Лекция 1 "Введение в программирование на Python"

часть 1

Финансовый университет при Правительстве РФ, лектор С.В. Макрушин

v 0.7 01.09.2021

Разделы:

-

- к оглавлению

Общая информация:

Присоединяйтесь к группе в Telegram http://t.me/AiSD_Python_21_22 (http://t.me/AiSD_Python_21_22)

Все лекции по дисциплине будут выкладываться в облаке:

- короткая ссылка: https://bit.ly/38rqILL (https://bit.ly/38rqILL)
- полная ссылка: https://yadi.sk/d/WSX0KIDE1mlugQ (https://yadi.sk/d/WSX0KIDE1mlugQ)

Введение / история

- к оглавлению

- Питон (Python) - **высокоуровневый язык программирования**, спроектированный с **акцентом на читаемости исходного кода**.
- Синтаксис Python позволяет создавать **ясный и выразительный исходный код**.
- Python имеет **"батарейки в комплекте"** - обширную стандартную библиотеку, охватывающую ключевые задачи разработки ПО.
- Python **поддерживает различные парадигмы программирования**: императивное, объектно-ориентированное и, частично, функциональное программирование.
- Основные характеристики языка:
  - динамическая неявная типизация
  - переменные ссылаются на значения и передаются по ссылке
  - объекты базовых типов (числа и строки) неизменяемые
  - автоматическое управление памятью.

- Python рассчитан на **исполнение с помощью интерпретатора**. На данный момент доступны интерпретаторы Python для всех основных платформ, и **программы, написанные на Python, могут без изменений запускаться под разными операционными системами**.

Распространенность Python

- Самый известный "рейтинг популярности" языков программирования: https://www.tiobe.com/tiobe-index/ (https://www.tiobe.com/tiobe-index/)
- Рейтинг распространенности языков в проектах на github: https://madnight.github.io/githut/#/pull_requests/2020/2 (https://madnight.github.io/githut/#/pull_requests/2020/2)
- Качество поддержки языка сообществом: https://insights.stackoverflow.com/trends?tags=python%2Cjava%2Cc%2Cc%2B%2B%2Cc%23%2Cjavascript%2Cr (https://insights.stackoverflow.com/trends?tags=python%2Cjava%2Cc%2Cc%2B%2B%2Cc%23%2Cjavascript%2Cr)
- Рейтиг "любимых языков": https://insights.stackoverflow.com/survey/2020#technology-most-loved-dreaded-and-wanted-languages-loved (https://insights.stackoverflow.com/survey/2020#technology-most-loved-dreaded-and-wanted-languages-loved)

Основные приложения Python:

- Обработка и анализ данных
- Разработки приложений (особенно веб-приложений)
- Написания сценариев (скриптов)
  - в т.ч. обработка и анализа данных
  - в специализированных приложениях
- Обучение программированию

Краткая история Питона:

Гвидо ван Россум (Guido van Rossum)

Гвидо ван Россум (Guido van Rossum) — нидерландский программист, автор языка программирования Python. До разработки Python участвовал в проекте по написанию языка для обучения программированию — ABC.

Среди разработчиков Python известен как «великодушный пожизненный диктатор» (BDFL) проекта, что означает, что он продолжает наблюдать за процессом разработки Python, принимая окончательные решения, когда это необходимо.

Для Python существует эффективный процесс развития языка на основе предложений сообщества разработчиков. Основным артефектом процесса является PEP (Python Enhancement Proposals) - предложения по улучшению Python. После обсуждения и принятия PEP создается реализация этого предложения в референсной реализации Python CPython.

- Python 1.0 – 01.1994
- Python 2.0 – 10.2000
  - 2.7 – 07.2010
- Python 3.0 – 12.2008
  - 3.7 – 06.2018
  - 3.8 – 10.2019
  - 3.9 – 10.2020

- Актуальное состояние истории версий Python: https://en.wikipedia.org/wiki/History_of_Python#Version_3 (https://en.wikipedia.org/wiki/History_of_Python#Version_3)

Исходный код, написанный на версии 2.7, распространен и создается. Код, написанный на Python 2.7, не совместим с кодом на Python 3.

Установка Python

- к оглавлению

Официальный диструбутив Python

- Официальный сайт языка Python : https://www.python.org/ (https://www.python.org/)
  - разработчик языка: Python Software Foundation - некоммерческая организация, распространяющая язык свободно и бесплатно
- CPython – эталонная реализация, написанная на C89
- PyPy - реализация Python, альтернативная CPython. PyPy часто работает быстрее, чем CPython, потому что PyPy - это оперативный компилятор, а CPython - интерпретатор. Большая часть кода Python хорошо работает на PyPy, за исключением кода, который зависит от расширений CPython, которые либо не работают, либо вызывают некоторые накладные расходы при запуске в PyPy. Интерпретатор PyPy написан на ограниченном подмножестве Python под названием RPython (Restricted Python).
- Jython – Python для платформы Java.
- IronPython – Python для платформ CLI (включая .NET и Mono)
- CLPython – реализация, написанная на Common Lisp
- Cython - язык программирования упрощающий написание модулей расширения на языках C и C ++ для среды выполнения CPython.
- Numba - это JIT-компилятор с открытым исходным кодом, который переводит подмножество Python и NumPy в быстрый машинный код с использованием LLVM.

Дистрибутивы

- *Дистрибутив эталонной реализации CPython* можно скачать тут: https://www.python.org/downloads/ (https://www.python.org/downloads/)
- *Anaconda* (https://www.anaconda.com/products/individual (https://www.anaconda.com/products/individual)) — дистрибутив языков программирования Python, включающий набор популярных свободных библиотек (~1.5 тыс. модулей), объединённых проблематиками науки о данных и машинного обучения. Ориентирован на упрощение разрешения возникающих зависимостей и конфликтов, которые неизбежны при одиночной установке библиотек.
  - Поддерживаются платформы Linux (x86-64), Windows (i686, x86-64), macOS. Распространяется по лицензии BSD, существует также коммерческая версия (Anaconda Enterprise).

Код на Python - первый взгляд

- к оглавлению

- В исходном коде не объявляется тип переменных, параметров функций и их возвращаемых значений. Это делает код более компактным и гибким, но приводит к потере возможности эффективной проверки кода на этапе компиляции.

- Python отслеживает тип всех переменных во время исполнения программы и сообщает об ошибках в момент, когда встречает их при попытке исполнить код.
- Обычно самым простым и эффективным способом понять, как будет вести себя код на Python, является запуск этого кода.
- Для работы с кодом на Python часто используется REPL (Read Eval Print Loop) - форма организации интерактивной среды программирования, которая получает введенное пользователем выражение, исполняет его и возвращает пользователю полученный результат. Программа, написанная в среде REPL, исполняется по частям.

In [6]:

```
# так оформляется комментарий
# объявление целочисленной переменной:
my_value = 7 # точка с запятой в конце не нужна (но допустима)!
# объявление списка (динамического массива), состоящего из строк:
text_strings = ["one", "two", "three"]

if my_value > 10: # начало блока if
    print("Значение больше пяти") # блок оформляется отступом (фигурные скобки или их анало
else:
    print("Значение меньше пяти")
    for i in text_strings: # цикл по итерируемому объекту
        print(f"Текущее значение {i}") # вывод переменной при помощи форматированной строки
    print("Цикл окончен!")
```

```
Значение меньше пяти
Текущее значение one
Текущее значение two
Текущее значение three
Цикл окончен!
```

В оформлении кода на Питоне очень важны пробелы (особенно отступы) и переходы на новую строку!

- Строка кода:
  - Переход на новую строку (без точки с запятой) начинает новую строку кода.
  - Если строку кода нужно разбить на две строки, то используется символ "\", который экранирует следующий за ним переход на следующую строку.
- Выделение блока кода:
  - В Python для выделения блока кода не нужны фигурные скобки (и их аналоги).
  - Новая строка с большим отступом начинает новый блок (применяются отступы, кратные 4 пробелам, во многих средах разработки табуляция (tab) эквивалентна печати 4х пробелов).
  - Во многих случаях новый блок предваряет двоеточие (например, в циклах, ветвлении и объявлении функций).
  - Новая строка с меньшим отступом находится вне предыдущего блока и, таким образом, заканчивает его.

Философия Python

- Красивое лучше, чем уродливое.
- Явное лучше, чем неявное.
- Простое лучше, чем сложное.
- Сложное лучше, чем запутанное.
- Плоское лучше, чем вложенное.

- Разреженное лучше, чем плотное.
- Читаемость имеет значение.
- Особые случаи не настолько особые, чтобы нарушать правила.
- При этом практичность важнее безупречности.
- Ошибки никогда не должны замалчиваться.
- Если не замалчиваются явно.
- Встретив двусмысленность, отбрось искушение угадать.
- Должен существовать один — и, желательно, только один — очевидный способ сделать это.
- Хотя он поначалу может быть и не очевиден, если вы не голландец[комм 2].
- Сейчас лучше, чем никогда.
- Хотя никогда зачастую лучше, чем прямо сейчас.
- Если реализацию сложно объяснить — идея плоха.
- Если реализацию легко объяснить — идея, возможно, хороша.
- Пространства имён — отличная вещь! Давайте будем делать их больше!

In [3]:

```
# загружаем стиль для оформления презентации
from IPython.display import HTML
from urllib.request import urlopen
html = urlopen("file:./lec_v1.css")
HTML(html.read().decode('utf-8'))
```

Out[3]: