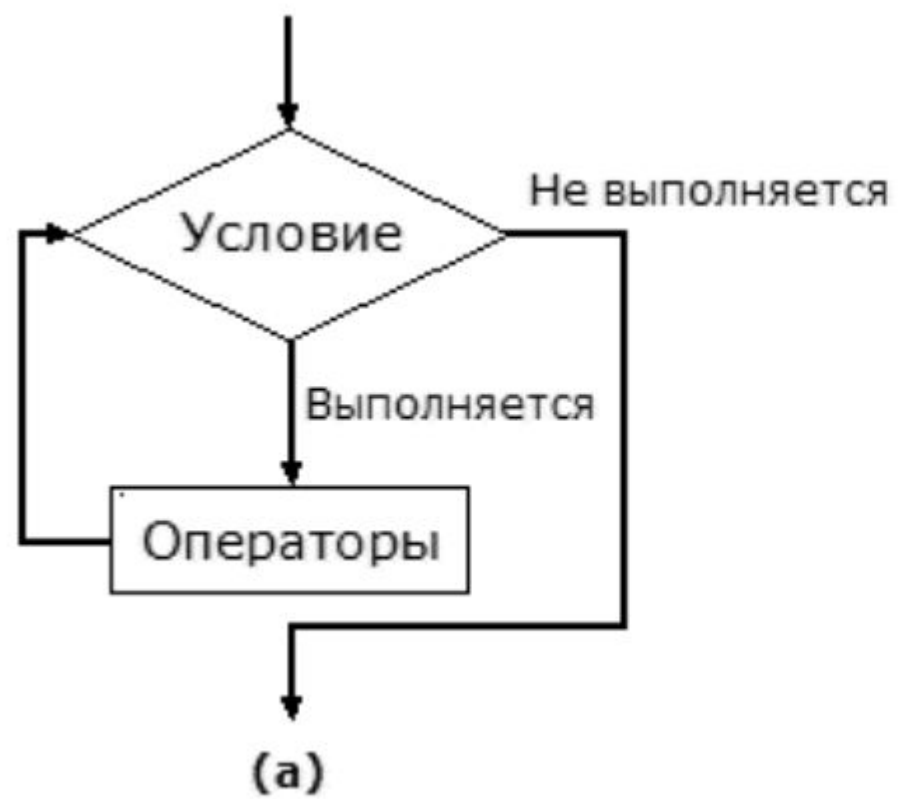
Сравнение циклов

WHILE ()

- Сначала **проверяется условие**, если оно верно выполняются действия в цикле.
- Действия могут ни разу не выполниться, если утверждение ложно.

DO .. WHILE ()

- Сначала **выполняются действия цикла**, затем проверяются условия.
- Хотя бы один раз действия выполняются при любом значении утверждения.



Цикл **WHILE ()**

- Ранее, при использовании цикла **WHILE ()**, мы, часто, создавали некоторую дополнительную переменную (**управляющую переменную**).
- Внутри цикла мы **увеличивали, либо уменьшали на 1** или на любое другое число, значение управляющей переменной.
- В условии для продолжения цикла, мы сравнивали значение управляющей переменной с неким контрольным значением и на этом основании принималось решение о прекращении или продолжении действий внутри цикла.

Цикл **WHILE** ()

```
int counter=0; // объявление управляющей переменной
while(counter<7) // проверка значения управляющей переменной
{
    counter++; // изменение управляющей переменной
    cout<<"Вы видите "<<counter<<" чудо света!!!\n";
    // действия для повторения
}
cout<<"\n";
```

Цикл `for ()`

- Цикл `for ()` организован таким образом, чтобы использование дополнительной переменной для управления циклом было более очевидным.
- В цикле `for ()` создание управляющей переменной предусмотрено синтаксисом этого цикла.
- На следующем слайде приведён пример решения предыдущей задачи при помощи цикла `for ()`.

Цикл **FOR()**

```
for (int counter=1; counter <= 7; counter ++)
```

```
// объявление управляющей переменной, проверка и изменения её значения.
```

```
{
```

```
    cout<<"Вы видите "<<counter<<" чудо света!!!\n";
```

```
    // действия для повторения
```

```
}
```

```
cout<<"\n";
```

Цикл **WHILE** ()

```
int counter; // объявление управляющей переменной
while(counter<7) // проверка значения управляющей переменной
{
    counter++; // изменение управляющей переменной
    cout<<"Вы видите "<<counter<<" чудо света!!!\n";
    // действия для повторения
}
cout<<"\n";
```

Цикл FOR

```
#include <iostream>
```

```
void main()
```

```
{
```

```
    for(int i=1 ; i<=15 ; i++ )
```

```
    {
```

```
        cout << "*" << " " << i << endl;
```

```
    }
```

```
}
```


Объявление
контрольной
переменной
i, которая
будет
управлять
циклом

Задание
условия при
котором цикл
будет
продолжен

Закон, по
которому, при
каждой
итерации, будет
изменяться
контрольная
переменная

```
{  
for(int i=1; i<=15; i++) )  
{  
    cout << "*" << " " << i << endl;  
}  
}
```

Цикл **FOR** повторяет действия заданное количество раз

for (инициализация переменной; проверка условия; изменение переменной)

{

действия;

//это тело цикла

}

Цикл **FOR** повторяет действия заданное количество раз

for (инициализация контрольной переменной; проверка условия; закон изменения контрольной переменной)

```
{
```

```
    действия;
```

```
    //это тело цикла
```

```
}
```

Контрольную переменную можно использовать в теле цикла

Варианты использования **FOR()**

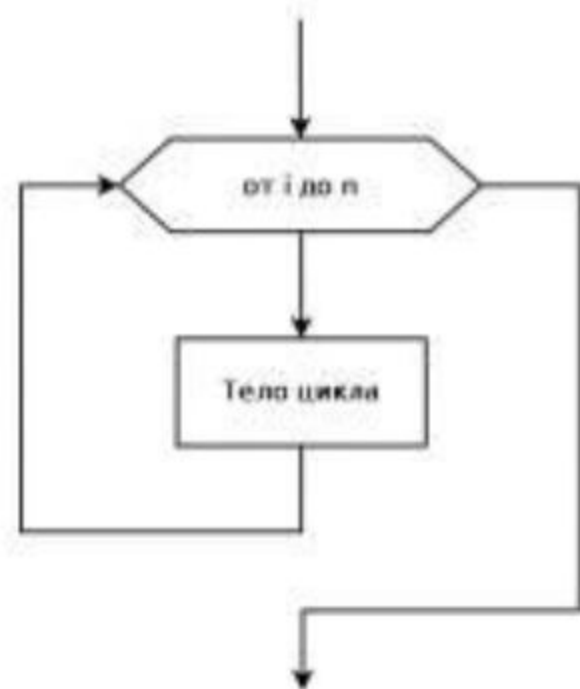
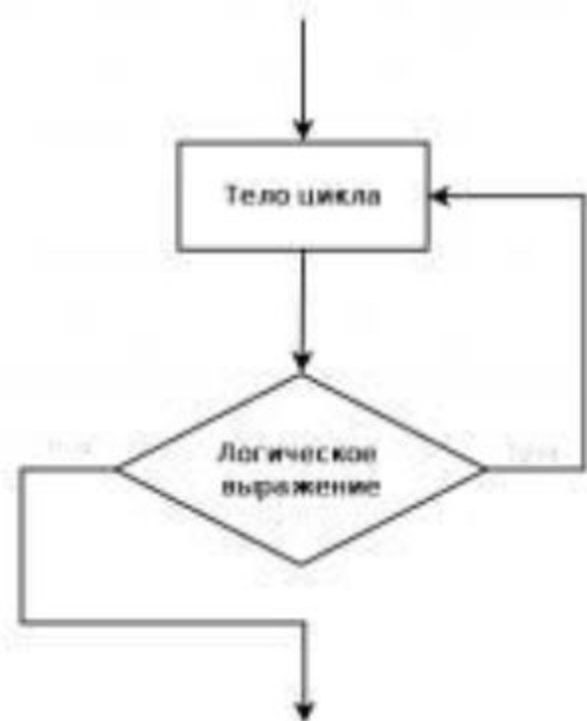
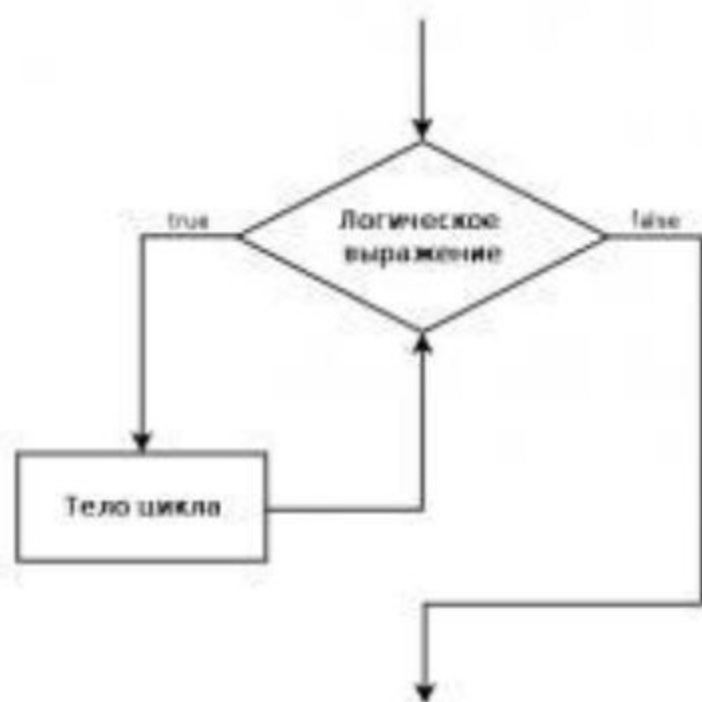
```
int counter=1 //объявление управляющей переменной вне цикла.  
for (; counter <= 7; counter ++) //проверка и изменения  
значения управляющей переменной.  
{  
    cout<<"Вы видите "<<counter<<" чудо света!!!\n";  
    // действия для повторения  
}  
cout<<«\n";
```

Варианты использования **FOR()**

```
int counter=1 //объявление управляющей переменной вне цикла.  
for (; counter <= 7;) //проверка значения управляющей переменной.  
{  
    counter ++; //изменение значения управляющей  
    переменной  
    cout<<"Вы видите "<<counter<<" чудо света!!!\n";  
    // действия для повторения  
}  
cout<<"\n";
```

Варианты использования **FOR()**

```
int counter=1 //объявление управляющей переменной вне цикла.  
for (;;) {  
    if (counter > 7) break; //проверка значения управляющей  
    переменной.  
    counter++; //изменение значения управляющей  
    переменной  
    cout<<"Вы видите "<<counter<<" чудо света!!!\n";  
    // действия для повторения  
    |  
}  
cout<<"\n";
```



Управление циклом

break – позволяет прервать цикл повторений независимо от условий цикла;

continue – позволяет прервать выполнение текущей итерации и начать следующую итерацию.

Цикл **FOR** повторяет действия заданное количество раз

for (инициализация переменной; проверка условия; изменение переменной)

```
{
```

```
    cin>>n;
```

```
    if (n<5) break;
```

```
    cout << i;
```

```
}
```

Цикл **FOR** повторяет действия заданное количество раз

```
int n;  
for (int i=0; i<20; i++)  
{  
    cout<<"Введите n"  
    cin>>n;  
    if (n<5) break;  
    cout << i << endl;  
}
```

Цикл **FOR** повторяет действия заданное количество раз

```
int n;  
for (int i=0; i<20; i++)  
{  
    cout<<"Введите n"  
    cin>>n;  
    if (n<5) continue;  
    cout << i << endl;  
}
```