


СВЯЗЬ ЦИКЛОВ В ЯЗЫКЕ Python

 Модуль 1. Занятие 10.



Тема **и цель**

Тема:

Классификация циклов и различие в их применении.

Цель занятия:

Изучение разных видов циклов на языке Python.



Глоссарий

- 1. Цикл** – разновидность управляющей конструкции в высокоуровневых языках программирования, предназначенная для организации многократного исполнения набора инструкций.
- 2. Программа** — данные, которые используются процессором как инструкции по управлению компьютерной системой.

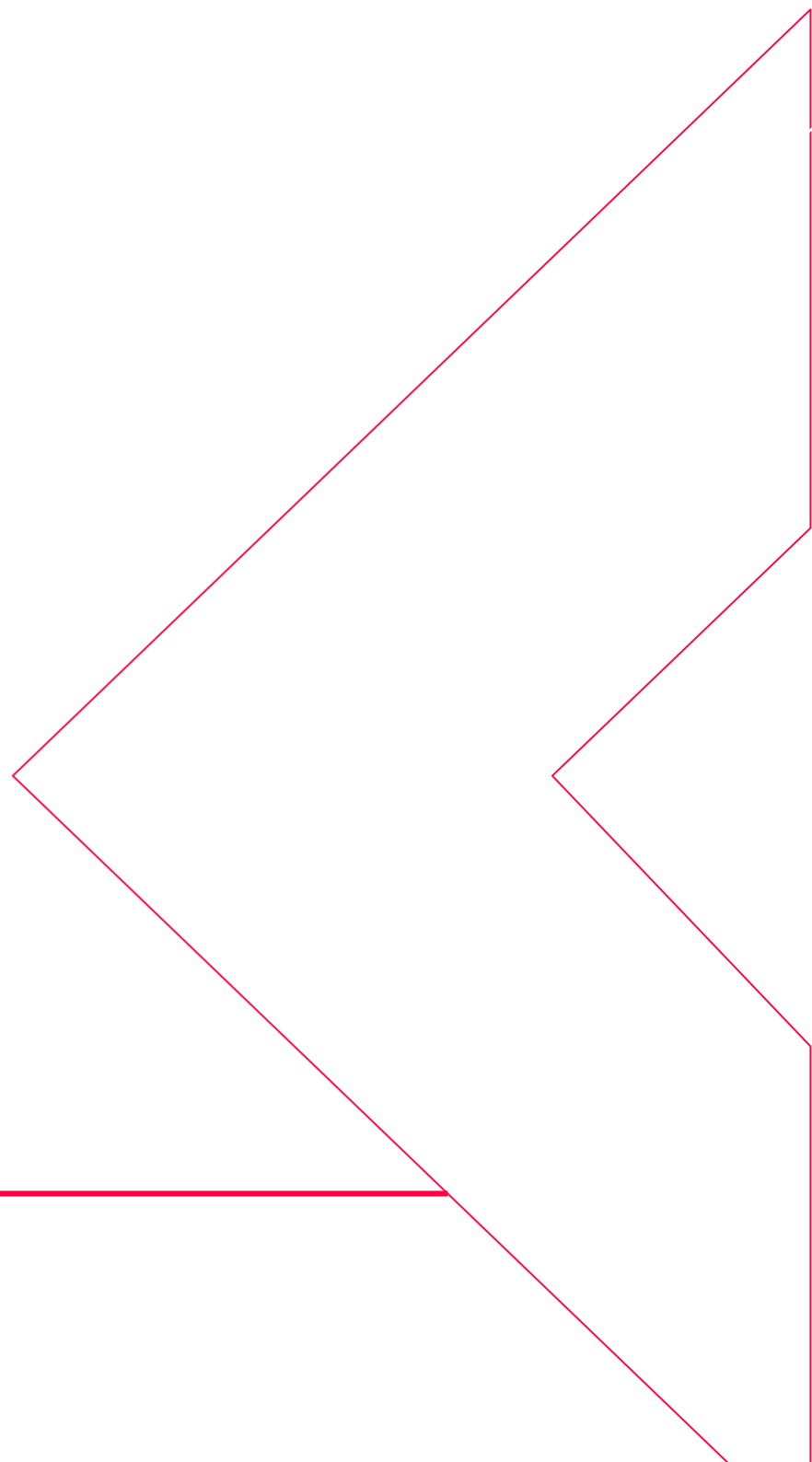


Подтемы

1. Какую роль выполняет цикл **while** в языке Python?
2. Что такое **программа**?
3. Как можно передавать аргументы циклам?
4. Какую роль выполняет функция **max(x)** в языке Python?
5. Какое максимальное количество аргументов можно передать циклу на языке Python за один раз?



Цикл `while` (повторение)



Конструкция цикла *while*



```
1 while условие_выражение:  
2     инструкции
```

Рисунок 1

Пример работы цикла `while`



```
1 number = 1
2
3 while number < 5:
4     print(f"number = {number}")
5     number += 1
6 print("Работа программы завершена")
```

Рисунок 2



Конструкции цикла

```
1 print(f"number = {number}")  
2 number += 1
```

Рисунок 3

Цикл `while` (повторение)



Процесс цикла можно представить следующим образом:

1. Сначала проверяется значение переменной **number** - больше ли оно 5. И поскольку вначале переменная равна 1, то это условие возвращает `True`, и поэтому выполняются инструкции цикла
2. Инструкции цикла выводят на консоль строку `number = 1`. И далее значение переменной `number` увеличивается на единицу - теперь она равна 2. Однократное выполнение блока инструкций цикла называется **итерацией**. То есть таким образом, в цикле выполняется первая **итерация**
3. Снова проверяется условие `number < 5`. Оно по прежнему равно `True`, так как `number = 2`, поэтому выполняются инструкции цикла

Цикл `while` (повторение)



4. Инструкции цикла выводят на консоль строку `number = 2`. И далее значение переменной `number` опять увеличивается на единицу - теперь она равна 3. Таким образом, выполняется вторая итерация
5. Опять проверяется условие `number < 5`. Оно по прежнему равно `True`, так как `number = 3`, поэтому выполняются инструкции цикла
6. Инструкции цикла выводят на консоль строку `number = 3`. И далее значение переменной `number` опять увеличивается на единицу - теперь она равна 4. То есть выполняется третья итерация.

Цикл `while` (повторение)



7. Снова проверяется условие `number < 5`. Оно по прежнему равно `True`, так как `number = 4`, поэтому выполняются инструкции цикла
8. Инструкции цикла выводят на консоль строку `number = 4`. И далее значение переменной `number` опять увеличивается на единицу - теперь она равна 5. То есть выполняется четвертая итерация
9. И вновь проверяется условие `number < 5`. Но теперь оно равно `False`, так как `number = 5`, поэтому выполняется выход из цикла. Все цикл - завершился. Дальше уже выполняются действия, которые определены после цикла. Таким образом, данный цикл произведет четыре прохода или четыре итерации



Консольный вывод при использовании цикла *while*

```
number = 1  
number = 2  
number = 3  
number = 4  
Работа программы завершена
```

Рисунок 4

Пример программы с циклом `while`



```
1 number = 1
2
3 while number < 5:
4     print(f"number = {number}")
5     number += 1
6 else:
7     print(f"number = {number}. Работа цикла завершена")
8 print("Работа программы завершена")
```

Рисунок 5

Консольный вывод при работе с **циклом while**



```
number = 1  
number = 2  
number = 3  
number = 4  
number = 5. Работа цикла завершена  
Работа программы завершена
```

Рисунок 6

Пример программы с блоком `else`

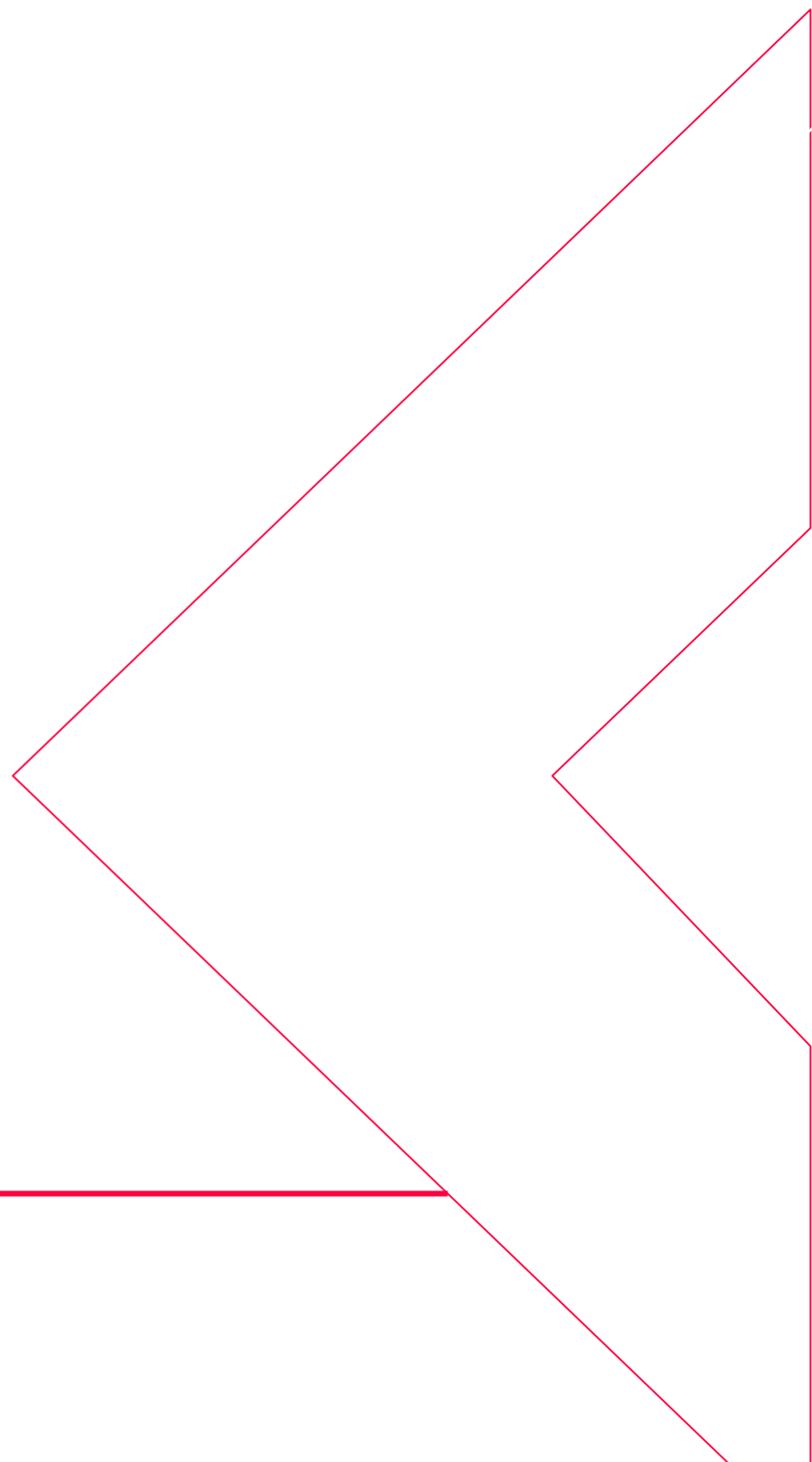


```
1 number = 10
2
3 while number < 5:
4     print(f"number = {number}")
5     number += 1
6 else:
7     print(f"number = {number}. Работа цикла завершена")
8 print("Работа программы завершена")
```

Рисунок 7



Цикл **for**



Конструкция цикла `for`



```
1  for переменная in набор_значений:  
2      инструкции
```

Рисунок 8

Пример программы с циклом `for`



```
1 message = "Hello"  
2  
3 for c in message:  
4     print(c)
```

Рисунок 9

Консольный вывод при работе с **ЦИКЛОМ for**



Рисунок 10

Пример программы с использованием цикла `for` и блока `else`



```
1 message = "Hello"
2 for c in message:
3     print(c)
4 else:
5     print(f"Последний символ: {c}. Цикл завершен");
6 print("Работа программы завершена") # инструкция не имеет отступа, поэтому не относится к else
```

Рисунок 11

Консольный вывод при работе с циклом `for` и блоком `else`

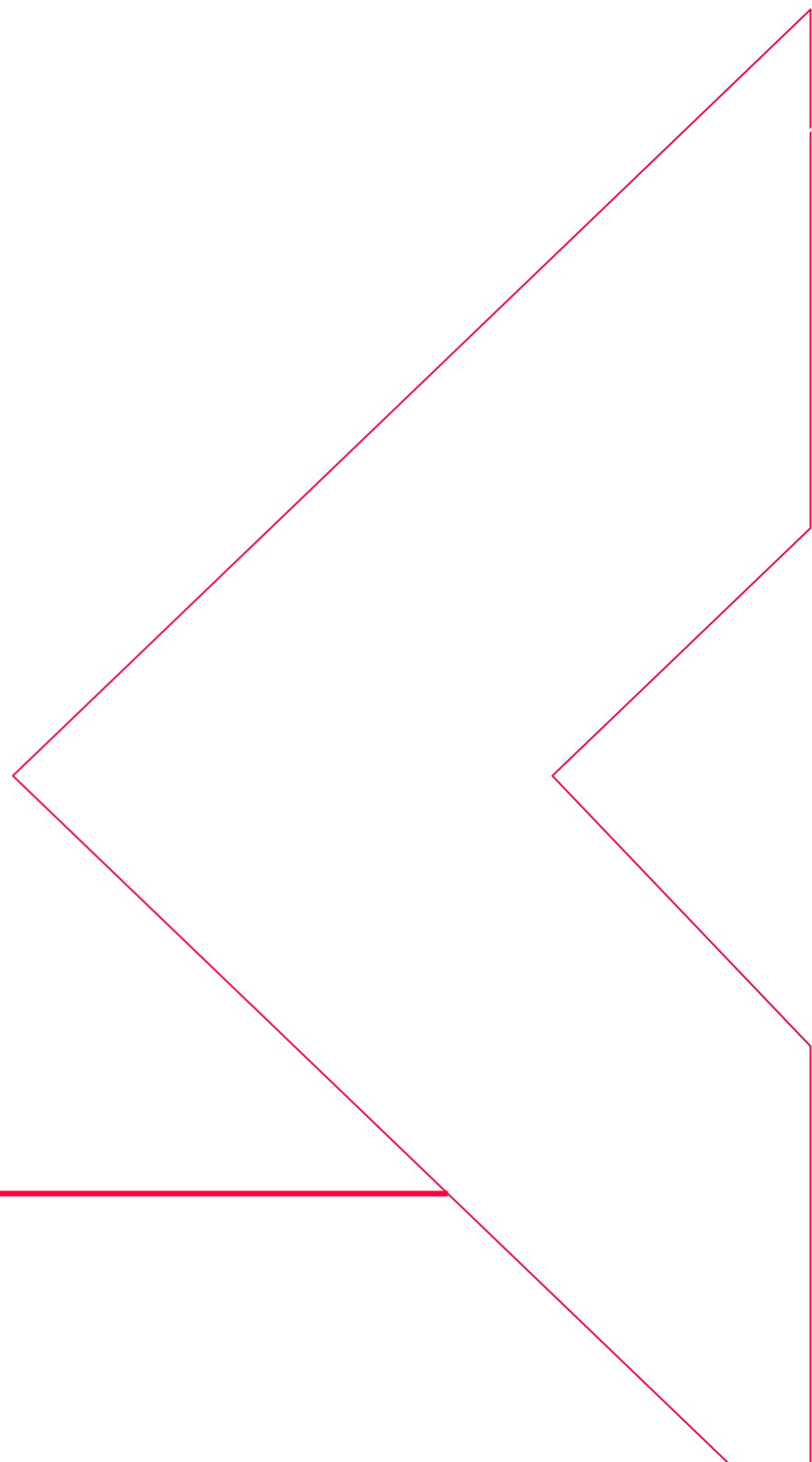


```
Н  
е  
|  
|  
о  
Последний символ: о. Цикл завершен  
Работа программы завершена
```

Рисунок 12



Вложенные циклы (повторение)



Пример программы с вложенными циклами



```
1 i = 1
2 j = 1
3 while i < 10:
4     while j < 10:
5         print(i * j, end="\t")
6         j += 1
7     print("\n")
8     j = 1
9     i += 1
```

Рисунок 13

Консольный вывод при работе с **вложенными циклами**



```
1  2  3  4  5  6  7  8  9
2  4  6  8 10 12 14 16 18
3  6  9 12 15 18 21 24 27
4  8 12 16 20 24 28 32 36
5 10 15 20 25 30 35 40 45
6 12 18 24 30 36 42 48 54
7 14 21 28 35 42 49 56 63
8 16 24 32 40 48 56 64 72
9 18 27 36 45 54 63 72 81
```

Рисунок 14

Пример программы с вложенными циклами



```
1 for c1 in "ab":  
2     for c2 in "ba":  
3         print(f"{c1}{c2}")
```

Рисунок 15

Консольный вывод при работе с **вложенными циклами**



```
ab
```

```
aa
```

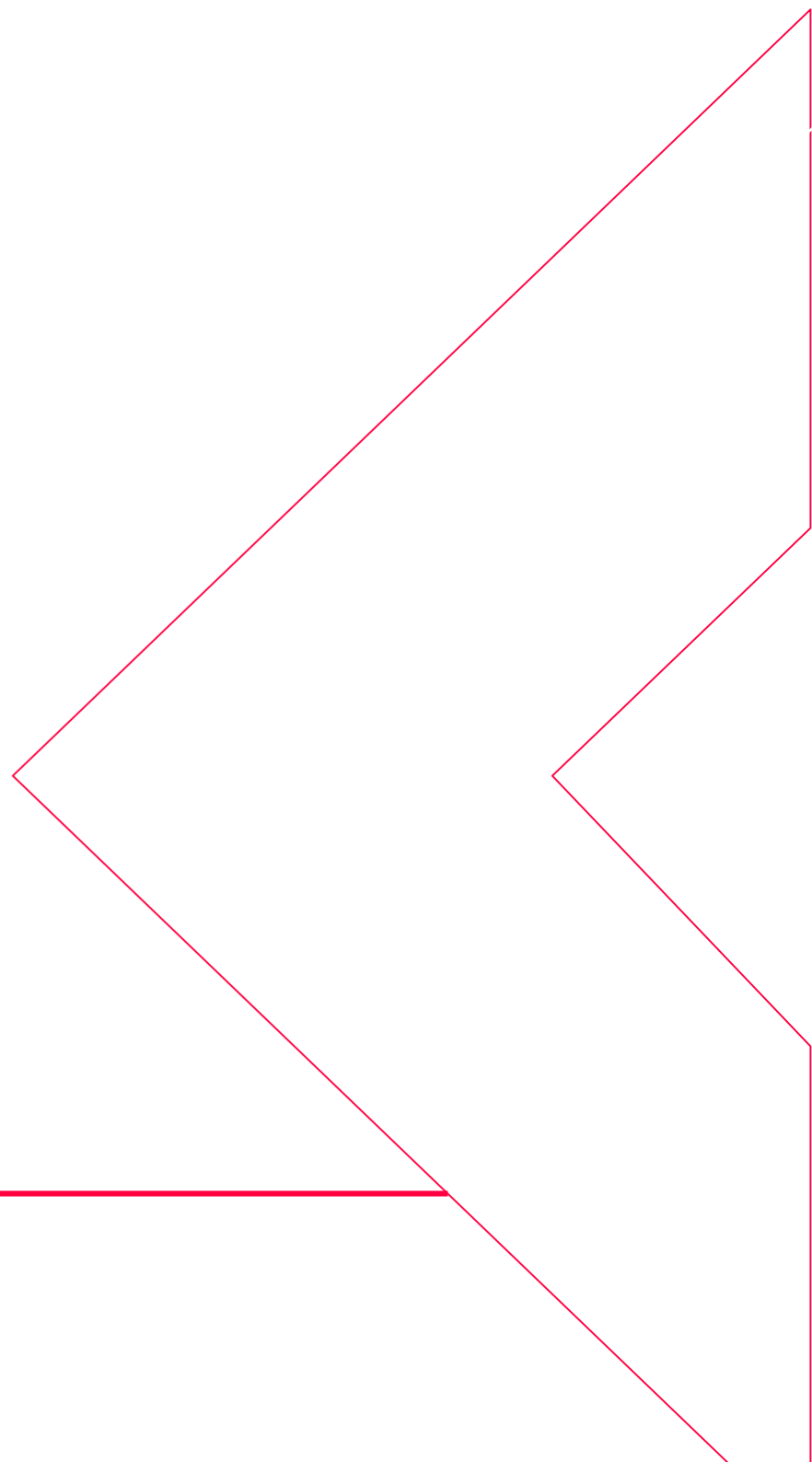
```
bb
```

```
ba
```

Рисунок 16



Выход из цикла. `break` и `continue` (повторение)



Пример программы с использованием оператора `break`



```
1 number = 0
2 while number < 5:
3     number += 1
4     if number == 3 :    # если number = 3, выходим из цикла
5         break
6     print(f"number = {number}")
```

Рисунок 17

Консольный вывод при работе с оператором break



```
number = 1  
number = 2
```

Рисунок 18

Пример программы с использованием оператора `continue`



```
1 number = 0
2 while number < 5:
3     number += 1
4     if number == 3 :    # если number = 3, переходим к новой итерации цикла
5         continue
6     print(f"number = {number}")
```

Рисунок 19

Консольный вывод

с использованием оператора `continue`



```
number = 1
```

```
number = 2
```

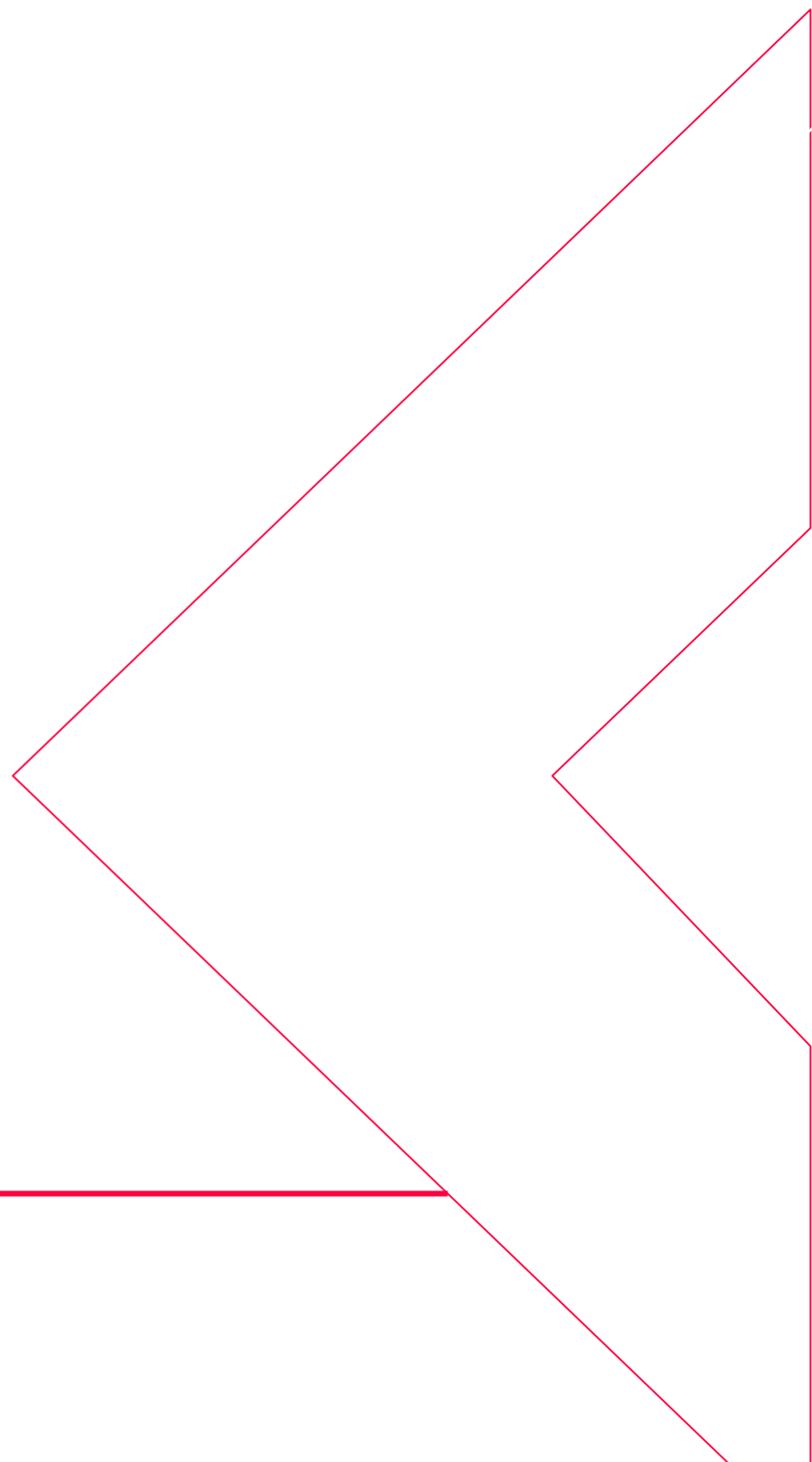
```
number = 4
```

```
number = 5
```

Рисунок 20



Практические задачи





Задача 1

Создать программу с использованием цикла for.
Данный цикл перебирает значения переменной 'q'
от 3 до 7 и выводит эти значения на консоль.



Решение

Напишем код для решения данной практической задачи и посмотрим на вывод:

The screenshot shows an IDE window titled 'pythonProject3' with a file named 'main.py'. The code in the editor is:

```
1  
2 # реализация цикла 'for'  
3 for q in range(3, 8):  
4     print('q = ', q)
```

Below the editor, the 'Run' console shows the output of the code:

```
for q in range(3, 8)  
q = 3  
q = 4  
q = 5  
q = 6  
q = 7
```



Задача 2

Создать программу с использованием цикла `while` и переменной `'y'`, значение которой, изначально, равно 4. Данный цикл выводит значения переменной `'y'` во второй степени до тех пор, пока значение переменной `'y'` не станет равным 8. При каждой итерации значение переменной `'y'` увеличивается на единицу



Решение

Напишем код для решения данной практической задачи и посмотрим на вывод:

```
pythonProject3 > main.py
1
2 # реализация цикла 'while'
3 y = 4
4 while y < 8:
5     print('Значение: ', y**2)
6     y += 1
```

```
Run: main
C:\Users\user\PycharmProjects\pythonProject3\venv
Значение: 16
Значение: 25
Значение: 36
Значение: 49

Process finished with exit code 0
```

PEP 8: W292 no newline at end of file 6:11 CRLF UTF-8 4 spaces Python 3.9 (pythonProject3)



Задача 3

Создать программу с использованием цикла `while` и переменной `'y'`, значение которой, изначально, равно 2. Данный цикл выводит значения переменной `'y'` в третьей степени до тех пор, пока значение переменной `'y'` не станет равным 10. При каждой итерации значение переменной `'y'` увеличивается на 2.



Решение

Напишем код для решения данной практической задачи и посмотрим на вывод:

```
pythonProject3 - main.py  
pythonProject3 > main.py  
1 # реализация циклов 'while'  
2 y = 2  
3 while y < 10:  
4     print('Значение: ', y**3)  
5     y += 2  
6  
while y < 10  
Run: main x  
C:\Users\user\PycharmProjects\pythonProject3\venv\  
Значение: 8  
Значение: 64  
Значение: 216  
Значение: 512  
Process finished with exit code 0
```



Задача 4

Создать программу с использованием цикла `for`. Данный цикл перебирает значения переменной 'a' от 3 до 7 и переменной 'b' от 10 до 13 и выводит соответствующие значения на консоль.



Решение

Напишем код для решения данной практической задачи и посмотрим на вывод:

The screenshot shows an IDE window for a Python project named 'pythonProject3'. The editor displays the following code in 'main.py':

```
1 # реализация цикла 'for'  
2 for a in range(3, 8):  
3     print('a = ', a)  
4 for b in range(10, 14):  
5     print('b = ', b)  
6
```

The Run console at the bottom shows the output of the code:

```
a = 3  
a = 4  
a = 5  
a = 6  
a = 7  
b = 10  
b = 11  
b = 12  
b = 13
```




Вопросы

1. Какую роль выполняет цикл **for** в языке Python?
2. Что такое **поток данных**?
3. Как передавать циклам глобальные переменные?
4. Какую роль выполняет функция **min(x)** в языке Python?
5. Какое минимальное количество аргументов можно передать циклу на языке Python за один раз?