## **Вложенная конструкция.**

В прошлых уроках вы познакомились с конструкцией под названием цикл и вариантами реализации цикла в языке С. Как вы уже успели заметить, цикл является одной из основополагающих конструкций программирования. С его помощью решается огромное количество задач. Также, вы уже столкнулись с тем, что в цикл можно вкладывать конструкции логического выбора, такие, как if и switch. Однако, не будем останавливаться на достигнутом и, попробуем вложить в цикл подобную ему конструкцию, т. е. - другой цикл. Рассмотрим простой пример:

#include <iostream>

using namespace std;

void main ()

{

int i=0,j;

while(i<3){

cout<<"\nOut!!!\n";

j=0;

while(j<3){

cout<<"\nIn!!!\n";

j++;

}

i++;

}

cout<<"\nEnd!!!\n";

}

Проанализируем пример:

1. Программа проверяет условие i<3, так как 0 меньше 3 условие является истинным и программа входит во внешний цикл.

2. Осуществляется показ на экран ***Out!!!***

3. Обнуляется переменная j.

4. Теперь проверяется условие j<3, так как 0 меньше 3 условие является истинным и программа входит во внутренний цикл.

5. Осуществляется показ на экран ***In!!!***

6. Осуществляется изменение управляющей переменной j.

7. Снова проверяется условие j<3, так как 1 меньше 3 условие является истинным и программа входит во внутренний цикл.

8. Осуществляется показ на экран ***In!!!***

9. Осуществляется изменение управляющей переменной j.

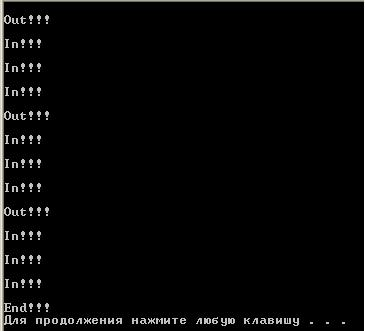
10. Снова проверяется условие j<3, так как 2 меньше 3 условие является истинным и программа входит во внутренний цикл.

11. Осуществляется показ на экран ***In!!!***

12. Осуществляется изменение управляющей переменной j.

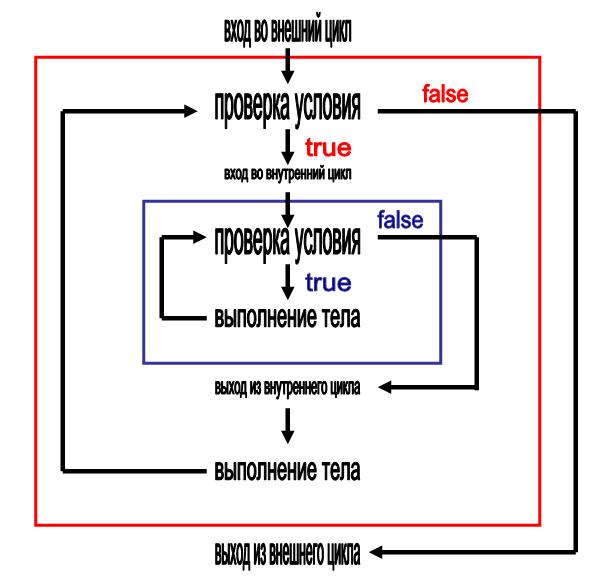
13. Снова проверяется условие j<3, так как 3 не меньше 3 условие является ложным и программа выходит из внутреннего цикла.

Далее код возвращается к пункту 1. Все вышеописанные действия (1-13) повторятся 3 раза, т. е. до тех пор пока i не станет равно значению 3. После этого программа выйдет из внешнего цикла и на экран выведется ***End!!!***



Принцип работы программы, реализующей вложенный цикл основан на том, что внутренний цикл полностью выполняется на каждом шаге внешнего цикла от начала до конца. Другими словами, пока программа не выйдет из вложенного цикла - выполнение внешнего не продолжится.

Схема



Как видите, все просто, но несмотря на это вложенные конструкции значительно упрощают реализацию большинства сложных алгоритмов. Убедитесь в этом, рассматривая следующий раздел урока, в котором мы подготовили для вас несколько примеров

## **Практические примеры.**

### **Пример 1.**

##### **Постановка задачи.**

Написать программу, которая выводит на экран таблицу умножения.

##### **Код реализации.**

#include <iostream>

using namespace std;

void main ()

{

for(int i=1;i<10;i++)

{

for(int j=1;j<10;j++)

{

cout<<i\*j<<"\t";

}

cout<<"\n\n";

}

}

##### **Комментарий к коду.**

1. Управляющие переменные внешнего и внутреннего циклов осуществляют функции множителей.

2. Управляющая переменная i создается и инициализируется значением 1.

3. Программа проверяет условие i<10, так как 1 меньше 10 условие является истинным и программа входит во внешний цикл.

4. Управляющая переменная j создается и инициализируется значением 1.

5. Программа проверяет условие j<10, так как 1 меньше 10 условие является истинным и программа входит во внутренний цикл.

6. Осуществляется показ на экран произведения i на j - ***1***

7. Осуществляется изменение управляющей переменной j.

8. Снова проверяется условие j<10, так как 2 меньше 10 условие является истинным и программа снова входит во внутренний цикл.

6. Осуществляется показ на экран произведения i на j - ***2***

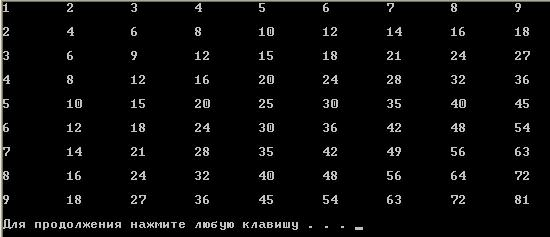
7. Осуществляется изменение управляющей переменной j.

. . .

Действия с 5 по 7 повторяются до тех пор пока j не становится равно 10, при этом текущее значение i (1) умножается на каждое значение j (от 1 до 9 включительно), результат показывается на экран. Получается строка таблицы умножения на 1.

Затем программа выходит из внутреннего цикла и переводит экранный курсор на две строки вниз. После этого, осуществляется увеличение переменной i на единицу и снова вход во внутренний цикл. Теперь уже для вывода цепочки умножения на 2.

Таким образом, в конце концов на экране появляется вся таблица умножения.



### **Пример 2.**

##### **Постановка задачи.**

Вывести на экран прямоугольник из символов 20 на 20.

##### **Код реализации.**

#include <iostream>

using namespace std;

void main(){

int str;

int star\_count;

int length=20;

str=1;

while(str<=length)

{

star\_count=1;

while(star\_count<=length)

{

cout<<"\*";

star\_count++;

}

cout<<"\n";

str++;

}

}

##### **Комментарий к коду.**

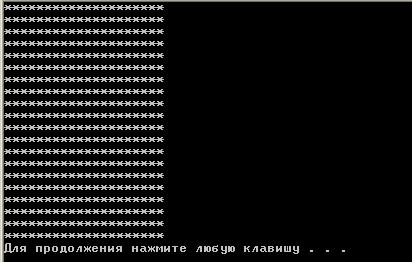
1. Управляющая переменная внешнего цикла - str контролирует количество строк в прямоугольнике.

2. Управляющая переменная внутреннего цикла - star\_count контролирует количество символов в каждой строке.

3. length - длина стороны прямоугольника

4. После отрисовки каждой строки, во внешнем цикле осуществляется переход на следующую строчку прямоугольника.

4. Результат таков:



**Примечание:** Обратите внимания, что несмотря на то, что количество строк соответствует количеству символов в строке - на экране не квадрат!!! Это связано с тем, что высота и ширина символа разные.

Вот и всё!!! Теперь у вас имеется полная информация о циклах, их разновидностях и принципах работы. Но, прежде чем выполнять домашнее задание, следует ознакомиться с еще одним разделом урока. Этот раздел поможет вам не только писать программы, но и анализировать их работу.