

Введение в Python

Игорь Рязанцев

Лекция 01

2021г.

План мини-курса

- **Лекция 01. Введение в Python**

- Переменные в Python
- Кортежи
- Списки
- Циклы

- **Лекция 02. Функции и объекты** [Открыть](#)

- Функции
- Объектно-ориентированное программирование

- **Лекция 03. Файлы и ввод-вывод в Python** [Открыть](#)

- Понятия файлов (текстовые и бинарные)
- Ввод-вывод данных

- **Лекция 04. Построение графиков** [Открыть](#)

- Вывод графиков функций
- Математическая библиотека numpy

Что есть Python?

Python – высокоуровневый язык программирования общего назначения с динамической строгой типизацией и автоматическим управлением памятью



Зачем изучать Python?

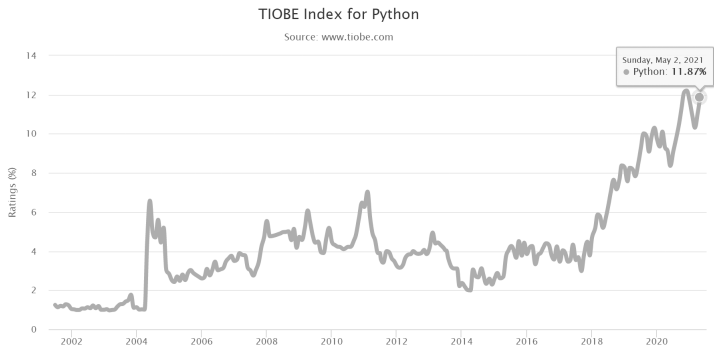
Зачем изучать Python, если вы не программист?

- Python – легкий язык программирования;
- Научит вас думать;
- Сделает вас более самостоятельным;
- Знание Python облегчит коммуникацию с разработчиками;
- Поможет визуализировать данные;
- Позволит автоматизировать повторяющиеся задачи;
- Рынок вас ждет...

TIOBE Index for May 2021

May 2021	May 2020	Change	Programming Language	Ratings	Change
1	1		C	13.38%	-3.68%
2	3	⬆️	Python	11.87%	+2.75%
3	2	⬇️	Java	11.74%	-4.54%
4	4		C++	7.81%	+1.69%
5	5		C#	4.41%	+0.12%
6	6		Visual Basic	4.02%	-0.16%
7	7		JavaScript	2.45%	-0.23%
8	14	⬆️	Assembly language	2.43%	+1.31%
9	8	⬇️	PHP	1.86%	-0.63%
10	9	⬇️	SQL	1.71%	-0.38%
11	15	⬆️	Ruby	1.50%	+0.48%
12	17	⬆️	Classic Visual Basic	1.41%	+0.53%
13	10	⬇️	R	1.38%	-0.46%
14	38	⬆️	Groovy	1.25%	+0.96%
15	13	⬇️	MATLAB	1.23%	+0.06%
16	12	⬇️	Go	1.22%	-0.05%

TIOBE Index for May 2021





Python начал разрабатывать в конце 80-х годов **Гвидо ван Россумом**. Python назван в честь телешоу «Летающий цирк Монти Пайтона»

Даты выпуска основных версий:

- Python 1.0 — январь 1994 года
- Python 2.0 — 16 октября 2000 года
- Python 3.0 — 3 декабря 2008 года

Python 2.x и Python 3.x не совместимы!

Что можно разрабатывать на Python?

- Web-сайты;
- Скрипты, утилиты, т.к. Python – интерпретируемый язык программирования;
- Программы с графическим интерфейсом;
- Arduino, Raspberry Pi, Интернет вещей и умный дом;
- Big-data;
- Нейронные сети;
- и т.п.

Плюсы Python:

- Легкий для обучения;
- Легко читаемый;
- Широкая стандартная библиотека;
- Наличие интерактивного режима;
- Портативность;
- Расширяемость;
- Работа с базами данных;
- Создание GUI;
- Объектно-ориентированный язык программирования.

Недостатки Python¹:

- Низкая скорость работы;
- Интерпретируемый язык программирования.

¹в сравнении с кодом на компилируемых языках, т.к. C/C++

Сайты написанные на Python:

- YouTube
- Instagram
- Google Search
- DropBox
- Spotify
- Netflix
- и многие другие ...

Установка Python

Welcome to Python.org

https://www.python.org

Python PSF Docs PyPI Jobs Community

python™

Donate Search GO Socialize

About Downloads Documentation Community Success Stories News Events

```
# Python 3: List comprehensions
>>> fruits = ['Banana', 'Apple', 'Lime']
>>> loud_fruits = [fruit.upper() for fruit in fruits]
>>> print(loud_fruits)
['BANANA', 'APPLE', 'LIME']

# List and the enumerate function
>>> list(enumerate(fruits))
[(0, 'Banana'), (1, 'Apple'), (2, 'Lime')]
```

Compound Data Types

Lists (known as arrays in other languages) are one of the compound data types that Python understands. Lists can be indexed, sliced and manipulated with other built-in functions. [More about lists in Python 3](#)

1 2 3 4 5

Python is a programming language that lets you work quickly and integrate systems more effectively. >>> [Learn More](#)

Get Started

Whether you're new to programming or an experienced developer, it's easy to learn and use Python.

Download

Python source code and installers are available for download for all versions!

Latest: Python 3.9.7

Docs

Documentation for Python's standard library, along with tutorials and guides, are available online.

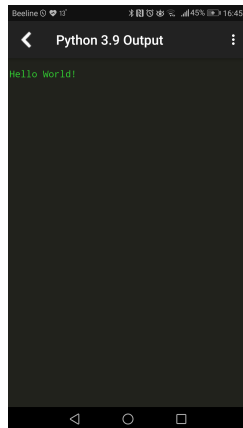
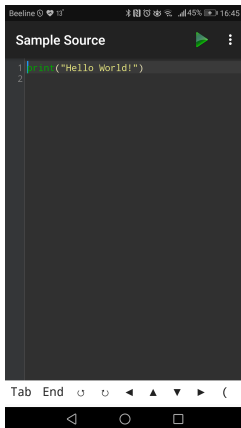
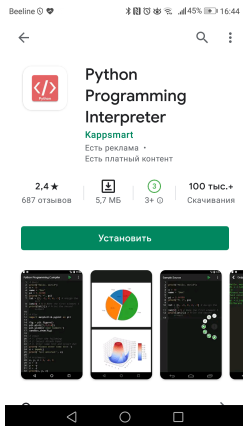
Jobs

Looking for work or have a Python related position that you're trying to hire for? Our **relaunched community-run job**

IDE (интегрированная среда разработки):

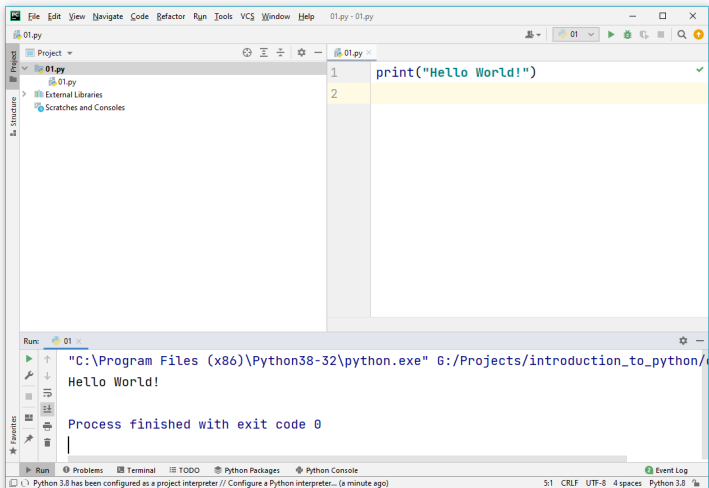
- VSCode [Открыть](#)
- PyCharm [Открыть](#)
- Google Search
- Блокнот и другие

Python Programming Interpreter



Открыть

PyCharm



Загрузить

Оглавление

- 1 Hello World!
- 2 Переменные в Python
- 3 Кортежи
- 4 Списки
- 5 Циклы
- 6 Оператор if

Hello World!

Первая программа, функция print

```
print("Hello World!")
```

или

```
print('Hello World!')
```

Вывод значений переменной, вариант 1

```
var = "world"  
print("Hello ,_! " % var)
```

```
var = None
```

s – для строковых переменных

d – для целых чисел

f – для чисел с плавающей запятой

Форматирование str.format

Вариант 2 (str.format)

```
var = "world"  
print("Hello , {0}!".format(var))
```

вывод нескольких значений переменных

```
text = "world"  
year = 2021  
print("Hello , {0} {1}!".format(text , year))  
  
print("Hello , {0} {year}!".format(text , year=2021))
```

Форматирование str.format

```
text = "world"  
year = 2021  
print("Hello , {0} {1}!".format(text , year))  
  
print("Hello , {0} {year}!".format(text , year=2021))
```

Кортежи tuple()

- # Кортежи – упорядоченная последовательность
- # произвольных объектов, типы которых могут
- # различаться

```
var = ("world", 2021, )  
print("Hello ,_{0}_{1}!".format(var[0], var[1]))
```

Списки list()

Список – это тоже упорядоченная последовательность
произвольных объектов, типы которых могут
различаться

```
days = [ 'Monday', 'Tuesday', 'Wednesday',  
         'Thursday', 'Friday', 'Saturday',  
         'Sunday', ]
```

```
print("Today is {}".format(days[0]))
```

Цикл for

В Python вложенные инструкции объединяются в блоки по величине отступов. Отступ может быть любым, главное, чтобы в пределах одного вложенного блока отступ был одинаков.²

```
days = [ 'Monday', 'Tuesday', 'Wednesday',  
         'Thursday', 'Friday', 'Saturday',  
         'Sunday', ]
```

```
for day in days:  
    print("Today is ", end=' ')  
    print("{0}".format(day))
```

²используйте 4 пробела

Цикл for. Функция range()

Функция range() упрощает построение числовых последовательностей.

```
for value in range(1,5):  
    print(value)
```

Результат:

```
1  
2  
3  
4
```


Операции со списком

`append()` – добавить элемент в конец

```
digits = [1, 2, 3, 4,]  
digits.append(5)  
for value in digits:  
    print(value)
```

Результат:

1
2
3
4
5

Операции со списком

`# insert(позиция, элемент)` – добавить элемент в указанную позицию

```
digits = [1, 2, 3, 4,]
digits.insert(0, 0)
for value in digits:
    print(value)
```

Результат:

```
0
1
2
3
4
```

Операции со списком

remove(элемент) – удаление элемента по значению
pop(позиция) – удаление элемента по индексу³

```
digits = [1, 2, 3, 4,]  
digits.remove(2)  
digits.pop(2)  
for value in digits:  
    print(value)
```

Результат:

1
3

³Значение индекса в списке начинается с 0

Операции со списком

```
# sort() – сортировка элементов списка  
# sort(reverse=True) – сортировка в обратном порядке
```

```
digits = [2, 1, 3, 4,]  
digits.sort()  
for value in digits:  
    print(value)
```

Результат:

```
1  
2  
3  
4
```

Тестовое задание

Необходимо вывести имена в алфавитном порядке

```
names = [(1, 'Sam'),  
          (2, 'Calvin'),  
          (3, 'Henry'),]
```

Результат:

Calvin

Henry

Sam

Тестовое задание

Вариант 1

```
names = [(1, 'Sam'), (2, 'Calvin'), (3, 'Henry'),]
names_new = []
for name in names:
    names_new.append(name[1])

names_new.sort()
for name in names_new:
    print(name)
```

Результат:

Calvin

Henry

Sam

Тестовое задание

Вариант 2.

```
names = [(1, 'Sam'), (2, 'Calvin'), (3, 'Henry'),]
```

```
def custom_key(people):  
    return people[1]
```

```
names.sort(key=custom_key)  
for name in names:  
    print(name[1])
```

Результат:

Calvin

Henry

Sam

Оператор if

Оператор условного перехода

```
cars = ['audi', 'bmw', 'toyota']  
for car in cars:  
    if car == 'bmw':  
        print(car.upper())  
    else:  
        print(car.title())
```

Результат:

Audi

BMW

Toyota

Оператор if

В `if` центральное место занимает выражение, результатом которого является логическая истина (**True**) или логическая ложь (**False**); это выражение называется *условием*.

Общая форма записи:

```
if condition_1:
    action
elif condition_2:
    action
else:
    action
```

Оператор if

==	равно
>	больше
>=	больше или равно
<=	меньше или равно
<	меньше

```
power = 5
if power == 5:
    print('power_==_5')
elif power < 5:
    print('power_<_5')
else:
    print('power_>_5')
```

Необходимо вывести наименование светильника с наибольшим световым потоком и наименьшей потребляемой мощностью

```
leds = [( 'LED1' , 40 , 6000) ,  
        ( 'LED2' , 60 , 9000) ,  
        ( 'LED3' , 90 , 12000) ,  
        ( 'LED4' , 80 , 12000) ,]
```

[1] Презентация [Лекции 01-04]

https://github.com/IRyazantsev/mpei_python_mini-course_2021/tree/main/bin

[2] Python 3. Самое необходимое | Дронов В.А., Прохоренок Н.А.

[3] Изучаем Python. Том 1, 2 | Лутц Марк

[4] Python 3 и PyQt 5. Разработка приложений | Прохоренок Н.А., Дронов В.А.

[5] Django 3.0. Практика создания веб-сайтов на Python | Дронов В. А.

[6] Разработка веб-приложений с использованием Flask | Гринберг Мигель

Вопросы

