



Основы программирования на Python

лектор: Босенко Тимур Муртазович
bosenkotm@mgpu.ru

Содержание

- Введение
Особенности и философия языка Python, области применения, средства разработки
- Типы данных в Python
Особенности типизации, преобразование типов, строки и срезы
- Управление потоком команд
- Функции в Python
Анонимные функции, область видимости имен, передача параметров, глобальные переменные, вложенные функции
- Модули и пакеты
- Коллекции и генераторы
Списки, кортежи, множества, словари, функции и методы списков, создание и итерирование генераторов списков
- Декораторы функций
Оператор декорирования, использование и создание декораторов
- Обработка исключений и менеджеры контекста

История языков программирования

- **50-е годы**

Начало общения с компьютером, язык ассемблера, появление языка Фортран, эффективной альтернативы ассемблеру, предназначавшегося для математических вычислений

- **60-70-е годы**

Обучение на языках ассемблера или Фортране требовало много сил, появление языков для обучения программированию Basic и Pascal, системное программирование – разработка языка C

- **80-е годы**

Появление ООП, которое должно было упростить создание крупных промышленных программ, язык C++







- **90-е годы**

Развитие персональных компьютеров и сети Интернет, потребность в новых технологиях и языках программирования, языки Java, Python, PHP

- **2000-е годы**

Тенденция объединения технологий вокруг крупных корпораций, развитие языка C# на платформе .NET

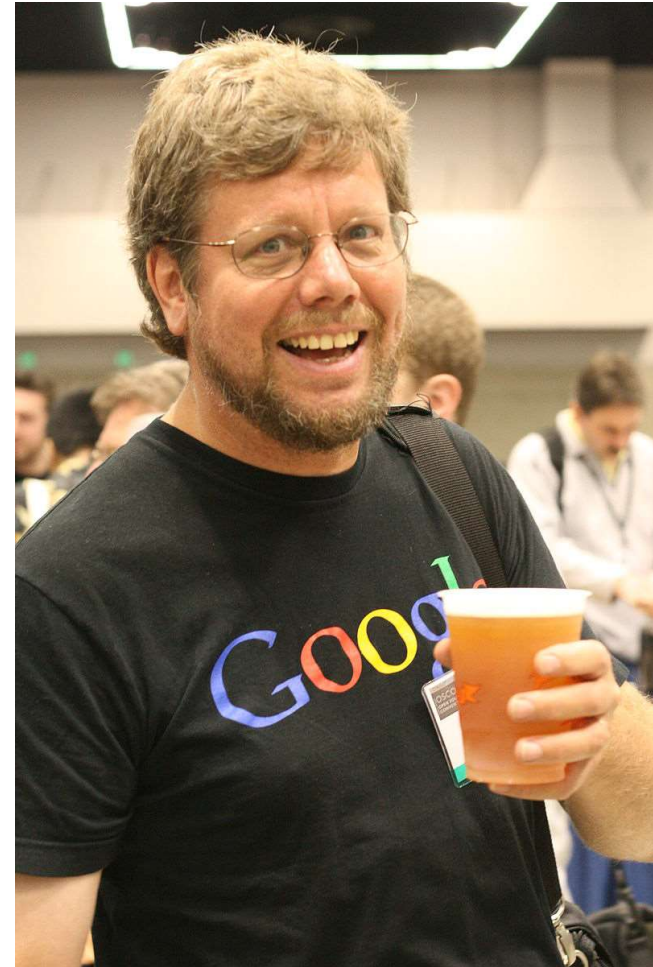
История языков программирования

	Язык ассемблера	Микрокомпьютеры с очень ограниченными ресурсами.		50-ые
	Fortran	Математические расчеты.		
	Basic	Языки для обучения программированию.		60-70-ые
	Pascal			
	C	Системное программирование (драйвера и пр.)		OS UNIX
	C++	Включает все возможности языка C, реализует ООП подход.	Потребность в больших программах - появился подход ООП.	80-ые
	Java	Крупные программы для бизнеса (ООП). Сложно написать плохую программу.	Потребность в программистах и переносимости. Автоматизировать кофемашину.	90-ые
	Python	Автоматизация рутинной деятельности (быстро), обучение программированию.		Персональные ПК, Интернет
	PHP	Разработка динамических сайтов.		
	C# (.NET)	Крупные программы для бизнеса (ООП). Много общего с Java. Зависимость от продуктов Microsoft.	Обобщение и объединение: собрать всё лучшее, что было до этого.	2000-ые

Язык Python

Python был разработан в конце 1989 года **Гuido ван Россумом** во время рождественских каникул, когда его исследовательская лаборатория была закрыта и ему просто нечего было делать

Гuido обожал телевизионную передачу Monty Python's Flying Circus (Летающий цирк Монти Пайтона), и для своего языка он выбрал имя Python



Guido van Rossum

Особенности Python

- **Простота**

Python – простой и минималистичный язык, что позволяет сосредоточиться на решении задачи, а не на самом языке

- **Лёгкость в освоении**

Python обладает исключительно простым удобочитаемым синтаксисом

- **Свобода использования и открытость**

Python – пример свободного и открытого программного обеспечения FLOSS (Free/Libre and Open Source Software)

Свободное распространение и использование, открытые исходные коды, возможность вносить изменения

Python был создан и постоянно улучшается сообществом, которое хочет сделать его лучше

Особенности Python

- **Язык высокого уровня**

Не придётся отвлекаться на низкоуровневые процедуры управления памятью, файловой системой и пр.

- **Кросс-платформенность**

Python можно использовать в GNU/Linux, Windows, FreeBSD, Macintosh, Solaris, OS/2, Amiga, AROS, AS/400, BeOS, OS/390, z/OS, Palm OS, QNX, VMS, Psion, Acorn RISC OS, VxWorks, PlayStation, Sharp Zaurus, Windows CE и даже на PocketPC

- **Интерпретируемый язык**

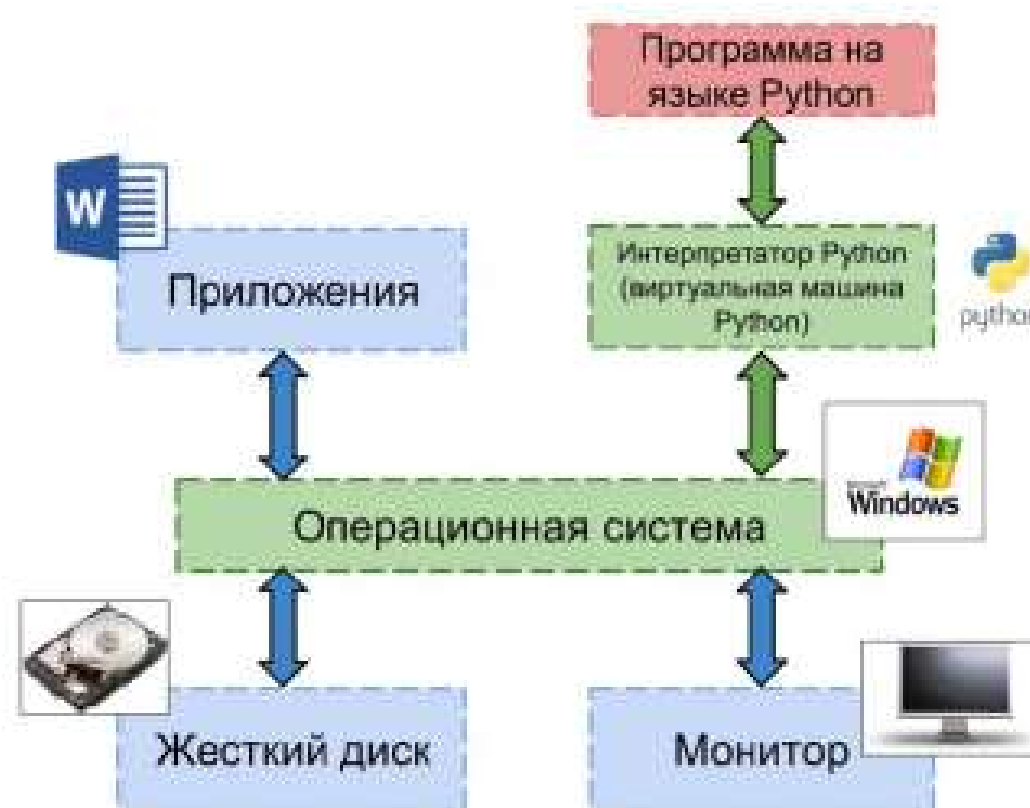
Python не требует компиляции в бинарный код

Особенности Python

- **Объектно-ориентированный язык**
Python поддерживает как процедурно-ориентированное, так и объектно-ориентированное программирование
- **Расширяемый язык**
Из программы на Python может быть вызвана программа, написанная на языке C или C++ (например, для скрывтия части алгоритма или повышения быстродействия)
- **Встраиваемый язык**
Код Python'а можно встраивать в программы на C/C++
- **Обширные библиотеки**
Стандартная библиотека Python предоставляет массу возможностей, в т.ч. для работы со строками, коллекциями, файлами и т.д., написано множество специальных библиотек

Python – интерпретируемый язык

Для запуска программ на языке Python необходима **программа-интерпретатор** (виртуальная машина) Python. Интерпретатор скрывает от Python-программиста все особенности операционной системы, что обеспечивает кросс-платформенность.



Области применения Python

- Системное программирование
- Разработка программ с графическим интерфейсом
- Разработка динамических веб-сайтов
- Интеграция компонентов
- Разработка программ для работы с базами данных
- Быстрое создание прототипов
- Разработка программ для научных вычислений
- Разработка игр
- ...

Области применения Python



Где используется Python?

- Компания Google использует Python в своей поисковой системе и оплачивала труд создателя Python Гвидо ван Россума
- Компании 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 (National Security Agency) использует Python для шифрования и анализа разведданных
- Компании JPMorgan Chase, UBS, Getco и Citadel применяют Python для прогнозирования финансового рынка
- Веб-фреймворк App Engine от компании Google использует Python в качестве прикладного языка программирования
- NASA, Los Alamos, JPL и Fermilab используют Python для научных вычислений

Философия Python

Разработчики языка Python придерживаются философии программирования, называемой “[The Zen of Python](#)”:

- Beautiful is better than ugly (Красивое лучше, чем уродливое)
- Explicit is better than implicit (Явное лучше, чем неявное)
- Simple is better than complex (Простое лучше, чем сложное)
- Complex is better than complicated (Сложное лучше, чем запутанное)
- Flat is better than nested (Плоское лучше, чем вложенное)
- Sparse is better than dense (Разреженное лучше, чем плотное)
- Readability counts (Читабельность имеет значение)
- ...

Полный текст выдаётся интерпретатором Питона по команде `import this`

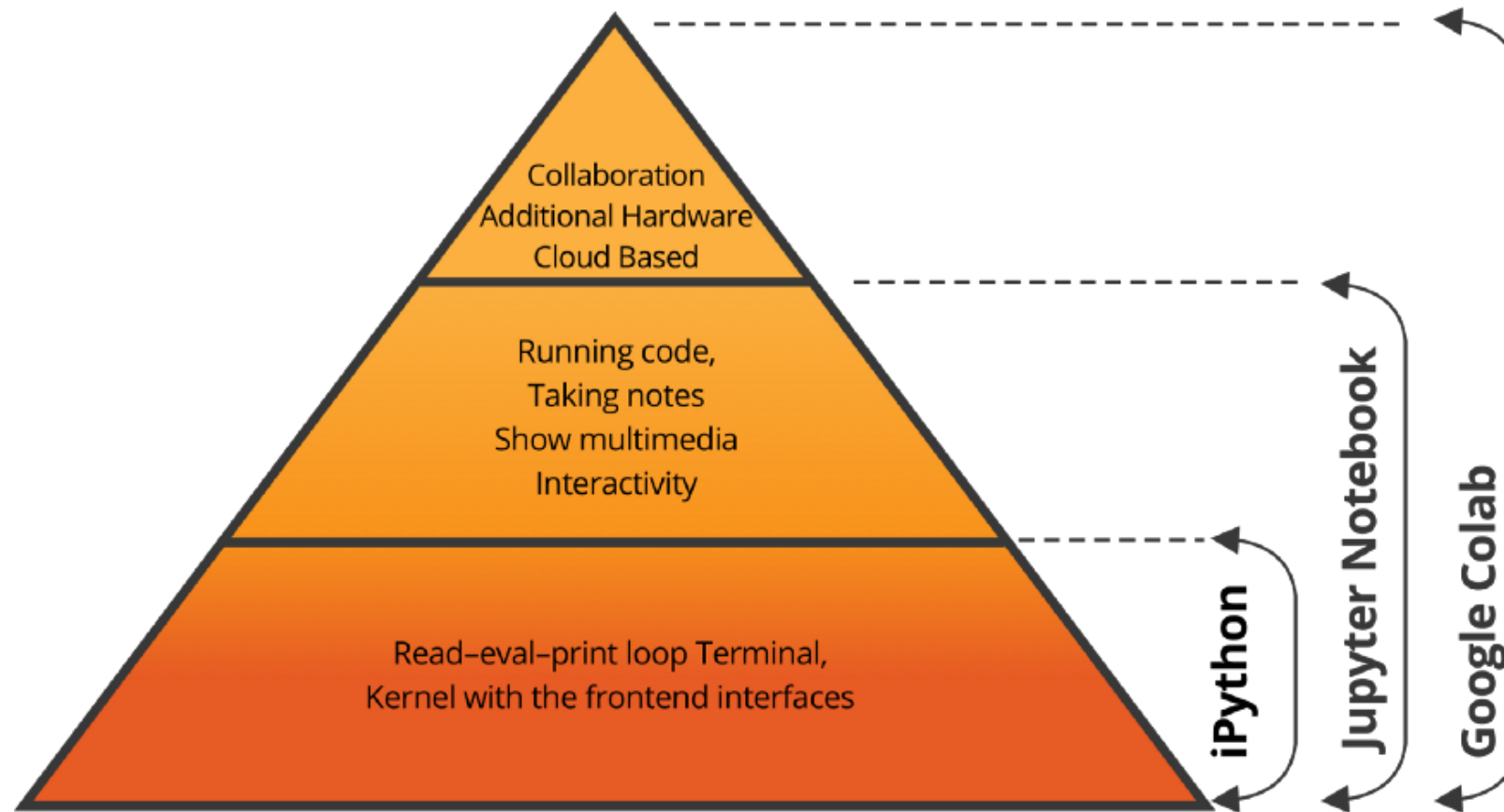
Что говорят программисты

- **Эрик С. Рэймонд** – автор работы «Собор и Базар», а также человек, который ввёл термин “Open Source”. Он говорит, что Python стал его любимым языком программирования
- **Брюс Экель** – автор книг «Думаем на Java» и «Думаем на C++». Он утверждает, что ни на одном языке программирования его работа не была столь эффективной, как на Python. Кроме того, он считает, что Python – это, пожалуй, единственный язык, стремящийся облегчить жизнь программисту
- **Питер Норвиг** – автор книг по программированию на Lisp, директор по качеству поиска в Google. Он говорит, что Python всегда был неотъемлемой частью Google. Python – один из официальных языков разработки Google (наряду с C++ и Java)

Среды разработки Python (IDE)

- Текстовые редакторы с поддержкой Python:
 - Eclipse + PyDev
 - Microsoft Visual Studio
 - Atom
 - GNU Emacs
 - ...
- IDE, разработанные для Python:
 - PyCharm
 - Spyder
 - Thonny
 - ...
- Веб-ориентированные IDE:
 - Jupyter Notebook (<https://jupyter.org>)
 - Google Colab (<https://colab.research.google.com>)

Jupyter Notebook vs Google Colab



Google Colab: большинство библиотек предустановлено, файлы хранятся в облаке, возможность коллаборации с другими разработчиками

PEP

PEP (Python Enhancement Proposal) – документ, содержащий рекомендации по написанию кода на Python

Самый известный PEP – PEP8, создан на основе рекомендаций Гuido ван Россума. Основная цель PEP8 – улучшить читабельность и логичность кода на Python

PEP8:

- Внешний вид кода
- Комментарии
- Контроль версий
- Соглашения по именованию
- Общие рекомендации

Примеры рекомендаций PEP8

Избегайте использования пробелов внутри круглых, квадратных или фигурных скобок

```
spam(ham[1], {eggs: 2}) # good  
spam( ham[ 1 ], { eggs: 2 } ) # bad
```

Никогда не используйте символы `l` (буква “эль”), `o` (буква “о”) или `i` (буква “ай”) как однобуквенные идентификаторы

Имена функций должны состоять из маленьких букв, а слова разделяться символами подчеркивания

Имена классов должны следовать соглашению CapWords

Сравнения с None должны обязательно выполняться с использованием операторов `is` или `is not`, а не с помощью `==`

Всегда используйте `def`, а не лямбда-выражения

Не сравнивайте логические типы с `True` и `False` с помощью `==`

```
if greeting: # good  
if greeting == True: # bad
```