Лекция 4

Циклы

Цикл

Цикл — разновидность управляющей конструкции в высокоуровневых языках программирования, предназначенная для организации многократного исполнения набора инструкций.

Последовательность инструкций, предназначенная для многократного исполнения, называется **телом цикла**.

Единичное выполнение тела цикла называется **итерацией**. Выражение, определяющее, будет в очередной раз выполняться итерация, или цикл завершится, называется **условием выхода** или условием окончания цикла (либо условием продолжения в зависимости от того, как интерпретируется его истинность — как признак необходимости завершения или продолжения цикла).

Типы циклов в Python

В Python существует два вида циклов:

while - наиболее универсальный вид цикла. Обеспечивает повторение последовательности операторов до тех пор, пока истинно условие продолжения цикла.
for - узкоспециализированная версия цикла. Предназначен для прохода по итерируемым последовательностям.

В этой лекции мы рассмотрим использование цикла **while**, а цикл **for** мы рассмотрим, когда изучим списки.

Для чего нужны циклы

Для чего же все-таки они нужны - эти циклы?

Предположим, что перед вами поставили задачу вывести на экран все числа от 1 до 100. Написать 100 строк с функцией print() явно не самый хороший выход.

Так вот, когда нужно повторить какую-то последовательность операторов много раз, и используются циклы.

python

Правило использования цикла while

```
while Логическое_условие:
   Оператор 1
   Оператор 2
   …….
```

Работа цикла начинается с проверки Логического_условия.

Если результат этой проверки вернет False, то тело цикла просто пропускается. Если проверка вернет True, то сначала выполнится последовательность операторов (Оператор 1, Оператор 2 ….).

После выполнения операторов, выполнение опять вернется в точку проверки Логического_условия и все повторится.

python

Пример использования цикла while

```
number = 1

while number <= 100:
    print(number)
    number = number + 1
```

Логическое условие продолжения цикла

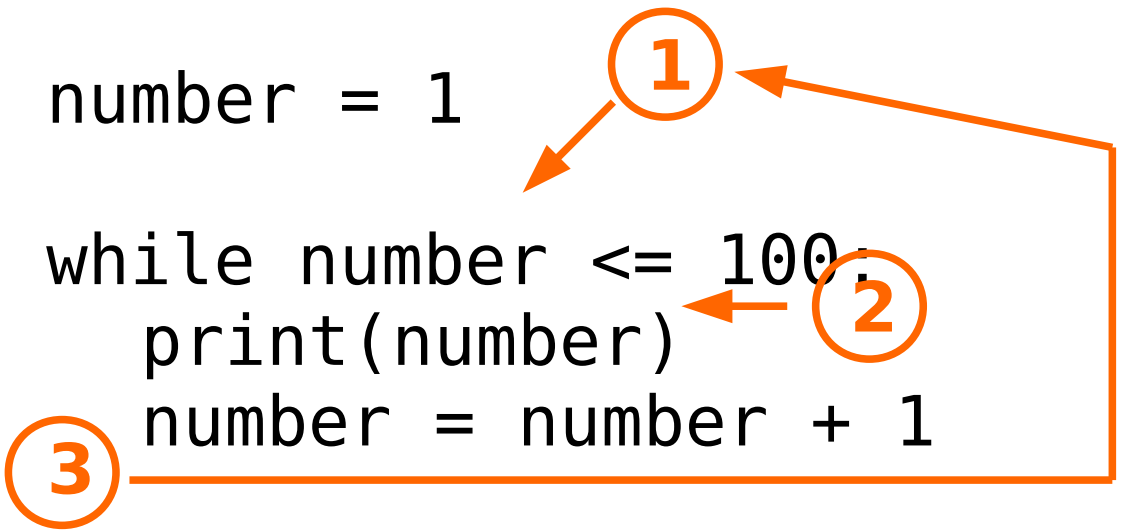
Тело цикла

Вначале значение переменной number равно 1. Поэтому, при проверке условия, оно вернет True. В теле цикла на экран выводится значение этой переменной и происходит увеличение ее значения на 1. Таким образом, условие будет истинным еще 100 итераций. Однако, когда number станет равна 101, то цикл завершится, так как условие вернет False.

Помните, что условие продолжения цикла должно хоть когда-то нарушиться. В противном случае произойдет «зацикливание» программы. Т.е. бесконечное выполнение одного цикла.

python

Подробнее о работе цикла while



1. Проверка условия.
2. Если условие истинно, выполнить операторы, ложно - закончить.
3. Возврат к проверке условия - переход к пункту 1.

Некоторые вопросы о цикле while

Если что-то общее между логическими условиями циклов и условных операторов?

Они абсолютно идентичны. Т.е. правила для условных операторов одинаковы и для циклов. Их можно объединять с помощью булевых операторов и т. д.

Сколько и какие операторы можно использовать в теле цикла while?

В теле цикла вы можете использовать любой оператор Python. Их количество не ограниченно. Их набор вы определяете сами.

Операторы досрочного прекращения цикла

В случае, когда нужно пропустить одну итерацию цикла или закончить цикл, можно использовать операторы прерывания цикла :

continue — прерывает текущую итерацию цикла и сразу переходит к следующей.

break — прекращает выполнение цикла.

python

Пример использования операторов прерывания цик

```
number = 0
while number < 10:
    number = number + 1
    if number == 5:
        continue    ← Прерывание итерации
    if number == 7:
        break       ← Прерывание цикла
    print(number)
```

Дополнение блока while оператором else

Как и условный оператор **if**, так и цикл **while** можно дополнить блоком **else**. Принцип работы такого оператора довольно интересен. Блок else выполнится лишь в том случае, если цикл завершится нормально. Если же цикл будет прерван с помощью оператора break или еще каким образом, то этот блок будет пропущен.

```
while Логическое_условие:
   Оператор 1
   Оператор 2
   …….
else:
   Что делать, если цикл завершился нормально
```

Пример использования блока while / else

```
number = int(input("input positive number "))
i = 2
while i < number:
   if number % i == 0:
      print("It is not a prime number")
      break
   i = i + 1
else:
   print("It is a prime number")
```

Этот блок выполнится, только если break в цикле не будет вызв

Эта программа определяет, является ли число, которое пользователь вводит с клавиатуры, простым.

Использование вложенных циклов while

Как и условный оператор if, так и цикл while можно вложить внутрь другого цикла while. Получится цикл в цикле. Например, такой цикл пригодился бы, если нужно пройтись по таблице. Внешний цикл передвигался бы по строкам, а внутренний по столбцам.

```
while Логическое_условие_1:
  Оператор внешнего цикла
  Оператор внешнего цикла
  while Логические_условие_2:
    Оператор вложенного цикла
    Оператор вложенного цикла
  …….
```

python

Пример вложенных циклов while

```
a = int(input("Input a "))
b = int(input("Input b "))
i = 0
while i < a:
    j = 0
    while j < b:                    ← Вложенный цикл
        print("*", end=' ')
        j += 1
    print()
    i += 1
```

Этот пример кода выведет на экран прямоугольник из символов «*». Его ширина составляет b звездочек. За это ответственен внутренний цикл while. Высота составляет a звездочек. За обрисовку строк ответственен внешний цикл while.

python

Дополнительный параметр для функции print

Функция print может принимать несколько именованных (имеющих имя) параметров. Один из них - **end**. Он используется для обозначения того, чем должен заканчиваться вывод данных на экран. По умолчанию используется символ перевода строки. Однако если вы не хотите, чтобы выполнялся перенос, то можете указать любой строковый символ.

```
print("*", end=' ')
```

Теперь строка не будет переведена, а будет поставлен пробел.

Список использованной литературы

1)Лутц М. Изучаем Python, 4-е издание. – Пер. с англ. – СПб.: Символ-Плюс, 2011 - 392 - 400 с.
2)Саммерфилд М. Программирование на Python 3. Подробное руководство. - Пер. с англ. - СПб.:Символ-Плюс, 2009. - 190 - 191 с.

python

Домашнее задание

1)Выведите на экран все числа в диапазоне от 1 до 100 кратные 7.

2)Вычислить с помощью цикла факториал числа n введенного с клавиатуры (4<n<16). Факториал числа - это произведение всех чисел от этого числа до 1. Например, 5!=5*4*3*2*1=120

3)Напечатайте таблицу умножения на 5. Предпочтительно печатать 1 x 5 = 5, 2 x 5 = 10, а не просто 5, 10 и т. д.

4)Выведите на экран прямоугольник из *. Причем, высота и ширина прямоугольника вводятся с клавиатуры. Например, ниже представлен прямоугольник с высотой 4 и шириной 5.

```
*****
*   *
*   *
*****
```

Дополнительное домашнее задание.

1) С помощью цикла (только одного) нарисовать такую фигуру. Причем, максимальная высота этой фигуры вводится с клавиатуры (в примере максимальная высота — 4.)

```
*
**
***
****
***
**
*
```

2) С помощью циклов вывести на экран все простые числа от 1 до 100. Простое число — число, которое делится нацело только на единицу или само на себя. Первые простые числа это — 2,3,5,7…

3) Выведите на экран «песочные часы», максимальная ширина которых считывается с клавиатуры (число нечетное). В примере ширина равна 5.

```
*****
 ***
  *
 ***
*****
```