**Уважаемые студенты!**

1. **Ход работы**

Вам необходимо выполнить указанные ниже задания, работая в приложении «JupyterNotebook», сохраняя скриншоты кодов в программе «Microsoft Word».

1. **Методические указания**

JupyterNotebook - это веб-приложение с открытым исходным кодом, которое можно использовать для создания и обмена документами, содержащими живой код, уравнения, визуализации и текст. Название Jupyter происходит от основных поддерживаемых языков программирования, которые он поддерживает: Julia, Python и R.

Выполненные задания по програмиированию в среде JupyterNotebookнеобходимо оформить в виде отчета – отдельного файла cназванием:

***«И\_ №группы\_ Фамилия\_Python\_П№13»***

Сохраняем выполненные задания с допустимыми расширениями для файлов: *.docx, .dox.*

Отчёт состоит из элементов:

1. Титульный лист

2. Содержание

3. Скриншоты кодов

4. Список использованных источников

**Задание 8**

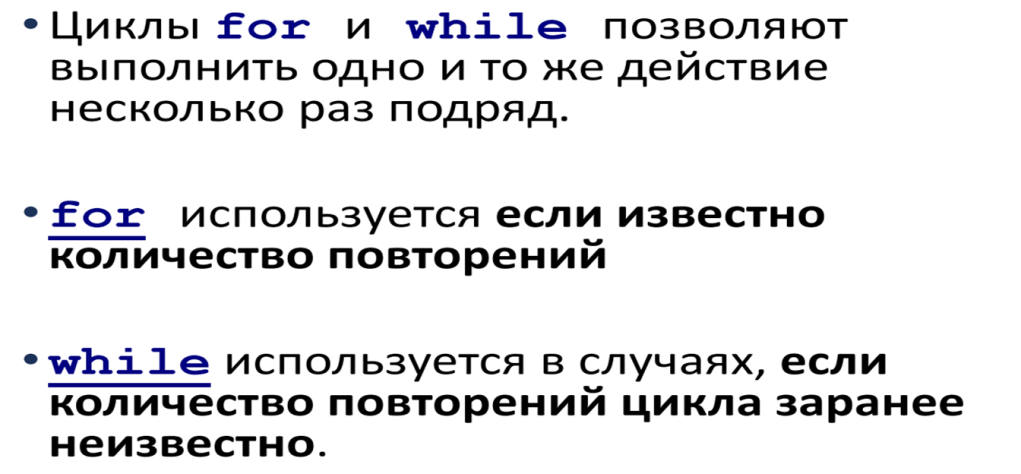
**Тема 3. Операторы цикла в Python.**

**Необходимо ввести в JupyterNotebook все программные коды, представленные в тексте и на рисунках.**

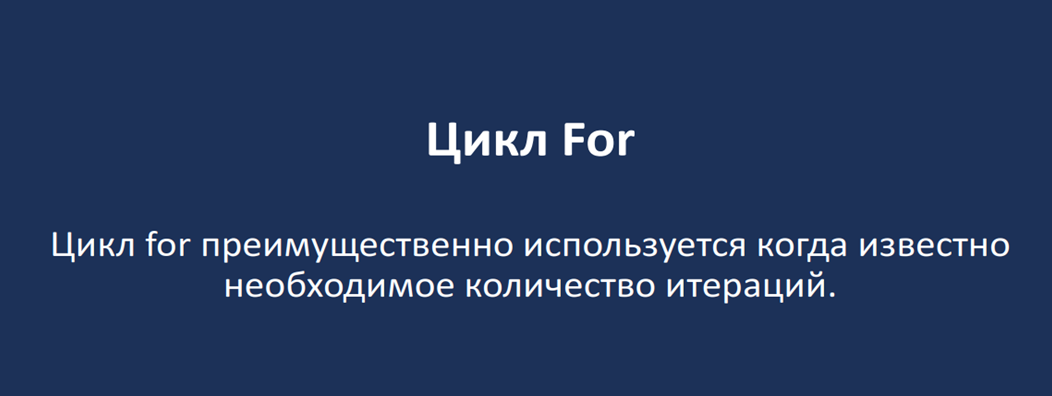
Одно из преимуществ компьютеров перед людьми - способность повторять одни и те же действия многократно, быстро и совсем не утомляясь.

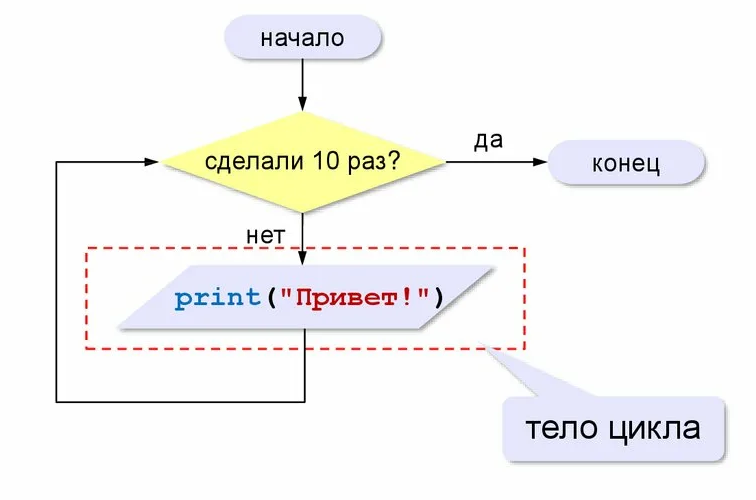
В Python существует две основных разновидности циклов:

* циклы, повторяющиеся определенное количество раз (for, счетные циклы, counting loops);
* циклы, повторяющиеся до наступления определенного события (while, условные циклы, conditional loops).



**1. Цикл for**

****

****

Структура цикла for в Python выглядит так:

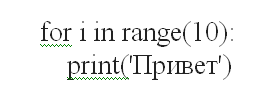
**for** название\_переменной\_цикла **in** range (*количество повторений*):

**блок кода**

Двоеточие (:) в конце строки с инструкцией for сообщает интерпретатору Python, что дальше находится блок команд. В блок команд входят все строки, расположенные с отступом от строки с инструкцией for, вплоть до следующей строки без отступа. **Блок команд**, который выполняется в цикле for, называется телом цикла.

Цикл for замечательно работает, если мы заранее знаем, сколько повторений (итераций) нам требуется сделать.

Рассмотрим код, который распечатает 10 раз слово "Привет":



Привет

Привет

Привет

Привет

Привет

Привет

Привет

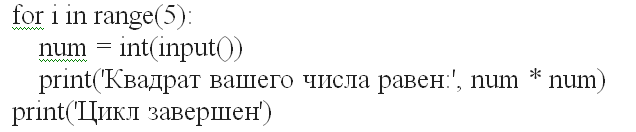
Привет

Привет

Привет

С помощью цикла for можно считывать и обрабатывать сколько угодно чисел.

Рассмотрим следующий программный код:



2

Квадрат вашего числа равен: 4

3

Квадрат вашего числа равен: 9

4

Квадрат вашего числа равен: 16

5

Квадрат вашего числа равен: 25

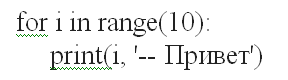
6

Квадрат вашего числа равен: 36

Цикл завершен

Такая программа считывает 5 чисел и выводит на экран их квадраты вместе с поясняющей надписью. Поскольку вторая и третья строки выделены отступом, Python считает, что это тело цикла, которое выполняется 5 раз. Четвертая строка не содержит отступа, поэтому не является частью цикла и будет выполнена всего один раз, после того как цикл завершится.

Когда цикл впервые начинает работу Python устанавливает значение переменной цикла i = 0. Каждый раз когда мы повторяем тело цикла Python увеличивает значение переменной на 1



Результатом выполнения такого кода будет:

0 -- Привет

1 -- Привет

2 -- Привет

3 -- Привет

4 -- Привет

5 -- Привет

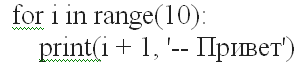
6 -- Привет

7 -- Привет

8 -- Привет

9 -- Привет

Если мы хотим начать с 1, то можем написать код:



Результатом выполнения такого кода будет:

1 -- Привет

2 -- Привет

3 -- Привет

4 -- Привет

5 -- Привет

6 -- Привет

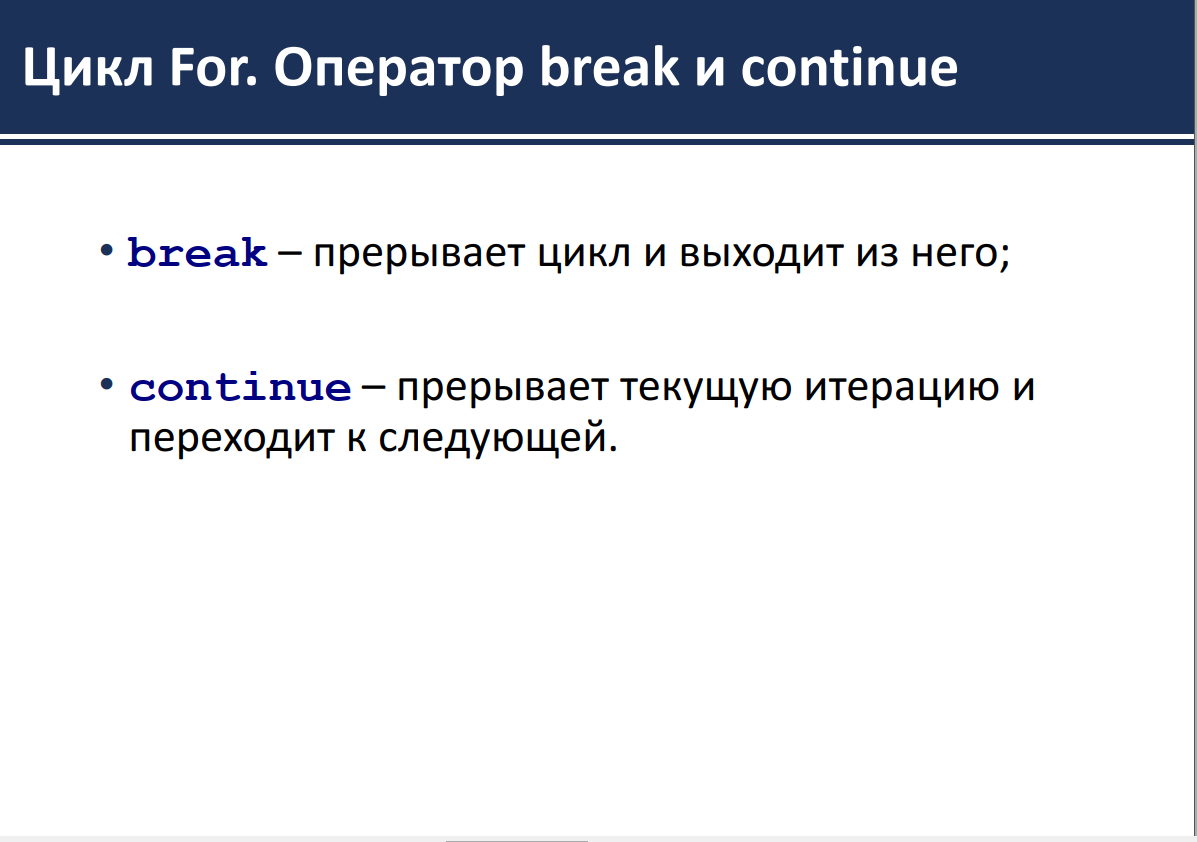
7 -- Привет

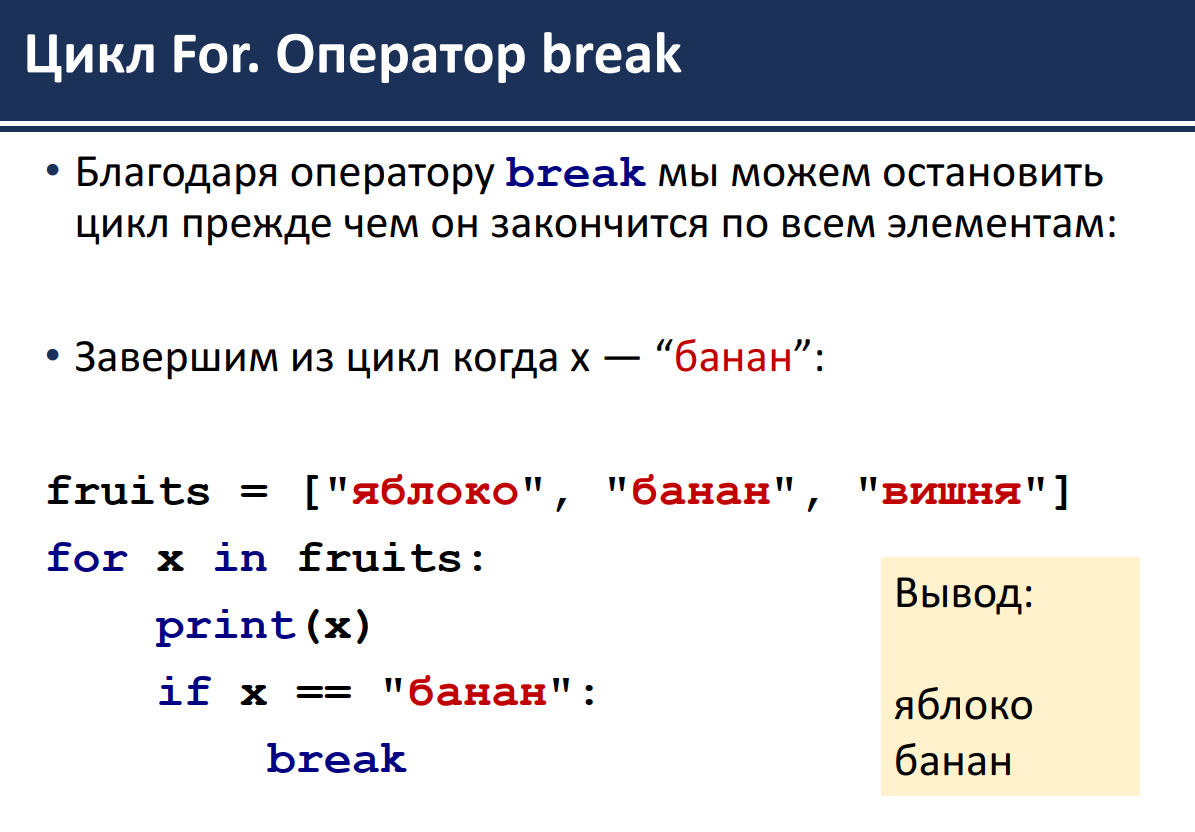
8 -- Привет

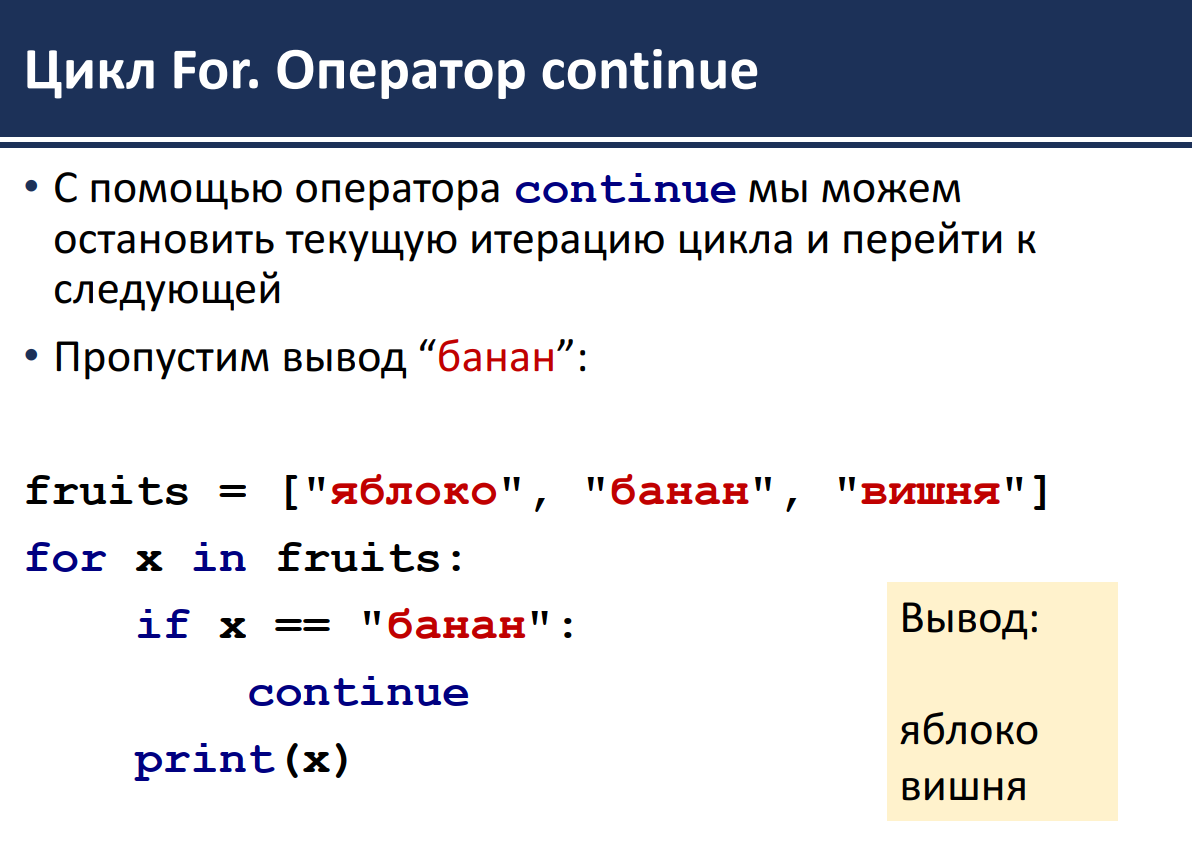
9 -- Привет

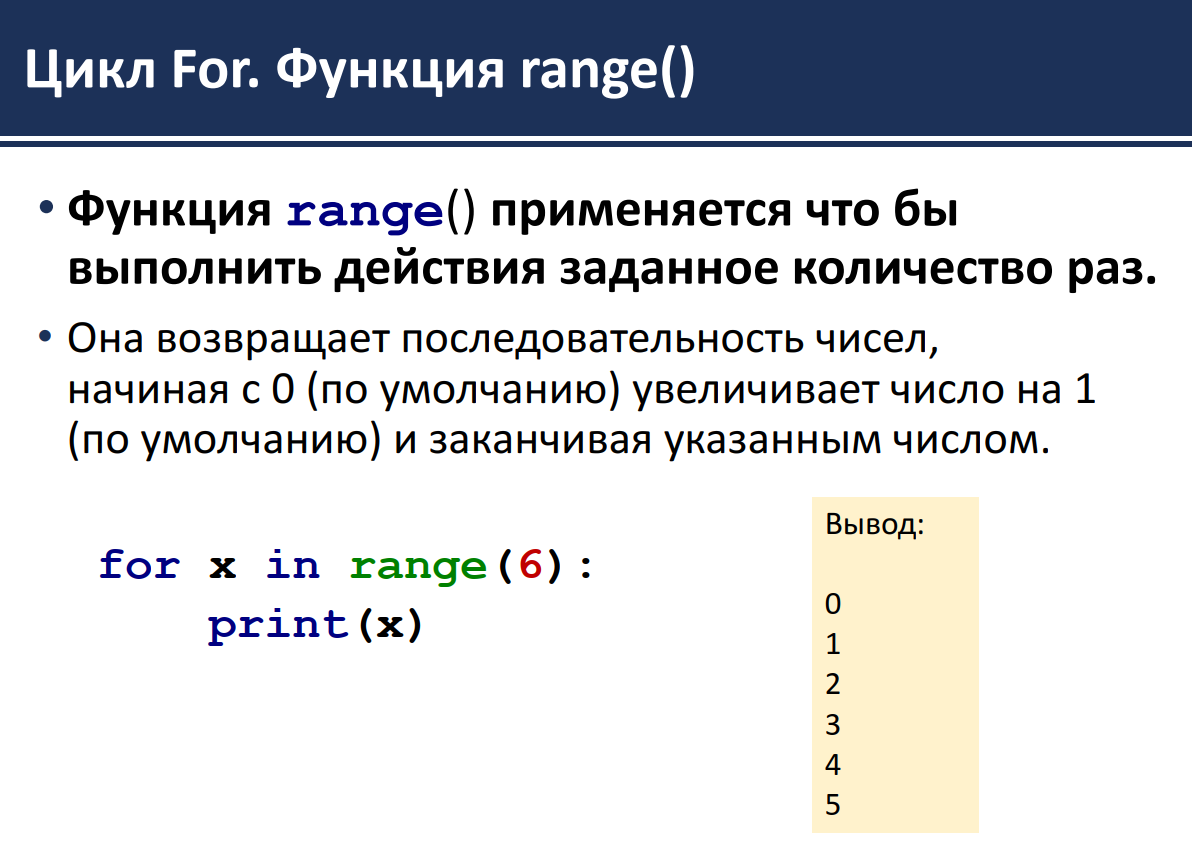
10 -- Привет

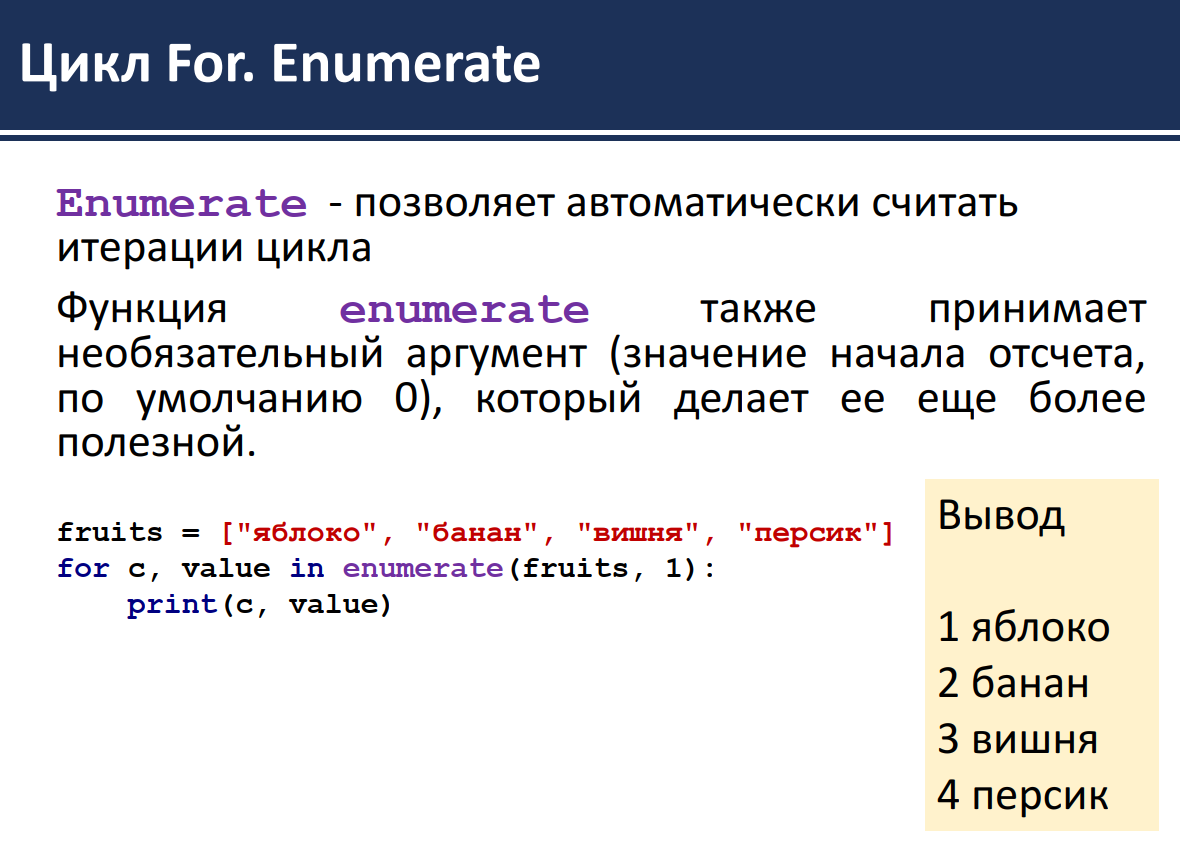
Обратите внимание, за счет выражения i + 1, мы начинаем вывод с 1, а не с 0.

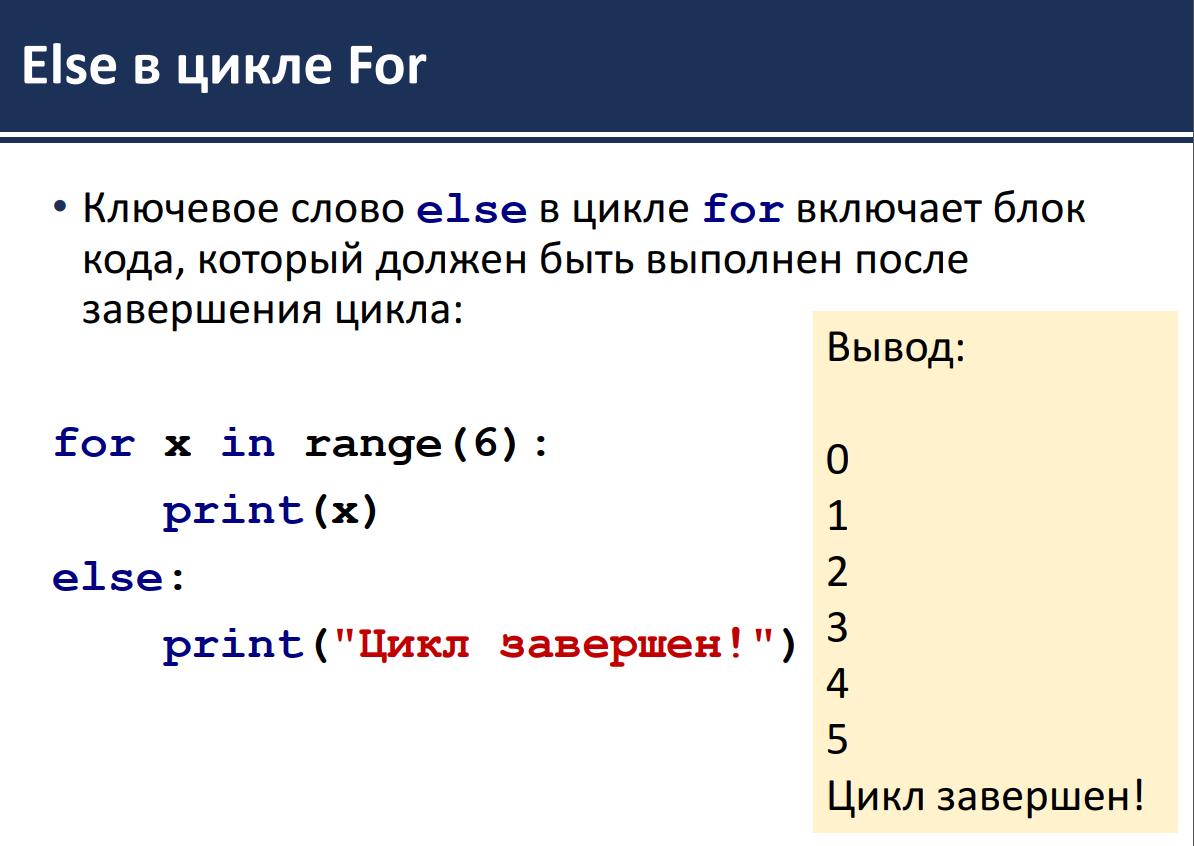


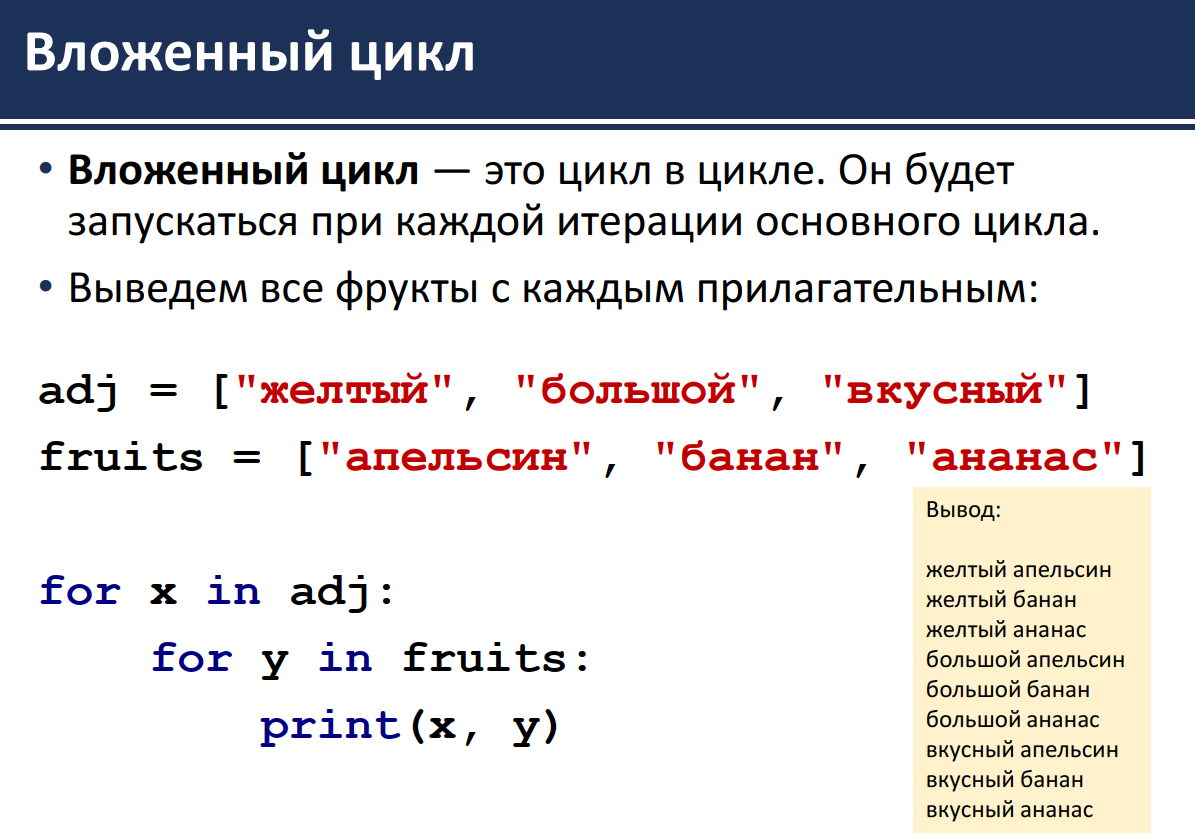




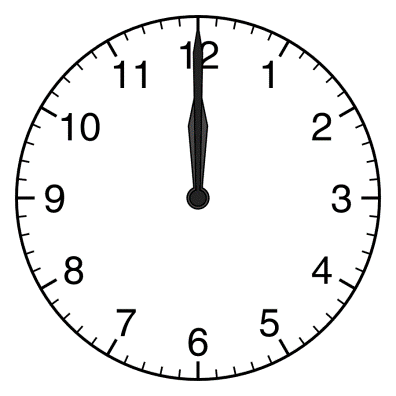








Вложенный цикл расположен в еще одном цикле. Часы служат хорошим примером работы вложенного цикла. Секундная, минутная и часовая стрелки вращаются вокруг циферблата. Часовая стрелка смещается всего на 1 шаг для каждых 60 шагов минутной стрелки. И секундная стрелка должна сделать 60 шагов для 1 шага минутной стрелки. Это означает, что для каждого полного оборота часовой стрелки (12 шагов), минутная стрелка делает 720 шагов.



Рассмотрим цикл, который частично моделирует электронные часы. Он показывает секунды от 0 до 59:

for seconds in range(60):

print(seconds)

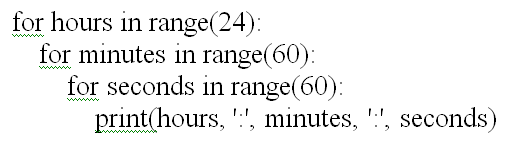
Можно добавить переменную minutes и вложить цикл написанный выше внутрь еще одного цикла, который повторяется 60 раз:

for minutes in range(60):

for seconds in range(60):

print(minutes, ':', seconds)

Для того, чтобы сделать модель законченной, можно добавить еще одну переменную для подсчета часов:



Результатом работы такого кода будет:

0 : 0 : 0

0 : 0 : 1

0 : 0 : 2

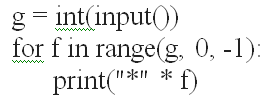
...

23 : 59 : 58

23 : 59 : 59

Самый внутренний цикл сделает 60 итераций для каждой итерации среднего цикла. Средний цикл сделает 60 итераций для каждой итерации самого внешнего цикла. Когда самый внешний цикл сделает 24 итерации, средний сделает 24 \cdot 60 = 144024⋅60=1440 итераций, и самый внутренний цикл сделает 24 \cdot 60 \cdot 60 = 8640024 ⋅60⋅60=86400 итераций!

На вход программе подается натуральное число n (n≥2) – катет прямоугольного равнобедренного треугольника. Напишите программу, которая выводит звездный треугольник в соответствии с примером.

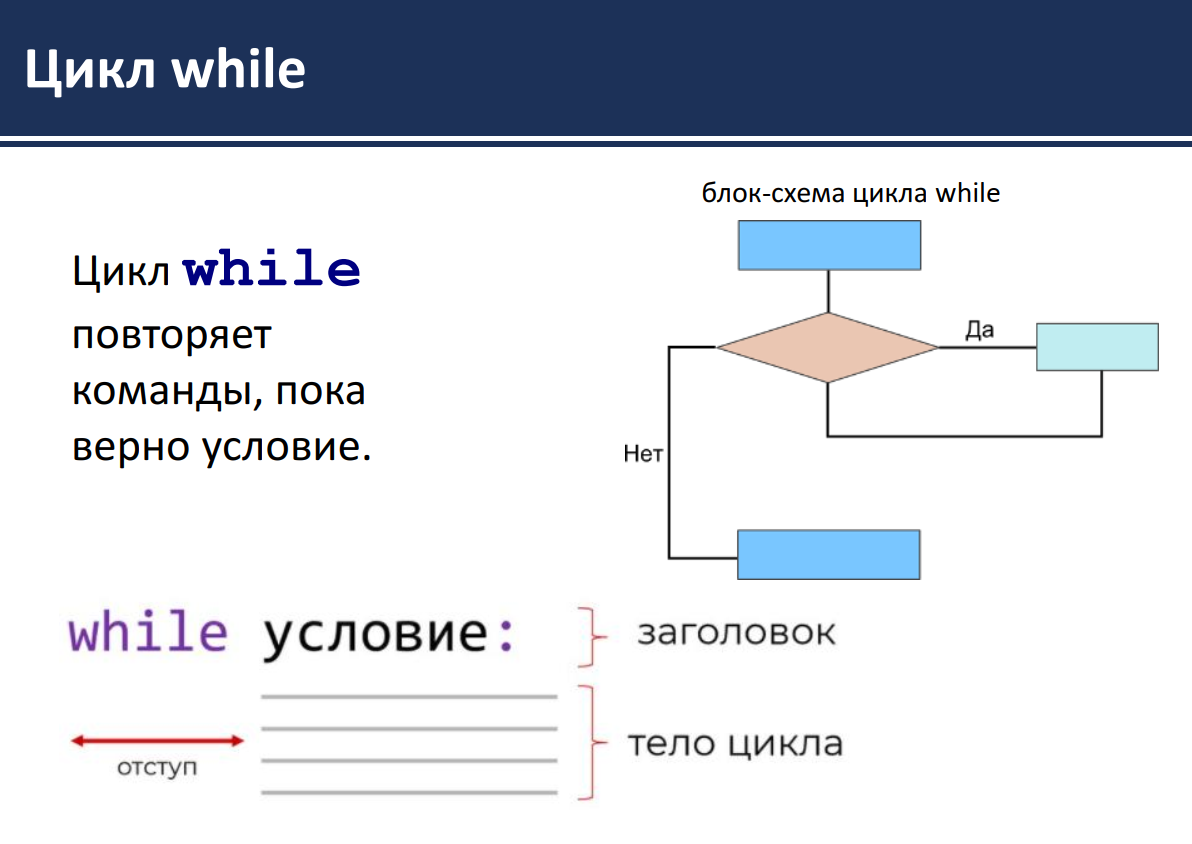


При запросе введите число 30.

**2. Цикл while**

Мы уже рассмотрели работу цикла for, который является счетным циклом. Цикл for замечательно работает, если мы заранее знаем, сколько повторений (итераций) нам потребуется сделать. Но иногда нужно, чтобы цикл выполнялся до наступления некоторого события, и количество итераций в этом случае заранее оценить просто невозможно. И здесь на помощь приходит цикл while.



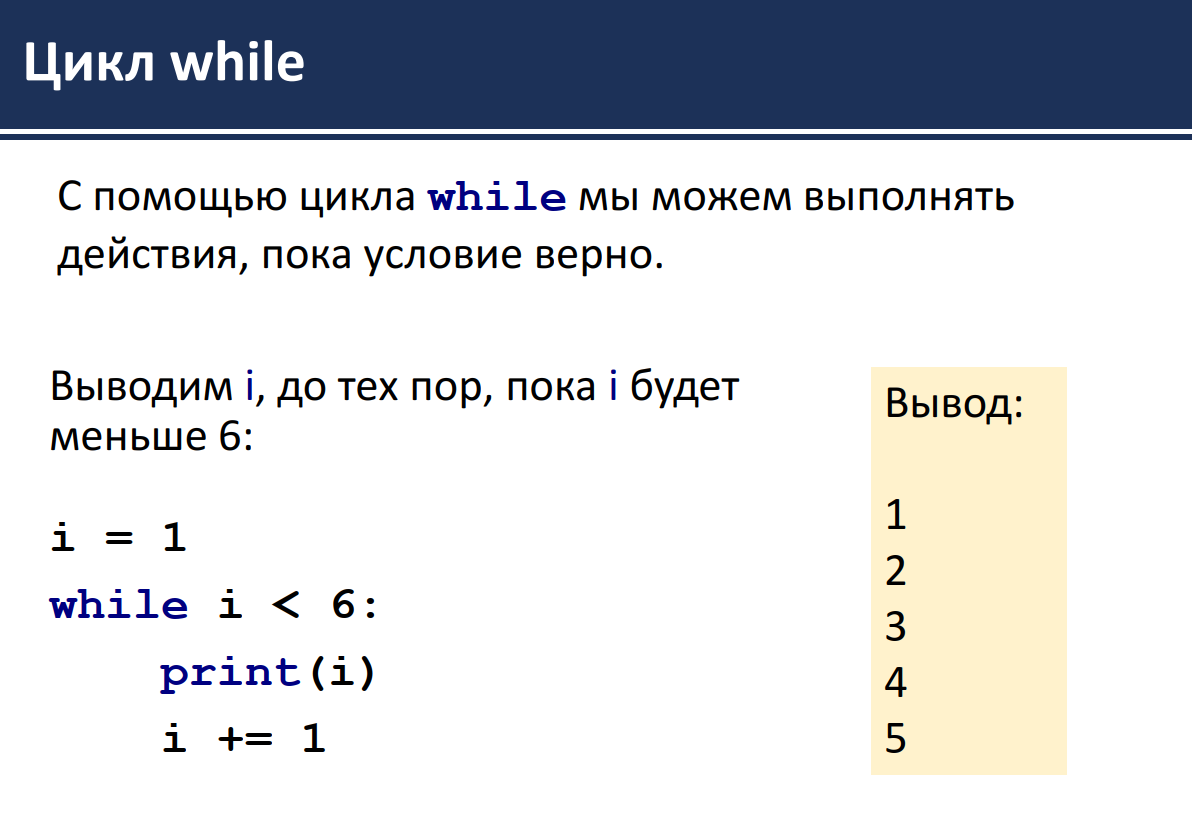


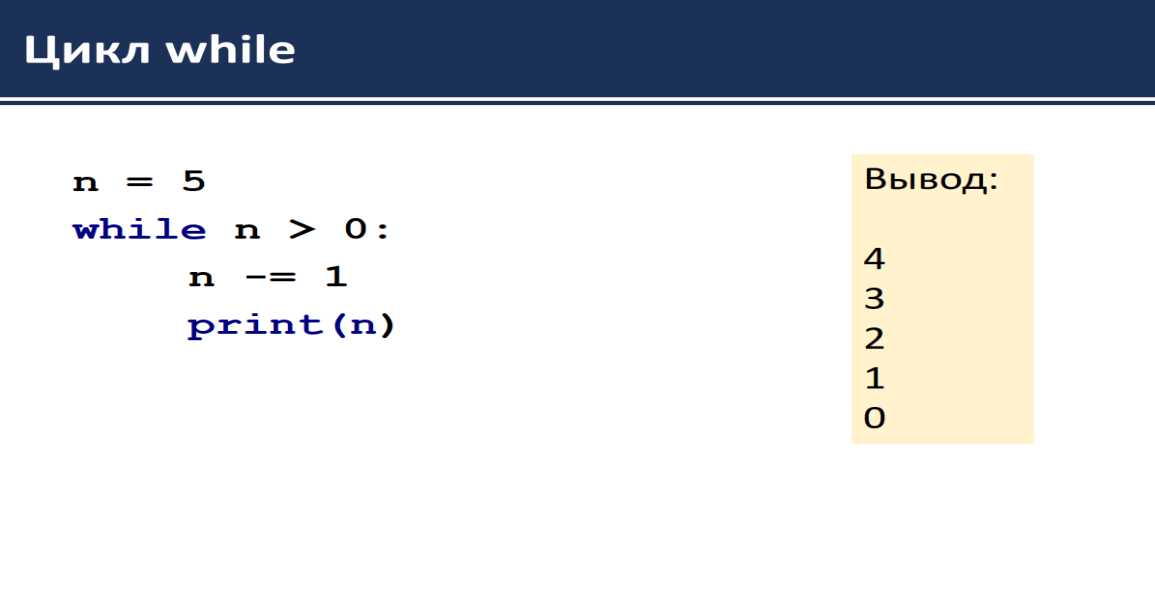
Структура цикла while в Python выглядит так:

**while** условие:

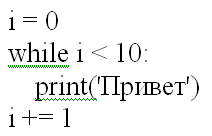
**блок кода**

Двоеточие (:) в конце строки с инструкцией while сообщает Python, что дальше находится блок команд. В блок входят все строки, расположенные с отступом от строки с инструкцией while, вплоть до следующей строки без отступа. **Блок команд**, который выполняется в цикле while, называется телом цикла.



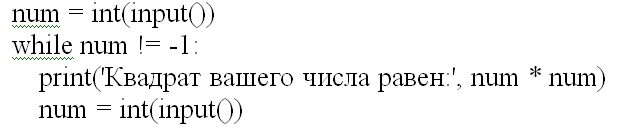


Рассмотрим код, использующий цикл while, который распечатает 10 раз слово Привет:



Такой код можно легко заменить циклом for, поскольку мы заранее знаем сколько раз нужно выполнить тело цикла. Однако так бывает не всегда.

Напишем программу, которая считывает числа и выводит их квадраты, пока не будет введено -1. При такой постановке задачи мы не можем воспользоваться циклом for, так как не знаем сколько чисел будет предшествовать числу -1.



В качестве начального значения переменной num, мы используем первое из чисел. Далее пока выполняется условие цикла, а именно, пока введенное число не равно -1, мы исполняем тело цикла. В тело цикла входит две команды:

1. напечатать квадрат введенного числа;
2. считать следующее число.

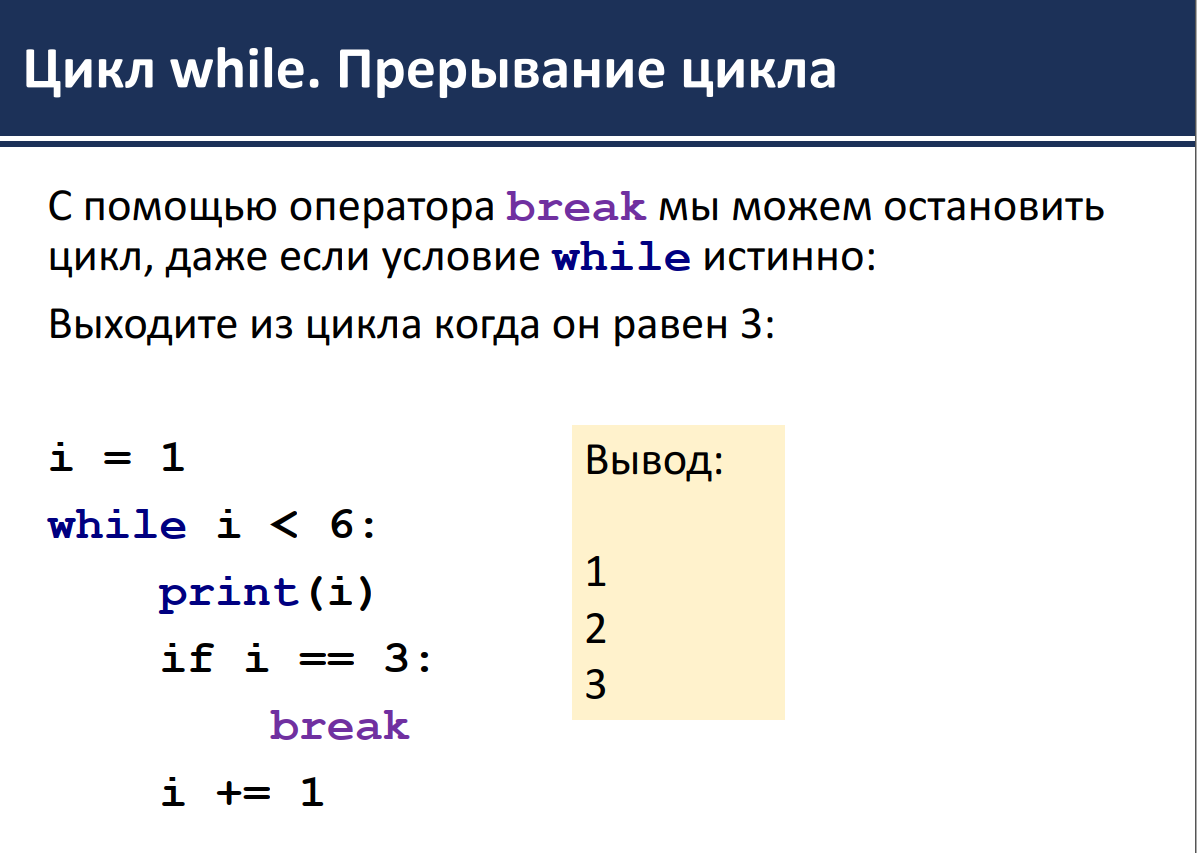
Важным являются два момента:

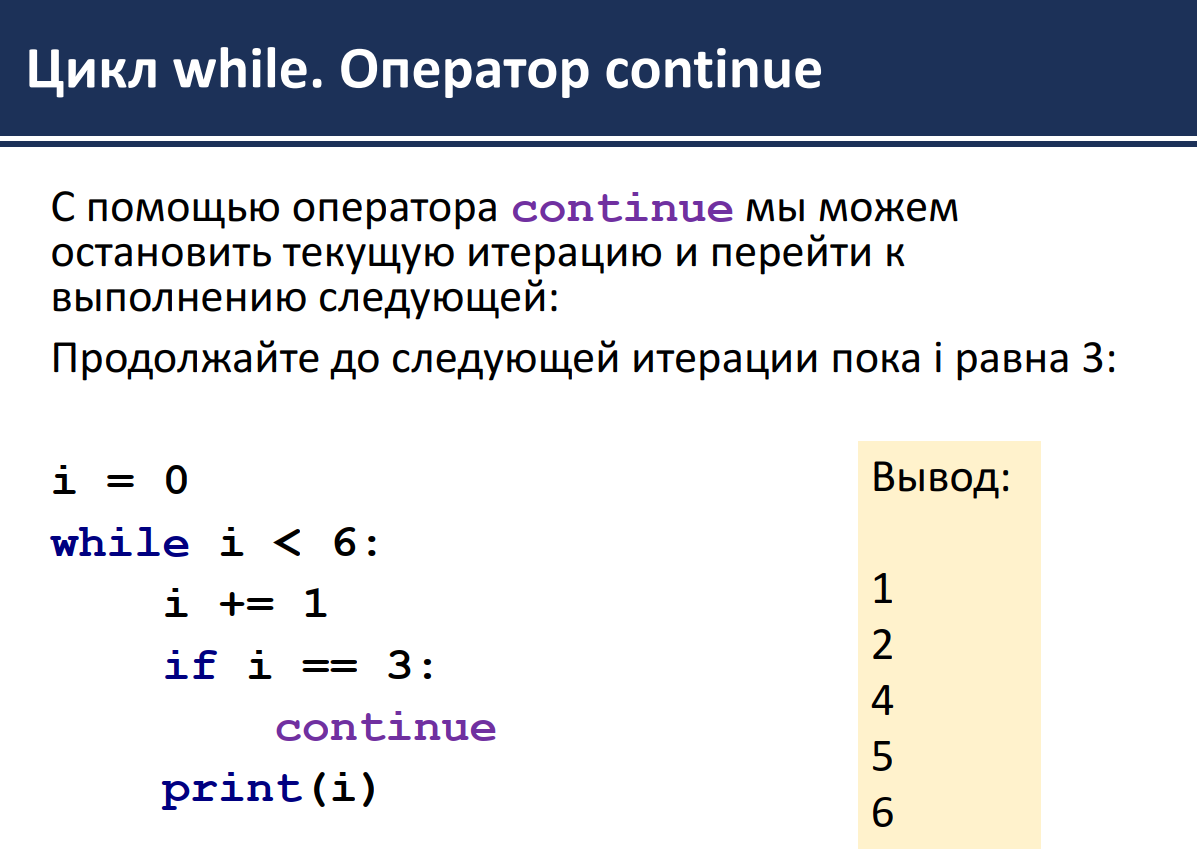
1. правильная инициализация переменной num;
2. изменение переменной num внутри цикла while.

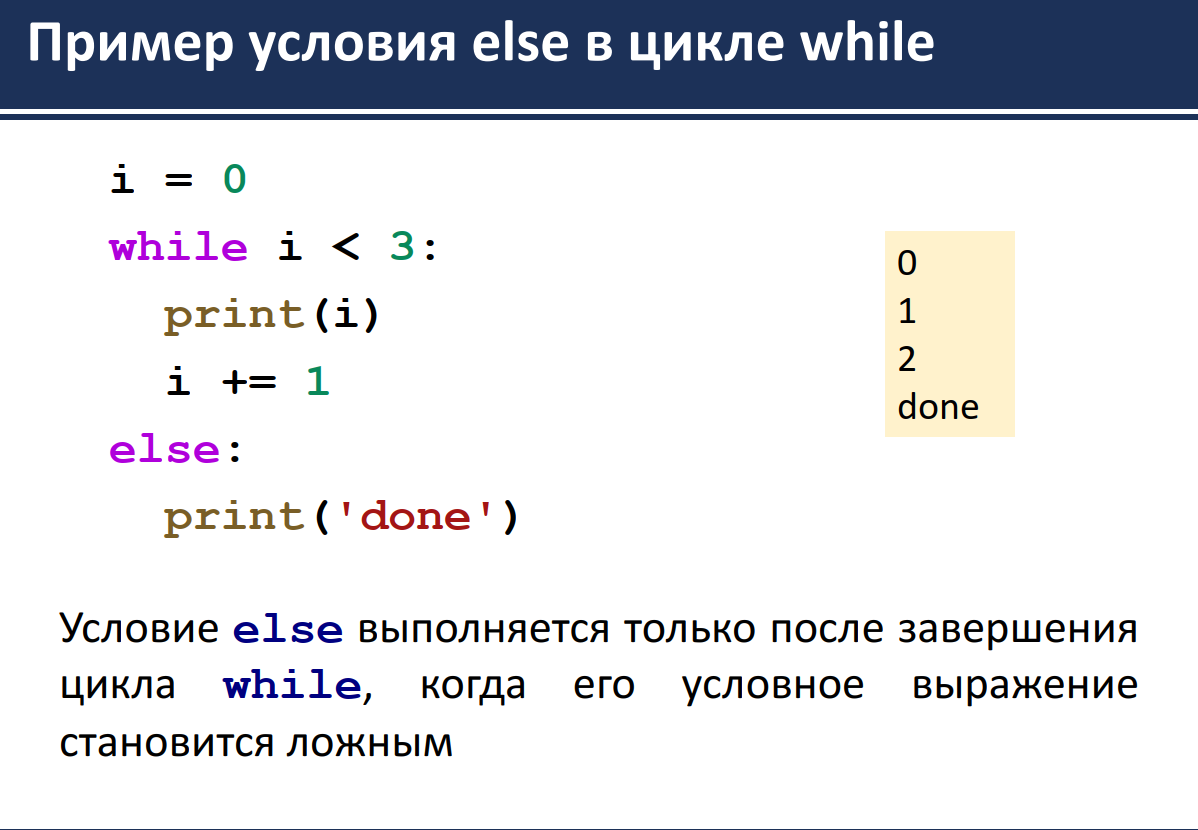
Важно: если не изменять переменную num внутри цикла, то можно получить так называемый бесконечный цикл, который будет выполняться бесконечно много раз.

Всегда, кроме редких случаев, цикл while должен содержать возможность завершиться. То есть в цикле что-то должно сделать проверяемое условие ложным. Если цикл не имеет возможности завершиться, то он называется бесконечным циклом. Бесконечный цикл продолжает повторяться до тех пор, пока программа не будет прервана. Бесконечные циклы обычно появляются, когда программист забывает написать программный код внутри цикла, который делает проверяемое условие ложным. В большинстве случаев следует избегать применения бесконечных циклов.

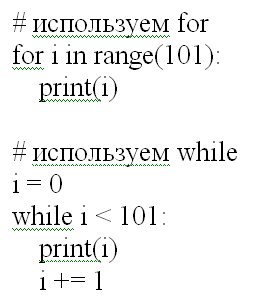
Цикл while очень похож на условный оператор if. Разница заключается в том, что в случае с условным оператором соответствующий блок кода будет выполняться только один раз, тогда как с циклом while блок кода будет выполнен многократно.





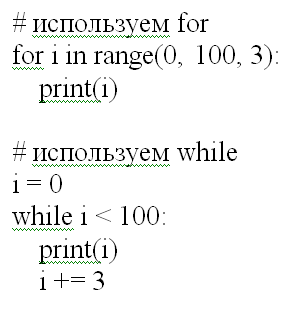


Мы всегда можем заменить цикл for с помощью цикла while. Следующие две программы выводят числа от 0 до 100:



В первом цикле переменная i последовательно принимает значения от 0 до 100. Для цикла while, нам пришлось завести самостоятельно переменную i и придать ей начальное значение. Тело цикла while содержит аналогичную инструкцию вывода print(i), однако помимо этого мы самостоятельно увеличиваем значение переменной i на 1, что делается автоматически в случае с циклом for.

Напишем программу, выводящую все числа, кратные 3, используя цикл for и while:



Не всегда, однако, удается заменить цикл while с помощью цикла for. Если заранее не известно количество итераций, то необходимо использовать цикл while и только его.

***Результат по теме 3***

***Уважаемые студенты!***

*Пожалуйста, сохраните выполненное задание в файле:*

*«И\_ №группы\_ Фамилия\_ Python\_П№13»*

Задания необходимо оформить в виде отчета, который состоит из элементов:

1. Титульный лист

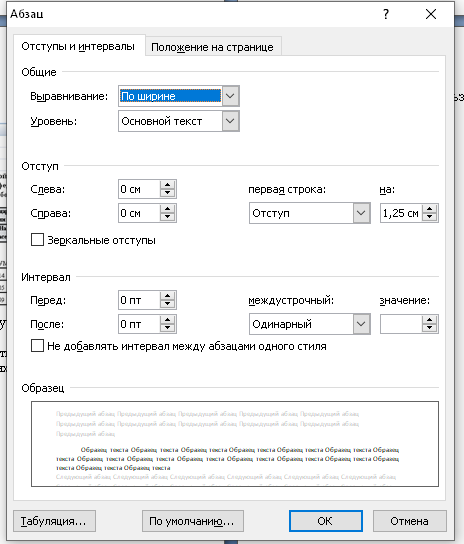
2. Содержание

3. Скриншоты кодов

4. Список использованных источников

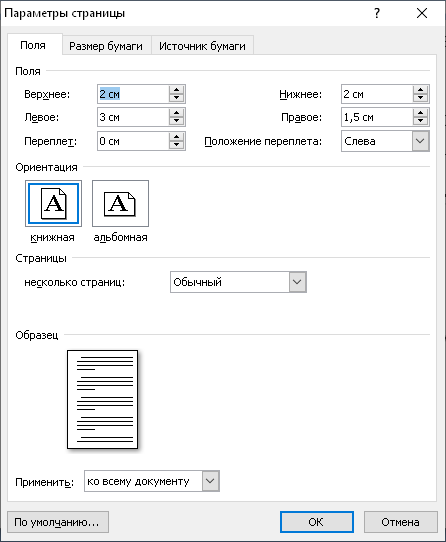
Соблюдайте требования к оформлению:

1. Шрифт- **TimesNewRoman**, размер – **14**



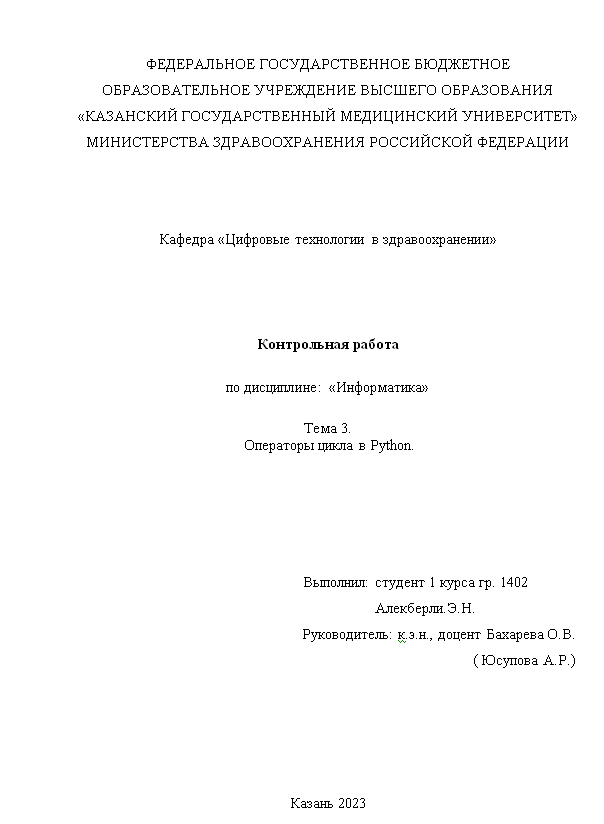
2. **Нумерация страниц** (титульный лист - без номера, содержание – 2 страница)

3. В параметрах страниц должны быть выставлены следующие значения:



4. Содержание создаётся с помощью инструмента **Автоматическое оглавление**. Содержание ОБЯЗАТЕЛЬНО оформить: Шрифт- TimesNewRoman, размер – 14, цвет – чёрный.

5. Каждая подтема должна начинаться с новой страницы, используем **Разрыв страницы**.



Содержание

1. Цикл for

2. Цикл while

Список использованных источников

1. Гуриков,С.Р.ОсновыалгоритмизацииипрограммированиянаPython: учеб.пособие/С.Р.Гуриков. —Москва:ФОРУМ: ИНФРА-М,2018. —343с.

2. Трофимов, В.В. Алгоритмизация и программирование: учебник для вузов /В.В.Трофимов,Т.А.Павловская;подредакциейВ.В.Трофимова.—Москва: Издательство Юрайт, 2021. — 137 с. — (Высшее образование). — ISBN 978-5-534-07834-3. — Текст : электронный // ЭБС Юрайт [сайт]. — URL:https://urait.ru/bcode/471125.