



ЯЗЫК
ПРОГРАММИРОВАНИЯ
PYTHON



Особенности языка Python

- Скриптовый язык
- Поддержка самых различных парадигм программирования
- Интерпретация программ
- Портативность и платформонезависимость
- Автоматическое управление памяти
- Динамическая типизация

Интерпретация программы Python



Ресурсы для знакомства с языком Python

- <http://pythontutor.ru/> - Интерактивный учебник языка Питон
- <https://pythonworld.ru/> - Python 3 для начинающих
- <https://metanit.com/python/> - Руководство по языку программирования Python
- <https://gto76.github.io/python-cheatsheet/> - Comprehensive Python Cheatsheet
- <http://stepik.org>

Введение в написание программ

```
print(2 + 3)
```

```
print("Hello")
```

#Неправильно

```
print(2 + 3)
```

```
    print("Hello")
```

Введение в написание программ

#Тут отступы необходимы

```
if 1 < 2:
```

```
    print("Hello")
```

#Регистрозависимость

```
Print("Hello World") #Команда не сработает
```

Основные функции

#функция print()

```
print("Hello Python")
```

```
print("Full name:", "Tom", "Smith")
```

#вывод через пробел: «Full name: Tom Smith»

Основные функции

```
#параметр sep
```

```
print('a', 'b', 'c', sep='*')
```

```
print('d', 'e', 'f', sep='**')
```

```
#a*b*c
```

```
#d**e**f
```


Основные функции

```
#параметр end
```

```
print('a', 'b', 'c', end='@')
```

```
print('d', 'e', 'f', end='@@')
```

```
#a b c@d e f@@
```

Основные функции

```
print('a', 'b', 'c', sep='*', end='finish')
```

```
print('d', 'e', 'f', sep='**', end='^__^')
```

```
print('g', 'h', 'i', sep='+', end='%')
```

```
print('j', 'k', 'l', sep='-', end='#')
```

```
print('m', 'n', 'o', sep='/', end='!')
```

Основные функции

```
#функция input()
```

```
name = input("Enter name: ")
```

```
print('Hello', name)
```

Основные функции

#множественное присваивание

```
name, surname = 'Timur', 'Guev'
```

```
print('Имя:', name, 'Фамилия:', surname)
```

```
name, surname = input(), input()
```

```
print('Имя:', name, 'Фамилия:', surname)
```

Некоторые рекомендации PEP 8

#Правильно:

```
print('Follow PEP8!')
```

#Неправильно:

```
print ('Follow PEP8!')
```

Некоторые рекомендации PEP 8

#Правильно:

```
print('PEP8', 'Rocks!')
```

#Неправильно:

```
print('PEP8', 'Rocks!')
```

Некоторые рекомендации PEP 8

#Правильно:

```
print('My name', 'is', 'Python', sep='**',  
end='+')
```

#Неправильно:

```
print('My name', 'is', 'Python', sep =  
'**', end = '+')
```

Типы данных

- `boolean` - логическое значение `True` или `False`
- `int` - представляет целое число, например, 1, 4, 8, 50.
- `float` - представляет число с плавающей точкой, например, 1.2 или 34.76
- `complex` - комплексные числа
- `str` - строки, например "hello". В Python 3.x строки представляют набор символов в кодировке Unicode
- `bytes` - последовательность чисел в диапазоне 0-255
- `byte array` - массив байтов, аналогичен `bytes` с тем отличием, что может изменяться
- `list` - список
- `tuple` - кортеж
- `set` - неупорядоченная коллекция уникальных объектов
- `frozen set` - то же самое, что и `set`, только не может изменяться (`immutable`)
- `dict` - словарь, где каждый элемент имеет ключ и значение

Типы данных

```
x = 3.9e3  
print(x)    # 3900.0
```

```
x = 3.9e-3  
print(x)    # 0.0039
```

```
x = "12tomsmith438"    # тип str  
print(x)    # "12tomsmith438"
```

Типы данных

```
#функция type()
```

```
user_id = "12tomsmith438"
```

```
print(type(user_id)) # <class 'str'>
```

```
user_id = 234
```

```
print(type(user_id)) # <class 'int'>
```

Арифметические операции

```
print(6 + 2)    # 8
print(6 - 2)    # 4
print(6 * 2)    # 12
print(7 / 2)    # 3.5
print(7 // 2)   # 3
print(7 ** 2)   # 49
print(7 % 2)    # 1
```

Арифметические операции (приоритеты операций)

Операции	Направление
**	Справа налево
* / // %	Слева направо
+ -	Слева направо

```
number = 3 + 4 * 5 ** 2 + 7  
print(number) # 110
```

Арифметические операции

■ +=

■ -=

■ *=

■ /=

■ // =

■ ** =

■ %=

Модуль math

```
import math
```

```
num1 = math.sqrt(2)          # вычисление  
корня квадратного из двух
```

```
num2 = math.ceil(3.8)       # округление  
числа вверх
```

```
num3 = math.floor(3.8)      # округление  
числа вниз
```

Модуль math

```
from math import *
```

```
num1 = sqrt(2)          # вычисление корня  
квадратного из двух
```

```
num2 = ceil(3.8)       # округление числа  
вверх
```

```
num3 = floor(3.8)      # округление числа  
вниз
```

Модуль math

```
from math import sqrt, ceil
```

```
num1 = sqrt(2)          # вычисление корня  
квадратного из двух
```

```
num2 = ceil(3.8)       # округление числа  
вверх
```

```
num3 = floor(3.8)      # приведет к ошибке,  
так как функция floor не подключена
```


Операции со строками

```
name = "Tom"
surname = 'Smith'
age = 33
fullname = name + " " + surname
print(fullname)    # Tom Smith
info = "Name: " + name + " Age: " + str(age)
print(info)       # Name: Tom Age: 33
```

Операции со строками

```
str1 = "Tom"
```

```
str2 = "tom"
```

```
print(str1 == str2)    # False - строки не  
равны
```

```
print(str1.lower() == str2.lower())    # True
```

```
#lower(), upper()
```

Операции со строками

```
s1 = 'abcdef'
```

```
# считаем длину строки из переменной s1
```

```
length1 = len(s1)
```

```
s = 'Hi' * 4
```

```
print(s)    #HiHiHiHi
```

Функции преобразования

```
first_number = "2"  
second_number = 3  
third_number = int(first_number) +  
second_number  
print(third_number) # 5
```

```
#функции int(), float(), str()
```

Функции преобразования

```
first_number = 2.0001
```

```
second_number = 0.1
```

```
third_number = first_number +  
second_number
```

```
print(round(third_number, 4)) # 2.1001
```

Условные выражения

- ==
- !=
- > (больше чем)
- < (меньше чем)
- >= (больше или равно)
- <= (меньше или равно)

Условные выражения

```
a = 5
```

```
b = 6
```

```
result = 5 == 6 # False
```

```
print(result) # False - 5 не равно 6
```

```
print(a != b) # True
```

```
print(a > b) # False - 5 меньше 6
```

```
print(a < b) # True
```

Условные выражения

```
bool1 = True
```

```
bool2 = False
```

```
print(bool1 == bool2)    # False - bool1 не  
равно bool2
```


Логические операции

```
age = 22
```

```
weight = 58
```

```
isMarried = False
```

```
result = age > 21 and weight == 58
```

```
print(result) # True
```

```
result = age > 21 or isMarried
```

```
print(result) # True, так как выражение age  
> 21 равно True
```

```
print(not age > 21) # False
```

Условная конструкция if

```
if логическое_выражение:
```

```
    инструкции
```

```
[elif логическое выражение:
```

```
    инструкции]
```

```
[else:
```

```
    инструкции]
```

Условная конструкция if

```
age = 22
if age > 21:
    print("Доступ разрешен")
print("Завершение работы")
```

Условная конструкция if

```
age = 22
```

```
if age > 21:
```

```
    print("Доступ разрешен")
```

```
    print("Завершение работы")
```

Условная конструкция if

```
age = 18
if age > 21:
    print("Доступ разрешен")
else:
    print("Доступ запрещен")
```

Условная конструкция if

```
age = 18
if age >= 21:
    print("Доступ разрешен")
elif age >= 18:
    print("Доступ частично разрешен")
else:
    print("Доступ запрещен")
```

Условная конструкция if

```
age = 18
if age >= 18:
    print("Больше 17")
    if age > 21:
        print("Больше 21")
    else:
        print("От 18 до 21")
```

Условная конструкция if

```
age = 18
if age >= 18:
    print("Больше 17")
if age > 21:
    print("Больше 21")
else:
    print("От 18 до 21")
```


Тернарный оператор

```
condition_if_true if condition else condition_if_false
```

#Пример

```
is_nice = True
```

```
state = "nice" if is_nice else "not nice"
```

Циклы While

```
while условие_выражение:  
    инструкции
```

Циклы While

```
choice = "y"
```

```
while choice.lower() == "y":
```

```
    print("Привет")
```

```
    choice = input("Для продолжения нажмите  
Y, а для выхода любую другую клавишу: ")
```

```
print("Работа программы завешена")
```

Циклы While

#! Программа по вычислению факториала

```
number = int(input("Введите число: "))
i = 1
factorial = 1
while i <= number:
    factorial *= i
    i += 1
print("Факториал числа", number, "равен",
factorial)
```

Циклы For

```
for int_var in функция_range:  
    инструкции
```

Циклы For

#! Программа по вычислению факториала

```
number = int(input("Введите число: "))
factorial = 1
for i in range(1, number+1):
    factorial *= i
print("Факториал числа", number, "равен",
      factorial)
```

Функция range()

`range(5)` # 0, 1, 2, 3, 4

`range(1, 5)` # 1, 2, 3, 4

`range(2, 10, 2)` # 2, 4, 6, 8

`range(5, 0, -1)` # 5, 4, 3, 2, 1

Блок else в циклах

```
for item in container:
    if search_something(item):
        # Нашли!
        process(item)
        break
else:
    # Ничего не найдено...
    not_found_in_container()
```


Блок else в циклах

```
for n in range(2, 10):  
    for x in range(2, n):  
        if n % x == 0:  
            print( n, 'equals', x, '*', n/x)  
            break  
    else:  
        # Цикл не нашел целочисленного делителя для n  
        print(n, 'is a prime number')
```