



Лекции. Практические занятия

Солдатов Е.Ю.

2024 г.

How to learn programming



In one day

ЧТО ТАКОЕ PYTHON?

Python – это интерпретируемый, объектно-ориентированный язык программирования высокого уровня с динамической типизацией, автоматическим управлением памятью и удобными высокоуровневыми структурами данных...

Поддерживает классы, модули, обработку исключений, а также многопоточные вычисления.

Python обладает простым и выразительным синтаксисом. Язык поддерживает несколько парадигм программирования: *структурное, объектно-ориентированное, функциональное и аспектно-ориентированное.*

```
31     self.file = None
32     self.fingerprints = self()
33     self.logdupes = True
34     self.debug = debug
35     self.logger = logging.getLogger(__name__)
36     if path:
37         self.file = open(os.path.join(path, "requests.log"),
38                         "a")
39         self.file.seek(0)
40         self.fingerprints.update({k: v} for k, v in self.fingerprints.items())
41
42     @classmethod
43     def from_settings(cls, settings):
44         debug = settings.getbool("REQUESTS_LOGGING_DEBUG")
45         return cls(job_dir(settings), debug)
46
47     def request_seen(self, request):
48         fp = self.request_fingerprint(request)
49         if fp in self.fingerprints:
50             return True
51         self.fingerprints.add(fp)
52         if self.file:
53             self.file.write(fp + os.linesep)
54
55     def request_fingerprint(self, request):
56         return request_fingerprint(request)
```

ИСТОРИЯ

- История языка началась в конце 1980 годов.
- Создатель: **Гвидо Ван Россум (Guido van Rossum)** из Голландии
- Язык получил название «Python» не в честь вида змей. Во времена разработки «Python» Гвидо любил смотреть комедийное шоу «Воздушный цирк Монти Пайтона», поэтому и назвал своей проект в честь Монти Пайтона.



- В 1991 году стали появляться первые средства ООП разработки.
- Python полностью бесплатен и может быть свободно использован для коммерческих задач. Много библиотек/модулей доступно для разработчиков.
- Существуют 2 ветки языка - версии 2 и версии 3:
 - Python 2.7 — релиз состоялся 3 июля 2010 года
 - Python 3.12 — релиз состоялся 2 октября 2023 года

С недавнего времени поддерживается только python3, но отличия синтаксиса от python2 – минимальные.



ГДЕ ИСПОЛЬЗУЕТСЯ?

- Компания **Google** использует Python в своей поисковой системе.
- Такие компании, как **Intel, Cisco, Hewlett-Packard, Seagate, Qualcomm и IBM**, используют Python для тестирования аппаратного обеспечения.
- Служба коллективного использования видеоматериалов **YouTube** в значительной степени реализована на Python.
- **NSA** использует Python для шифрования и анализа разведданных.
- Компании **JPMorgan Chase, UBS, Getco и Citadel** применяют Python для прогнозирования финансового рынка.
- Популярная программа **BitTorrent** для обмена файлами в пиринговых сетях написана на языке Python.
- Популярный веб-фреймворк **App Engine** от компании Google использует Python в качестве прикладного языка программирования.
- **NASA, Los Alamos, JPL, Fermilab и CERN** используют Python для научных вычислений.

ПОЛЕЗНЫЕ ССЫЛКИ

Сайт языка Пайтон: <https://www.python.org/>

Скачать любую версию можно тут:

<https://www.python.org/downloads/>

Есть версии для большинства ОС: Windows, UNIX, MacOS и т.д.

Документация: <https://www.python.org/doc/>

Онлайн приложение “notebook”, позволяющее интерактивно писать и интерпретировать код Python: <https://jupyter.org/>

Репозиторий модулей для Python: <https://pypi.org/>

Рейтинг сторонних модулей для Python: <https://pythonwheels.com/>

Рекомендуемый редактор: <https://www.sublimetext.com/3>



ИНСТАЛЛЯЦИЯ И ПЕРВАЯ ПРОГРАММА

- Скачиваем последнюю версию языка: <https://www.python.org/>
Сейчас это 3.12.3.
- Устанавливаем на вашу ОС.

После установки `python` запускается в командной строке.

Запуск кода:

```
python code.py
```

где `code.py` – пользовательский код, написанный в редакторе.

Расширение `.py` говорит о том, что внутри программа, написанная на языке `python`.

Напишем программу

```
print("Hello world!")
```

Сохраним в файл и запустим:

‘ = ‘

```
d:\python>python code.py
Hello world!

d:\python>_
```

Либо запустим в интерактивном режиме:

```
C:\Users\Asus>python
Python 3.8.2 (tags/v3.8.2:7b3ab59, Feb 25 2020, 22:45:29) [MSC v.1916 32 bit (Intel)] on win32
Type "help", "copyright", "credits" or "license" for more information.
>>> _
```

Чтобы выйти: `exit()`

БЛОКНОТ

- Во время этого курса мы будем пользоваться и работой напрямую и более удобно – через notebook

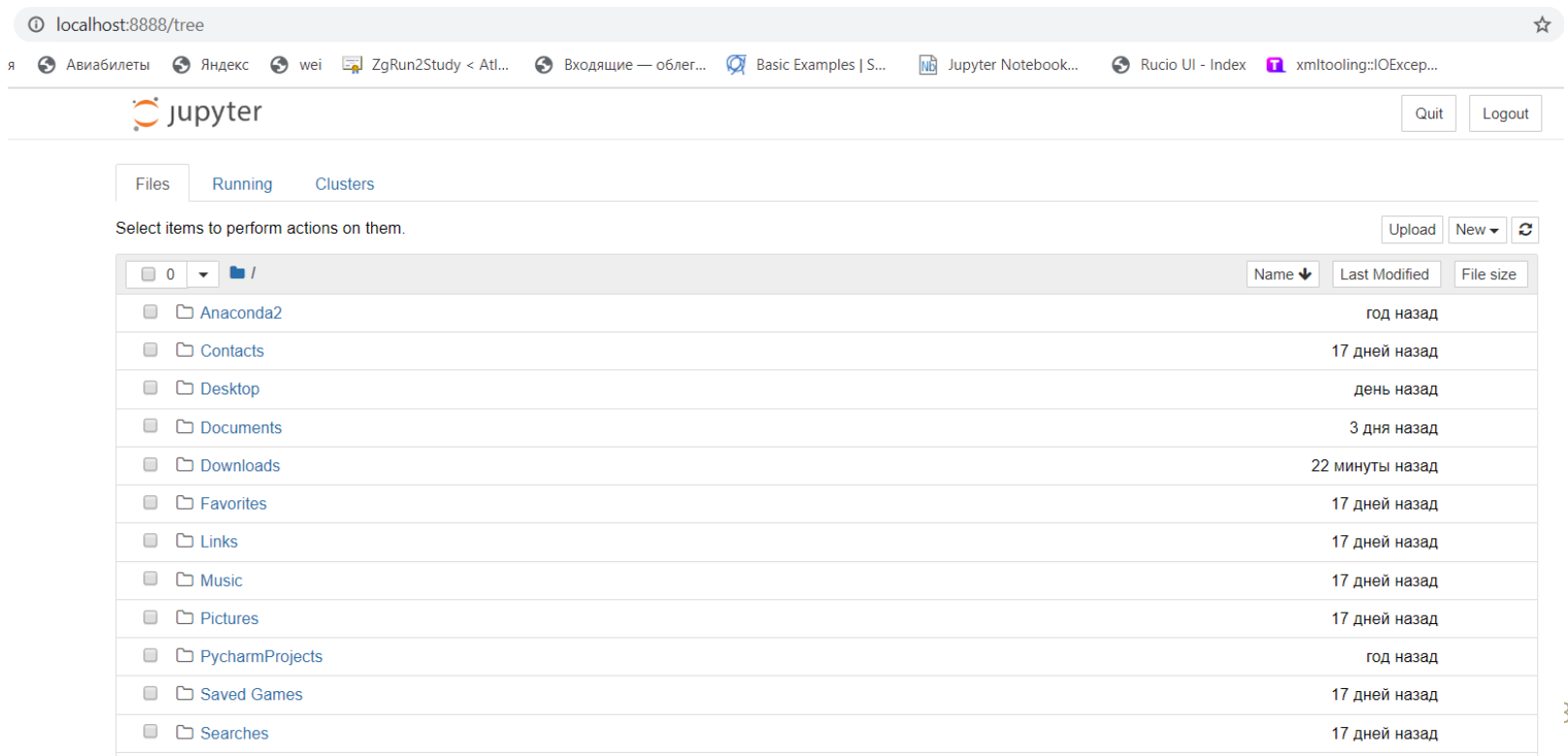
Что такое «**notebook**» (блокнот)? Блокнот объединяет код и его вывод в единый документ, который объединяет визуализацию, текст, математические уравнения и другие мультимедиа...

- Ставим Jupyter Notebook с помощью командной строки:

```
pip install notebook
```

- Запускаем notebook в командной строке:

```
jupyter notebook
```

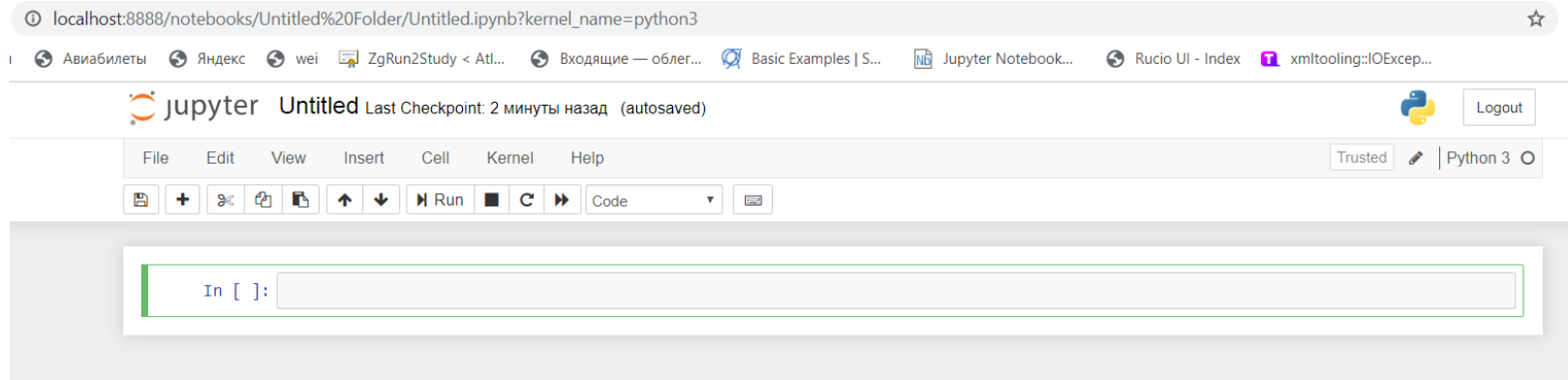


The screenshot shows the Jupyter Notebook web interface. The browser address bar displays `localhost:8888/tree`. The page header includes the Jupyter logo and navigation buttons for "Quit" and "Logout". Below the header, there are tabs for "Files", "Running", and "Clusters". A message states "Select items to perform actions on them." with "Upload", "New", and "Refresh" buttons. The main content area is a file browser showing a directory tree with columns for "Name", "Last Modified", and "File size".

Name	Last Modified	File size
0		
/		
Anaconda2	год назад	
Contacts	17 дней назад	
Desktop	день назад	
Documents	3 дня назад	
Downloads	22 минуты назад	
Favorites	17 дней назад	
Links	17 дней назад	
Music	17 дней назад	
Pictures	17 дней назад	
PycharmProjects	год назад	
Saved Games	17 дней назад	
Searches	17 дней назад	

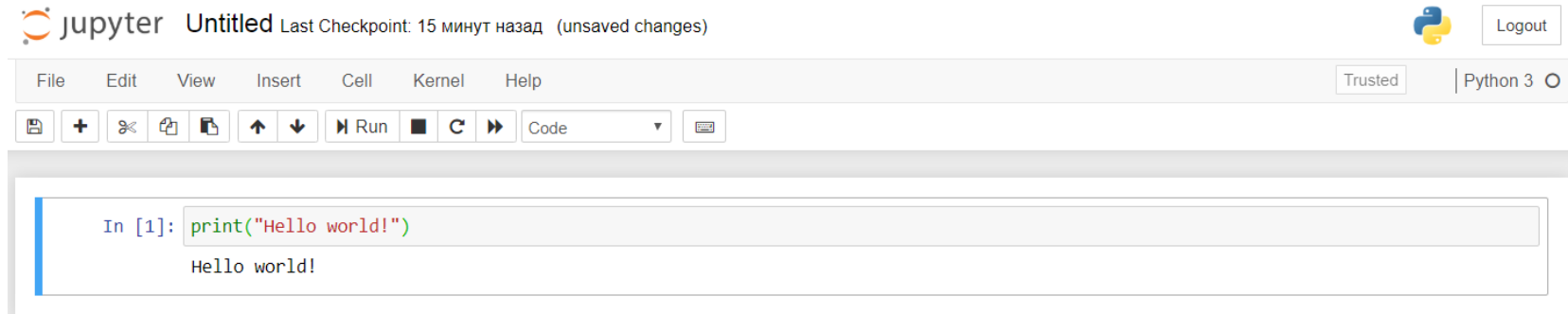
БЛОКНОТ

- Делаем новую папку New->Folder
- Далее создаём проект Python3: New->Python3



The screenshot shows a web browser window with the URL `localhost:8888/notebooks/Untitled%20Folder/Untitled.ipynb?kernel_name=python3`. The browser's address bar and tabs are visible. The Jupyter Notebook interface includes a header with the Jupyter logo, the text "jupyter Untitled Last Checkpoint: 2 минуты назад (autosaved)", and a "Logout" button. Below the header is a menu bar with "File", "Edit", "View", "Insert", "Cell", "Kernel", and "Help". A toolbar contains icons for file operations, a "Run" button, and a "Code" dropdown menu. The main area features a code cell with the prompt `In []:` and an empty text input field.

- Сделаем то же самое и нажмём Ctrl+Enter:



The screenshot shows the same Jupyter Notebook interface as above, but now with a code cell containing the Python code `print("Hello world!")`. The code is highlighted in blue. Below the code, the output `Hello world!` is displayed. The header now shows "Last Checkpoint: 15 минут назад (unsaved changes)".

Вы можете также работать в том стиле, который вам более удобен.

НАША ПЕРВАЯ ОШИБКА

Если Вы решили не использовать Notebook, то далее будем писать программы в *файле-скрипте* *.py

- Используйте удобный редактор для кода!

В Python очень строго с отступами (древовидная конфигурация программы). Без них никакие структуры работать не будут (циклы, условия и т.п.). Отступы можно делать табуляцией или 4 пробелами.

```
if 5>6:  
    h=0
```

Правильно!

```
if 5>6:  
h=0
```

Неправильно! Сообщение об ошибке – `IndentationError`

```
    h=0  
    ^  
IndentationError: expected an indented block
```

Нельзя в одной программе иметь отступы и табуляцией и пробелами, это приведёт к ошибке, которую сложно отследить. В редакторах для кода можно настроить автоматическое преобразование пробелов в табуляцию или наоборот по всей программе.

КОНКАТЕНАЦИЯ

Функция `print()` – вывод аргумента на экран.
Аргумент должен быть строковый, например

`print("Привет мир!")`

Если аргумент числовой, например

`print(5)`

то он будет автоматически переведён в строковый и выведен:

```
D:\python>python code.py
5
```

Для того, чтобы вывести несколько элементов, можно сделать так:

`day = 4`

`print("Сегодня " + day + " число!")`

Это «сцепление» или сумма называется **конкатенацией**.

Однако, выведет это ошибку:

```
D:\python>python code.py
Traceback (most recent call last):
  File "code.py", line 3, in <module>
    print("Сегодня " + day + " число!")
TypeError: can only concatenate str (not "int") to str
```

← Основные подсказки обведены рамочкой: где и какая ошибка.

Поправим так:

`day = 4`

`print("Сегодня " + str(day) + " число!")`

И получится:

```
Сегодня 4 число!
```

ТИПЫ ПЕРЕМЕННЫХ

Python относится к языкам с неявной сильной динамической типизацией. Неявная типизация означает, что при объявлении переменной вам не нужно указывать её тип, при явной – это делать необходимо. В качестве примера языков с явной типизацией можно привести C++.

Запись: `day=4` или `number=5.6` или `stih="Белеет парус одинокий"` сразу определяет тип созданной переменной.

К основным встроенным типам относятся:

`None` (неопределенное значение переменной)

Логические переменные (*Boolean Type*)

Числа (*Numeric Type*)

до 18 значимых

СИМВОЛОВ

`int` – целое число

`float` – число с плавающей точкой

`complex` – комплексное число

Списки (*Sequence Type*)

`list` – список

`tuple` – кортеж

`range` – диапазон

Строки (*Text Sequence Type*)

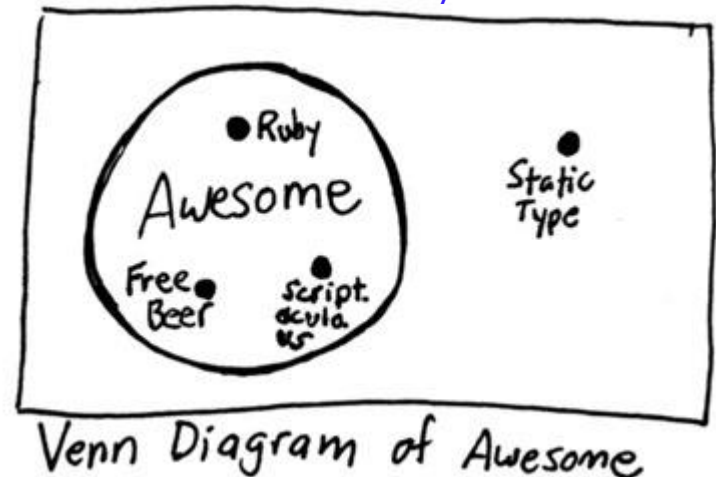
`str`

Бинарные списки (*Binary Sequence Types*)

Множества (*Set Types*)

Словари (*Mapping Types*)

`type(variable)` –
позволит узнать тип



! Нельзя в названии переменных использовать спецсимволы!
! Нельзя начинать имя переменной с цифры!

ОПЕРАЦИИ

Арифметические:

```
>>> 4+3
7
>>> 5-1
4
>>> 6/3
2.0
>>> 5*3
15
>>> --8
8
>>> 4**2
16
>>> 25//7
3
>>> 25%7
4
>>> abs(-7)
7
```

Сложение

Вычитание

Деление

Умножение

Изменение знака

Возведение в степень

Деление без остатка

Деление по модулю

Модуль числа

ОПЕРАЦИИ

Логические:

Во-первых, это операции сравнения: <, >, ==, >=, <=, !=

```
>>> a = 8
>>> b = 3
>>> a + b > 10
True
>>> a - b < 4
False
>>> a != b
True
>>> a >= b
True
>>> b >= a
False
>>> a == b
False
```

Во-вторых, это специальные объединяющие условия операторы, а именно: логические И (**and**) и ИЛИ (**or**) И также инвертирующий **not**

```
>>> a > b and b <= 5
True
>>> a > b or b == 5
True
>>>
```

ОПЕРАЦИИ

Работа с числами:

- Уже известный нам оператор присваивания =

- Краткие формы записи выражений

$i += 2$ соответствует $i = i + 2$

$i -= 2$ соответствует $i = i - 2$

$i *= 2$ соответствует $i = i * 2$

$i /= 2$ соответствует $i = i / 2$

$i //= 2$ соответствует $i = i // 2$

$i \% = 2$ соответствует $i = i \% 2$

$i ** = 2$ соответствует $i = i ** 2$

Операторов инкремента и декремента в **python** нет.

Функция `round()` – округление числа.

Число можно представлять и в других системах счисления:

Двоичная: $x = 0b101$ # число 5

Восьмеричная: $y = 0o11$ # число 9

Шестнадцатеричная: $z = 0x1b$ # число 27

РАБОТА СО СТРОКАМИ

Вывод спецсимвола делается экранированием:

```
print("Сегодня \"четверг\"")
```

```
D:\python>python code.py
Сегодня "четверг"
```

`len()` – выдаст длину строки

`ord()` – выдаст числовое значение строки из 1 символа

Перевод на следующую строку:

```
print("Сегодня \nчетверг")
```

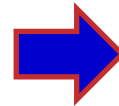
```
D:\python>python code.py
Сегодня
четверг
```

Программирование на Python

```
In [1]: print(ord("q"))
113
```

Если нужен многострочный текст, то можно сделать так:

```
print("Буря мглою небо кроет
Вихри снежные крутя
То как зверь она завоет,
То заплачет как дитя")
```



```
D:\python>python code.py
Буря мглою небо кроет
Вихри снежные крутя
То как зверь она завоет,
То заплачет как дитя
```

Если текст нужно не вывести, а наоборот ввести, используем функцию

`input()`:

```
input("Как Вас зовут?")
```



```
D:\python>python code.py
Как Вас зовут?
```

Программа ожидает ввода.

Как это можно использовать? А вот, например, так:

```
name = input("Как Вас зовут? ")
print("Я знаю, что Вас зовут " + name)
```



```
D:\python>python code.py
Как Вас зовут? Евгений
Я знаю, что Вас зовут Евгений
```

Нужно использовать `raw_input` в `python2`

ЕЩЁ НЕМНОГО О ПЕРЕМЕННЫХ

Если создание переменной делается просто присвоением ей значения, то удалить переменную можно так:

```
a=78
```

```
del a
```

```
print(a)    выведет ошибку:
```

```
Traceback (most recent call last):
  File "code.py", line 6, in <module>
    print(a)
NameError: name 'a' is not defined
```

Метасинтаксические переменные – названия используются в различных примерах: *foo*, *bar*

Имена переменных в `python` регистрозависимые.

ЗАДАНИЕ

Создайте программу, которая просит ввести строку из 3 символов, а выводит сумму числовых значений каждого символа строки.

Обратиться к элементу строки можно так `string[i]`, где $i=0,1,\dots$

УСЛОВИЯ

Тип **Boolean**: два значения **True**, **False**

Для условий будем использовать оператор **if** с логическими выражениями, которые выдаёт значение типа **Boolean**:

```
print(2*2 == 5)
```

```
D:\python>python code.py  
False
```

Лексиграфическое сравнение позволяет сравнивать строки друг с другом:

```
print("test" > "tess")
```

Вес букв на основе их положения в алфавите

```
D:\python>python code.py  
True
```

Научимся делать ветвление на основе логических выражений:

```
if 2*2 == 5:
```

```
    print("Мир сошёл с ума")
```

```
elif 2*2 == 4:
```

```
    print("Всё в порядке!")
```

```
else:
```

```
    print("Ошибка!")
```

И результат:

```
D:\python>python code.py  
Всё в порядке!
```

Логическое выражение может быть сложнее: содержать логические **И/ИЛИ**

ЗАДАЧА

Напишите программу, которая будет запрашивать Ваше имя, а в ответ выведет порядковый номер первой буквы Вашего имени в алфавите.

*) Напишите программу, которая будет запрашивать Ваше имя, а дальше, если оно начинается с гласной буквы, то напишет об этом.