

PYTHON | БЕЗ ВОДЫ

Практический курс по Python

#Блок 3 | Циклы

Для быстрого старта

О курсе

- ▶ Это ПРАКТИЧЕСКИЙ КУРС по программированию
- максимальная польза от курса в самостоятельном решении наибольшего числа задач
- вопросы на занятиях – самое ценное для учеников
- разбор задач учеников НА ЗАНЯТИЯХ
- разбор вопросов по задачам НА ЗАНЯТИЯХ

Регламент

- ▶ Длительность одного занятия 90 минут (может +)
- ▶ 1 занятие - теория | 3 занятия - практика
- ▶ Регулярность занятий 1 раз в неделю
- ▶ Проходим блоками
- ▶ Занятия по скайпу
- ▶ Запись занятий
- ▶ Видео доступно на youtube в закрытом плейлисте

#1 | Что такое циклы

Для быстрого старта

Азбука

▶ Циклы:

- специальные конструкции, позволяющие запускать один и тот же фрагмент кода несколько раз

▶ Зачем нужны:

- чтобы не писать многократно один и тот же код и потом править каждую запись
 - делать программы компактными и эффективными
 - реализовывать алгоритмы
 - работать с итерируемыми объектами
- ▶ повторяющийся код – кандидат в цикл

Азбука

```
1 print(0)
2 print(1)
3 print(2)
4 print(3)
5 print(4)
6 print(5)
7 print(6)
8 print(7)
9 print(8)
10 print(9)
11 print(10)
12 print(11)
13 print(12)
14 print(13)
15 print(14)
16 print(15)
```

```
1 for i in range(16):
2     print(i)
3
4
5
6
7
8
9
10
11
12
13
14
15
16
```



Азбука

▶ Тело цикла:

- Код внутри цикла

▶ Итерация цикла:

- одно выполнение кода внутри цикла



Азбука

► Общий вид цикла:

Название цикла + проверка условия | перебор элементов коллекции:

ТАВ тело цикла

```
text = 'Hello world'  
i = 0  
while i < len(text):  
    print(text[i])  
    i += 1
```

```
text = 'Hello world'  
for i in range(len(text)):  
    print(text[i])
```

Азбука

- ▶ управляющие команды цикла:
- `break` – принудительное завершение текущего цикла
- `continue` – принудительный переход к следующей итерации

```
1 while a > 10**10:  
2     a = int(input('Введите число: '))  
3     if a == 0:  
4         break  
5     elif a > 200:  
6         continue  
7     else:  
8         print(a)
```

#2 | Цикл `While`

Для быстрого старта

Джентельменский набор для быстрого старта

▶ Цикл While:

- Цикл с предусловием
- Дает возможность выполнять набор инструкций до тех пор, пока соответствующее условное выражение возвращает True

```
a = 123456789
while a > 0:
    print(f'{a%10}', end=" ")
    a //= 10
```

- внутри while = внутри if
- конструкции внутри цикла (тело цикла) оформляются отступами (tab)

Джентельменский набор для быстрого старта

▶ Как пошагово работает цикл while:

- Проверяется выражение после while
- Если True, тело выполняется
- Снова производится проверка условия. Если оно по-прежнему возвращает истину, тело выполняется вновь, и т. д.
- Тело цикла будет выполняться до тех пор, пока условное выражение не вернет значение False
- Как только это произошло, тело цикла будет пропущено, и выполнение программы возобновится с первой инструкции, следующей после тела While

```
a = 123456789
while a > 0:
    print(f'{a%10}', end=" ")
    a //= 10
```

Джентельменский набор для быстрого старта

▶ Пример программы с вводом данных

```
1 # Запрашиваем ввод у пользователя
2 x = int(input("Введите целое число (0 для выхода): "))
3 # Запускаем цикл, пока пользователь не введет ноль
4 while x != 0:
5     # Положительное или отрицательное?
6     if x > 0:
7         print("Это положительное число.")
8     else:
9         print("Это отрицательное число.")
10    # Запрашиваем очередное значение
11    x = int(input("Введите целое число (0 для выхода): "))
```

Цикл может не выполниться ни разу!

Джентельменский набор для быстрого старта

- ▶ Где обычно применяют цикл `while`
 - задачи, где нам неизвестно точное количество итераций

```
1 while x != 0:
2     x = int(input("Введите целое число (0 для выхода): "))
3     if x > 0:
4         print("Это положительное число.")
5     else:
6         print("Это отрицательное число.")
7
```

Джентельменский набор для быстрого старта

► Бесконечный цикл While

```
1 while True:
2     a = int(input('Введите число: '))
3     if a == 0:
4         break
5     elif a > 200:
6         continue
7     else:
8         print(a)
```

Помогает точнее определять значения переменных по выходу из цикла

#3 | Цикл For

Для быстрого старта

Азбука

▶ range(start, stop, step):

```
1 a = range(10)
2 print(type(a))
3 print(*a)
```

```
block_3 x
/Library/Frameworks/Python.framework
<class 'range'>
0 1 2 3 4 5 6 7 8 9
```

```
1 a = range(2,10)
2 print(*a)
3
```

```
block_3 x
/Library/Frameworks/Python.framework
2 3 4 5 6 7 8 9
```

```
1 a = range(2,10,2)
2 print(*a)
3
```

```
block_3 x
/Library/Frameworks/Python.framework
2 4 6 8
```

```
1 a = range(10,0,-1)
2 print(*a)
3
```

```
block_3 x
/Library/Frameworks/Python.framework
10 9 8 7 6 5 4 3 2 1
```

Азбука

▶ range(start, stop, step):

```
# Запрашиваем у пользователя верхнюю границу
limit = int(input("Введите целое число: "))
# Выводим все числа, кратные трем, вплоть до указанного пользователем значения
print("Все числа, кратные трем, вплоть до", limit, ":")
for i in range(3, limit + 1, 3):
    print(i)
```

Run

block_3 ×

```
/Library/Frameworks/Python.framework/Versions/3.10/bin/python3 /Users/raybin/PyCh
```

```
Введите целое число: 20
```

```
Все числа, кратные трем, вплоть до 20 :
```

```
3
6
9
12
15
18
```

Джентельменский набор для быстрого старта

▶ Цикл For:

- Цикл с параметром
- Тело цикла for будет выполняться по разу для каждого элемента коллекции
- Коллекция:
 - ▶ диапазон целых чисел (range)
 - ▶ буквы в строке
 - ▶ элементы списков
 - ▶ элементы любых итерируемых последовательностей

```
for <переменная> in <коллекция>:  
    <тело цикла>
```

Джентельменский набор для быстрого старта

► Цикл For для итерируемых элементов коллекций:

- Можно использовать как одну переменную, так и несколько

```
s = [(1,2),(2,3),(3,4),(4,5)]  
for x,y in s:  
    print(x,y)
```

block_3 ×

```
/Library/Frameworks/Python.framework/
```

```
1 2
```

```
2 3
```

```
3 4
```

```
4 5
```

Джентельменский набор для быстрого старта

► Тело цикла For:

- Одна конструкция или несколько конструкций
- Перед каждым запуском тела цикла очередной элемент из коллекции копируется в переменную <variable>

```
for x in 'QWERTYUIOPASDFGHJKLZXCVBNM':  
    print(x, end=" -> ")
```

Run

block_3 x

```
/Library/Frameworks/Python.framework/Versions/3.10/bin/python3 /Users/raybin/PycharmProjects/course/block_3.py  
Q -> W -> E -> R -> T -> Y -> U -> I -> O -> P -> A -> S -> D -> F -> G -> H -> J -> K -> L -> Z -> X -> C -> V ->  
B -> N -> M ->
```

- Эта переменная создается для цикла for при его запуске (и только !!!)
- В конце цикла в этой переменной **последний элемент коллекции**

Азбука

▶ Функция enumerate:

- Позволяет получить номер элемента последовательности и его значение

```
# заменим каждый пятый символ предложения, начиная с 0-го, на *
text = "Курс Python без воды. Сегодня изучаем циклы"
for i,x in enumerate(text):
    print(f'{i}-{x}',end=" | ")
Run
block_3 x
/Library/Frameworks/Python.framework/Versions/3.10/bin/python3 /Users/raybin/PycharmProjects/course/block_3.py
0-K | 1-y | 2-p | 3-c | 4- | 5-P | 6-y | 7-t | 8-h | 9-o | 10-n | 11- | 12-6 | 13-e | 14-з | 15- | 16-в | 17-o |
 18-д | 19-ы | 20-. | 21- | 22-С | 23-e | 24-г | 25-o | 26-д | 27-н | 28-я | 29- | 30-и | 31-з | 32-y | 33-ч |
 34-a | 35-e | 36-м | 37- | 38-ц | 39-и | 40-к | 41-л | 42-ы |
Process finished with exit code 0
```

Джентельменский набор для быстрого старта

▶ Оператор else для цикла for:

- Код внутри него выполняется, когда выполнятся все итерации цикла for

```
for i in range(10):  
    print(i, end=" ")  
    if i == 7:  
        break  
else:  
    print('Прошли все итерации цикла')
```

block_3 ×

/Library/Frameworks/Python.framework/Versions/

0 1 2 3 4 5 6 7

```
for i in range(10):  
    print(i, end=" ")  
    # if i == 7:  
    #     break  
else:  
    print('Прошли все итерации цикла')
```

block_3 ×

/Library/Frameworks/Python.framework/Versions/

0 1 2 3 4 5 6 7 8 9 Прошли все итерации цикла

Джентельменский набор для быстрого старта

► Как называть переменные:

- Обычно в цикле for переменные i | j | k (для range) и x (для прочего)
- Хотите сохранить значение i к концу итерации – перезапишите её значение в другую переменную

```
for i in range(5,10):  
    a = i  
    s = ''  
    while a > 0:  
        s = str(a%3) + s  
        a = a//3  
    print(i,s)
```

block_3 ×

/Library/Frameworks/Python.framework/Versions/3.1

5 12

6 20

7 21

8 22

9 100

Джентельменский набор для быстрого старта

▶ Типовые задачи для for:

- Перебор чисел – поиск и сохранение подходящих под критерий
- Перебор строк – поиск в ней подстрок разного вида
- Перебор списков для выявления и сохранения особенных элементов
- Перебор прочих итерируемых объектов (itertools, map, file, т.п.)
- Внутри генераторов (!!!)

```
print(x for x in range(100))
```

```
block_3 ×
```

```
/Library/Frameworks/Python.framework/Versions/3.10  
<generator object <genexpr> at 0x1028b1ee0>
```

#4 | Вложенные циклы

Для быстрого старта

Джентельменский набор для быстрого старта

▶ Вложенный цикл = цикл в цикле:

- В любой тип цикла может быть вложен любой тип цикла
- Уровень вложенности может быть как 1 так и более
- Чем больше вложенность тем дольше выполняется программа
- Генераторы также могут быть вложенными

```
a = [(f'{x} * {y} = {x*y}') for x in range(1,10) for y in range(1,10)]  
print(a)
```

```
Run  
block_3 × [Library/Frameworks/Python.framework/Versions/3.10/bin/python3 /Users/raybin/PycharmProjects/course/block_3.py  
['1 * 1 = 1', '1 * 2 = 2', '1 * 3 = 3', '1 * 4 = 4', '1 * 5 = 5', '1 * 6 = 6', '1 * 7 = 7', '1 * 8 = 8', '1 * 9 =  
9', '2 * 1 = 2', '2 * 2 = 4', '2 * 3 = 6', '2 * 4 = 8', '2 * 5 = 10', '2 * 6 = 12', '2 * 7 = 14', '2 * 8 = 16', '2  
* 9 = 18', '3 * 1 = 3', '3 * 2 = 6', '3 * 3 = 9', '3 * 4 = 12', '3 * 5 = 15', '3 * 6 = 18', '3 * 7 = 21', '3 * 8  
= 24', '3 * 9 = 27', '4 * 1 = 4', '4 * 2 = 8', '4 * 3 = 12', '4 * 4 = 16', '4 * 5 = 20', '4 * 6 = 24', '4 * 7 =  
28', '4 * 8 = 32', '4 * 9 = 36', '5 * 1 = 5', '5 * 2 = 10', '5 * 3 = 15', '5 * 4 = 20', '5 * 5 = 25', '5 * 6 =  
30', '5 * 7 = 35', '5 * 8 = 40', '5 * 9 = 45', '6 * 1 = 6', '6 * 2 = 12', '6 * 3 = 18', '6 * 4 = 24', '6 * 5 =  
30', '6 * 6 = 36', '6 * 7 = 42', '6 * 8 = 48', '6 * 9 = 54', '7 * 1 = 7', '7 * 2 = 14', '7 * 3 = 21', '7 * 4 =  
28', '7 * 5 = 35', '7 * 6 = 42', '7 * 7 = 49', '7 * 8 = 56', '7 * 9 = 63', '8 * 1 = 8', '8 * 2 = 16', '8 * 3 =  
24', '8 * 4 = 32', '8 * 5 = 40', '8 * 6 = 48', '8 * 7 = 56', '8 * 8 = 64', '8 * 9 = 72', '9 * 1 = 9', '9 * 2 =  
18', '9 * 3 = 27', '9 * 4 = 36', '9 * 5 = 45', '9 * 6 = 54', '9 * 7 = 63', '9 * 8 = 72', '9 * 9 = 81']
```

Джентельменский набор для быстрого старта

```
# Запрашиваем у пользователя сообщение
message = input("Введите сообщение (оставьте его пустым для выхода): ")
# Начало цикла, пока сообщение не станет пустым
while message != "":
    # Запрашиваем количество повторений
    n = int(input("Сколько раз повторить сообщение? "))
    # Показываем сообщение заданное количество раз
    for i in range(n):
        print(message)
    # Запрашиваем следующее сообщение
    message = input("Введите сообщение (оставьте его пустым для выхода): ")
```

Run

block_3 ×

```
/Library/Frameworks/Python.framework/Versions/3.10/bin/python3 /Users/raybin/
Введите сообщение (оставьте его пустым для выхода): Блок по циклам
Сколько раз повторить сообщение? 5
Блок по циклам
Блок по циклам
Блок по циклам
Блок по циклам
Блок по циклам
Введите сообщение (оставьте его пустым для выхода):
```

Задача блока 3

▶ НАУЧИТЬСЯ ГРАМОТНО ПИСАТЬ ЦИКЛЫ

```
a = a//10  
print(a)  
a = a//10  
print(a)  
a = a//10  
print(a)  
a = a//10  
print(a)  
a = a//10  
print(a)  
a = a//10  
print(a)
```

