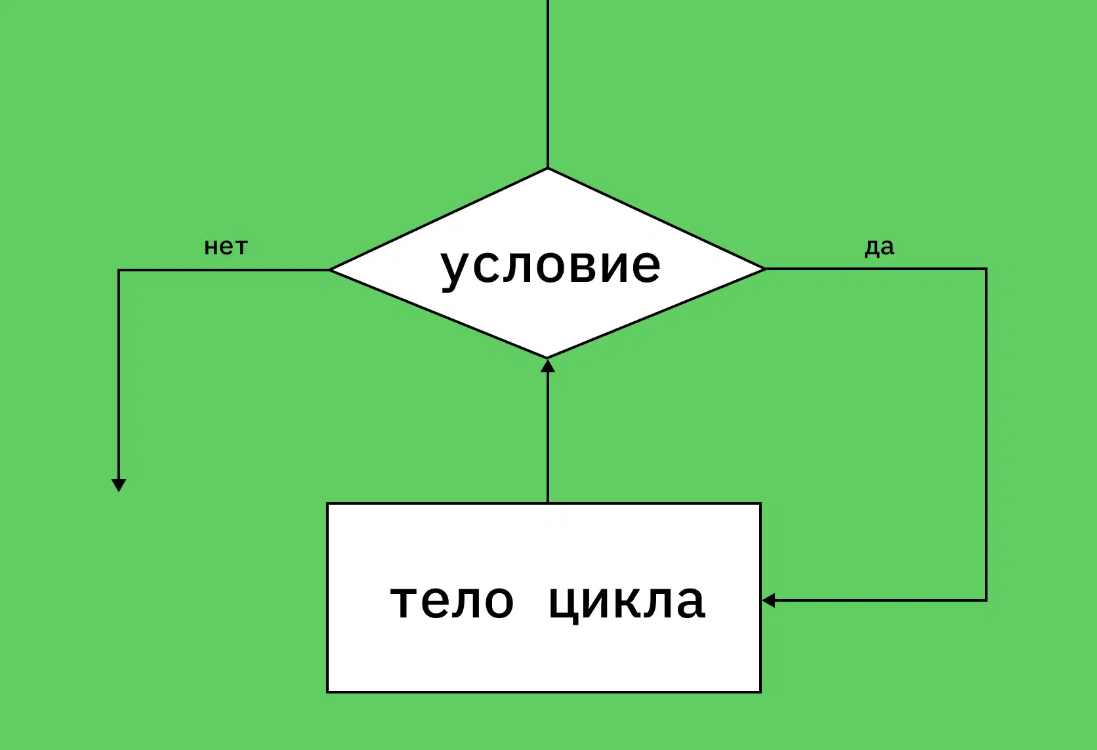
**Лекция 6**

**Циклы в Python**

**1) Что такое циклы?**

В Python циклы работают также, как в большинстве других языков программирования. Программа повторяет определённое действие несколько раз. Действие задано в теле цикла, число повторений — в его условии. В условии может быть вполне конкретное число, например пять или десять. А может быть более сложная проверка — например, анализ переменной из тела цикла. Одно повторение тела цикла называется итерацией.



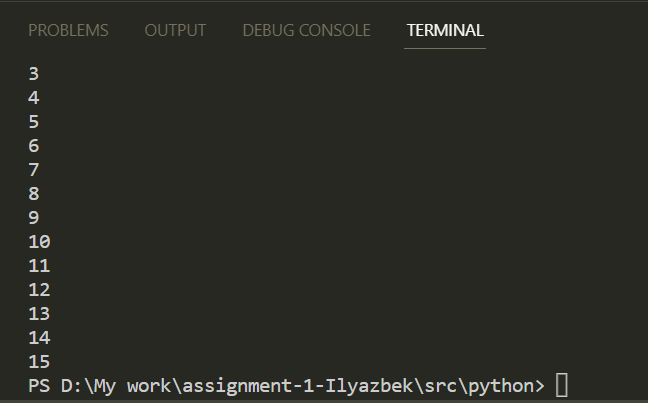
*Рис.1. Схема работы циклов в упрощенном виде*

У циклов в Python есть особенность. Внутри тела цикла можно использовать оператор else, который сработает, когда условие цикла перестанет выполняться. То есть цикл не просто завершится, а перед окончательным выходом совершит последнее действие. Например, в условии задано повторение цикла 25 раз, в теле — вывод чисел из условия последовательно, а в операторе else — слово «Хватит». Тогда цикл сначала выведет по очереди числа от 1 до 25, а потом «Хватит».

**2) Зачем нужны циклы?**

Цикл нужен для того, чтобы упаковать множество повторений в более короткий и легко читаемый код. Например, если нужно вывести числа от 1 до 15, можно 15 раз написать вывод, или сделать цикл, который займет две строки.



Результат в обоих случаях: 

*Рис.2. Сравнение программы без цикла и с циклом*

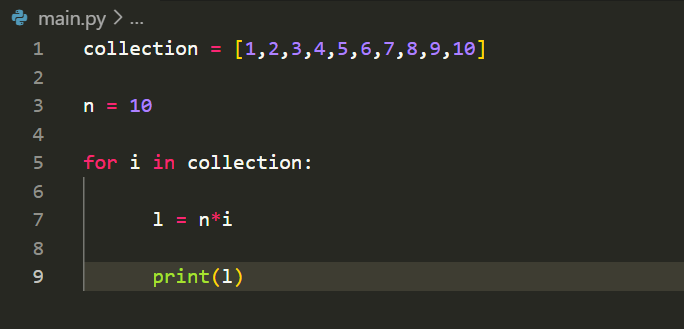
**3) Какие бывают циклы?**

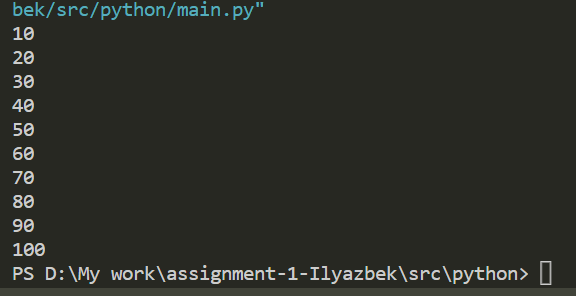
**Цикл for**

«For» с английского переводится как «для». Этот вид цикла используют, когда количество итераций зависит от того, сколько в условии задано элементов.  
Его синтаксис выглядит так:

for [элемент] in [последовательность]:  
[тело цикла]

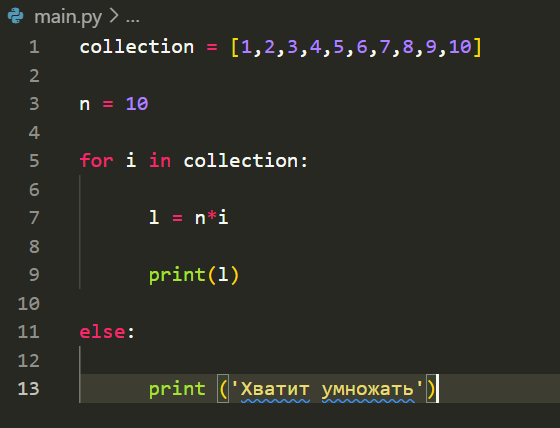
Цикл for в Python повторит тело цикла столько раз, сколько в этой последовательности элементов. Например, вот так:

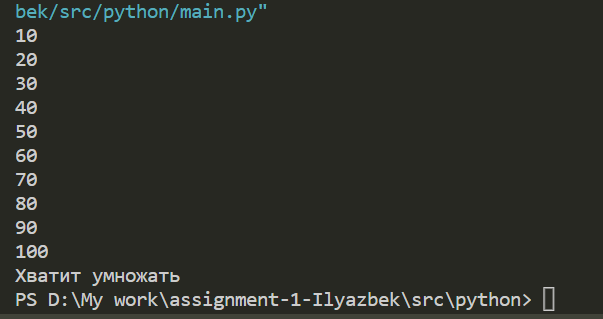


Этот код выведет результаты умножения заданного числа n на элементы из переменной collection по очереди: «10, 20, 30, 40, 50, 60, 70, 80, 90, 100».

Переменные collection и n можно задать вручную, а можно предложить пользователю ввести их с клавиатуры. Тогда цикл получится использовать для умножения и вывода любой вводимой последовательности данных.

Внутрь цикла for можно добавить оператор else, который описывали выше:



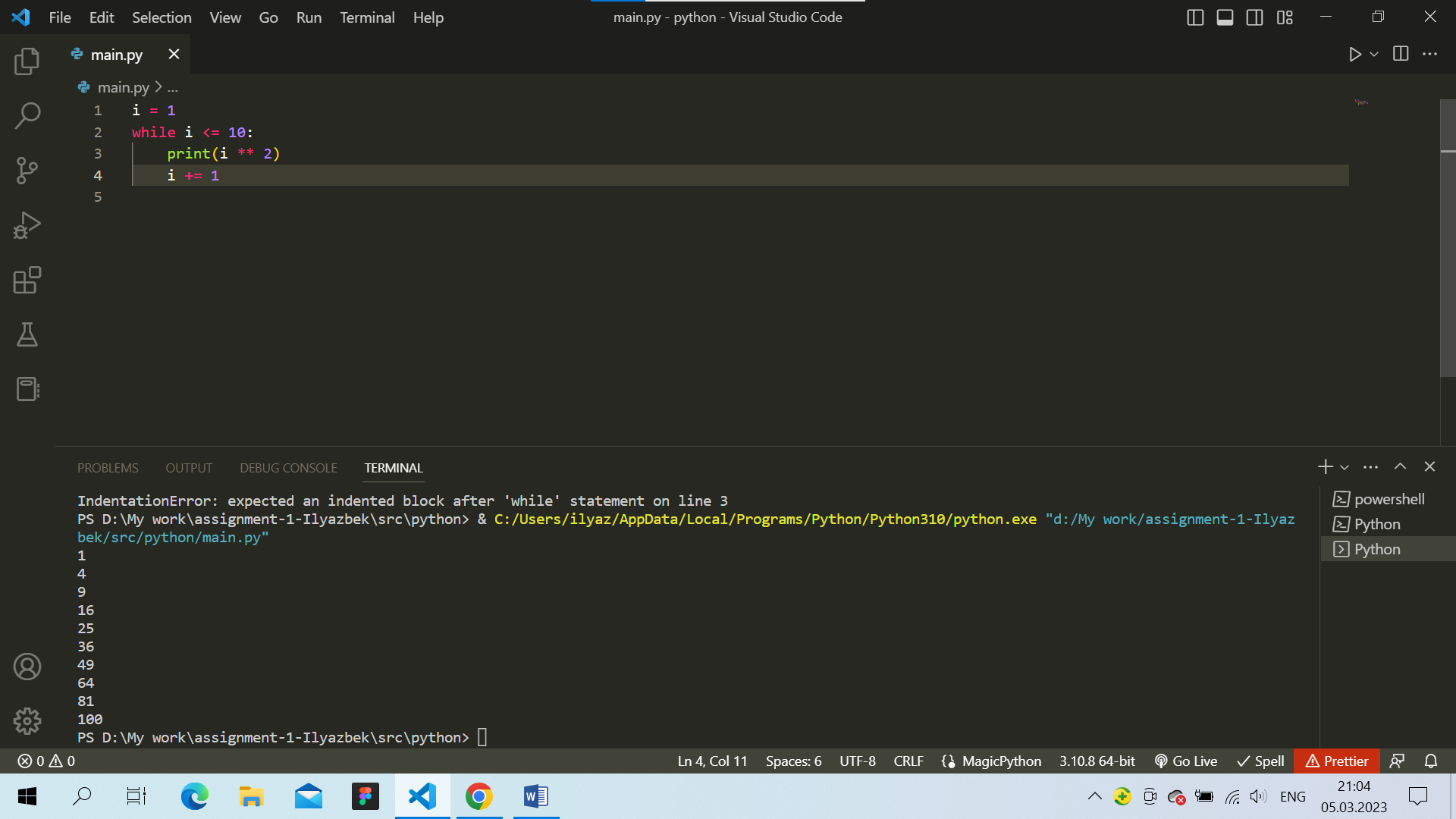


В этом случае в конце итерации по элементам коллекции программа выведет на экран фразу «Хватит умножать», и только потом цикл завершится.   
  
Внутри цикла for нельзя изменять итерируемый объект (переменная i) — это может привести к ошибкам. Его можно использовать для вывода или параметров в промежуточных вычислениях, но новые значения лучше не присваивать. Если нужно изменить переменную, возможно, понадобится другой цикл — while.

**Цикл While**

Цикл while с условием в Python используется, когда точное число повторений неизвестно и может изменяться в зависимости от поведения переменной в теле цикла.  
  
«While» можно перевести с английского как «до тех пор, пока». Этот оператор будет выполнять тело цикла до тех пор, пока выражение в его условии остаётся истинным. Чтобы условие действительно имело шанс стать ложным, в теле цикла нужно изменить переменную, которая может использоваться как счётчик итераций.

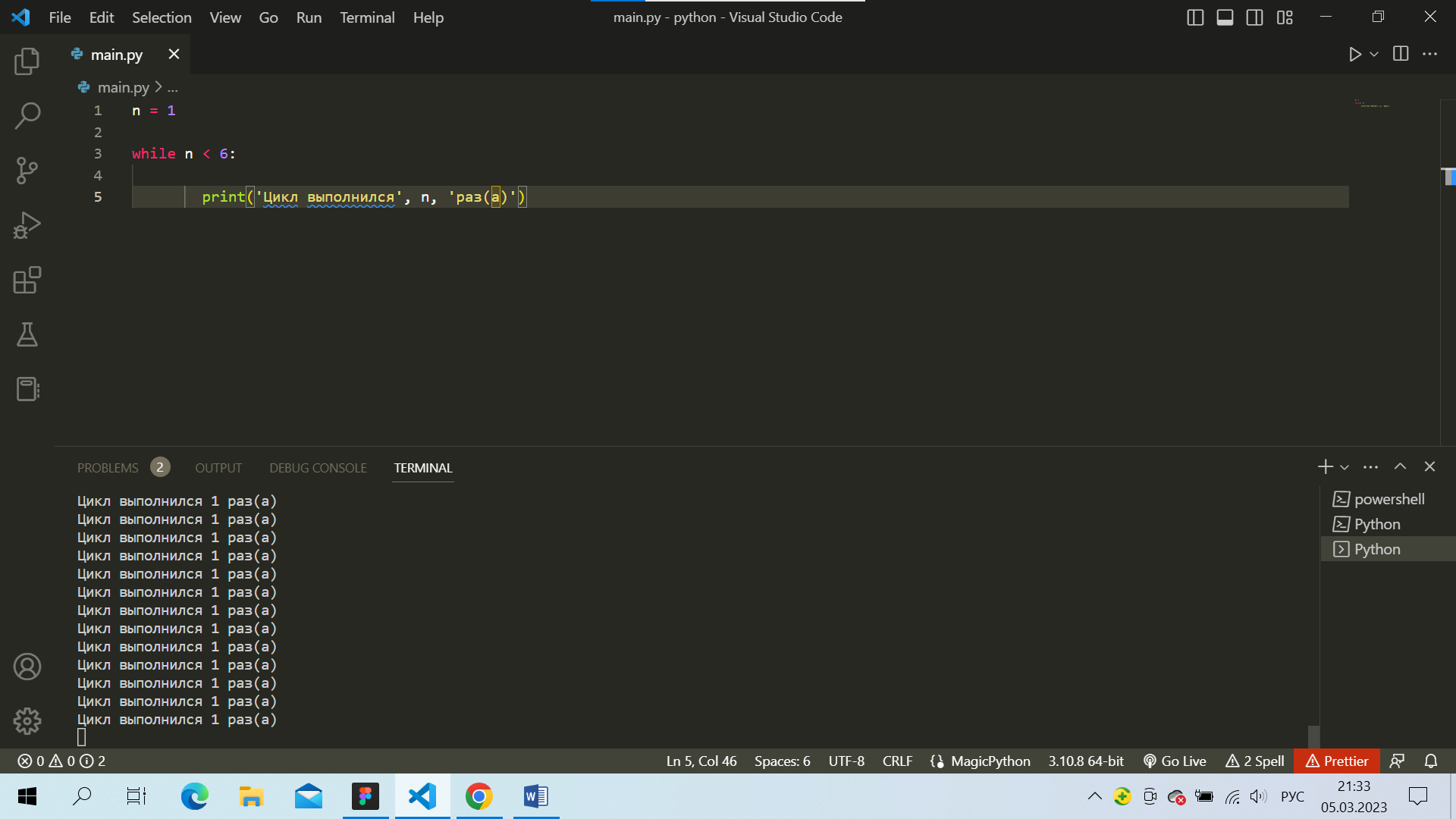
Например, рассмотрим такой код:



В этом примере переменная i внутри цикла изменяется от 1 до 10. Такая переменная, значение которой меняется с каждым новым проходом цикла, называется счетчиком. Заметим, что после выполнения этого фрагмента значение переменной i будет равно 11, поскольку именно при i == 11 условие i <= 10 впервые перестанет выполняться.

**Бесконечные циклы**

В Python цикл while может стать бесконечным. Это происходит, если его условие никогда не становится ложным, то есть момент выхода из цикла не наступит. Например, возьмём код цикла while с некоторыми изменениями:

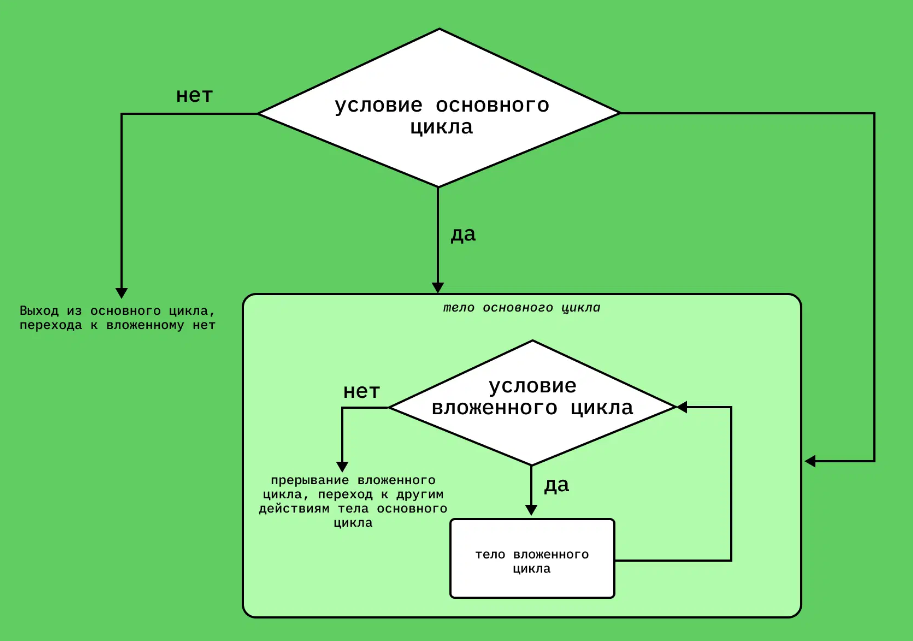
Прибавление к n числа 1 было удалено. В итоге n не будет изменяться и навсегда останется единицей, а цикл никогда не прервётся, и будет снова и снова выводить фразу «Цикл выполнился 1 раз(а)».

Чтобы этого не допустить, при использовании конструкций с циклами нужно проверять, что условие выхода из цикла будет выполнено. Иначе программа зависнет и придётся завершать её принудительно.

Бесконечных циклов стараются избегать и используют редко, например для периодического вывода информации о времени, статусе выполнения программы или блока кода, при логировании и отладке приложения.

**Вложенные циклы**

Внутри тела цикла можно определить и вызвать другой цикл. Его называют вложенным. Внутренний цикл выполнится и будет повторяться столько раз, сколько описано в условии внешнего.

****

Вложенный цикл также может стать бесконечным, если не выполнится условие выхода из него.