# Python

Python - многоцелевой многофункциональный мультипарадигменный язык программирования.

**Advantages**

1. Качество ПО - удобочитаемость, ясность (Python обладает своей философией (import this). Обладает парадигмами модульности, ООП)
2. Высокая скорость разработки -простой синтаксис, динамическая типизация, отсутствие компиляции, встроенные инструментальные средства (ускоряют процесс разработки), маленький объем кода.
3. Кроссплатформенный код -program portability
4. Библиотеки поддержки (!) - стандартная библиотека (например, NumPy может заменять в некоторых аспектах MatLab)
5. Интеграция компонентов - сценарии могут легко взаимодействовать с другими частями приложения благодаря различным механизмам интеграции. Пример:

* Python может вызывать функции из библиотек C/C++
* интегрироваться с компонентами Java
* может выполнять обмен данными через последовательный порт или по сети с помощью протоколов (SOAP, XML-RPC, CORBA)

1. Удовольствие :)

**Disadvantages**

скорость выполнения программ не такая высокая как у С/С++. Python транслирует инструкции кода в ---> байт-код, а затем интерпретирует. Так как пайтон не создает бинарный код некоторые программы могут работать медленнее аналогов на С.

Python используют: Google (в поисковой системе),YouTube, BitTorrent, движок гугл AppEngine (веб-фреймоворк) использует питон как прикладной язык программирования, ммо игры, Maya, Intel, Cisco, HP для тестирования ПО, машинное обучение, научные вычисления (NASA, Citadel). Пайтон хорошо подходит для прикладного программирования.

**Применение Python**

1. Вообще может использоваться или как инструмент управления другими программными компонентами (управляющий язык), или как для реализации самостоятельных программ.
2. Системное программирование - сценарии могут отыскивать файлы, запускать другие программы, производить парал. вычисления. Стандартная библиотека обладает стандартом POSIX и имеет инструменты (переменные окружения, файлы, сокеты, каналы, процессы).
3. GUI - отличное средство разработки граф. интерфейса(tkinter - стандартный)
4. Internet Scripting - Django для создания веб-сайтов, Python был расширен в rich Internet apps (RIAs) с помощью Silvelight в IronPython, Python также перебрался в cloud computing (с App engine и другими инструментами описанными в пункте 6).
5. Интеграция компонентов - возможность расширения или встраивания Python в системы на языке С/С++. Питон может быть встроен в программные продукты чтобы автоматически настраивать их без необходимости их пересобирать.
6. Приложения баз данных - Python имеет много интерфейсов для работы с БД, имеет переносимый прикладной прог. интерфейс баз данных (portable API) для доступа к SQL бд из скриптов Питона. переносимость означает, что написанный скрипт для одной бд работает и для другой. Всё, что нужно сделать - это заменить низкоуровневый интерфейс (the underlying vendor interface)
7. Быстрое создание прототипов
8. Программирование математических и научных вычислений (NumPy, SciPy и ScientificPython для научного программирования, PyPy для ускоренного численного программирования)

Игры, изображения, искусственный интеллект, XML роботы и тд:  
• Game programming and multimedia with pygame, cgkit, pyglet, PySoy, Panda3D, and others

• Serial port communication on Windows, Linux, and more with the PySerial extension

• Image processing with PIL and its newer Pillow fork, PyOpenGL, Blender, Maya, and more

• Robot control programming with the PyRo toolkit

• Natural language analysis with the NLTK package

• Instrumentation on the Raspberry Pi and Arduino boards

• Mobile computing with ports of Python to the Google Android and Apple iOS

platforms

• Excel spreadsheet function and macro programming with the PyXLL or DataNitro add-ins

• Media file content and metadata tag processing with PyMedia, ID3, PIL/Pillow, and more

• Artificial intelligence with the PyBrain neural net library and the Milk machine

learning toolkit

• Expert system programming with PyCLIPS, Pyke, Pyrolog, and pyDatalog

• Network monitoring with zenoss, written in and customized with Python

• Python-scripted design and modeling with PythonCAD, PythonOCC, FreeCAD,

and others

• Document processing and generation with ReportLab, Sphinx, Cheetah, PyPDF, and so on

• Data visualization with Mayavi, matplotlib, VTK, VPython, and more

• XML parsing with the xml library package, the xmlrpclib module, and third-party extensions

• JSON and CSV file processing with the json and csv modules  
• Data mining with the Orange framework, the Pattern bundle, Scrapy, and custom code

**Сила Питона**

1. ООПешный и функциональный (есть генераторы, замыкания, включения - comprehensions, мапы, декораторы, анонимные лямбда функции и функции-объекты первого класса)
2. Он свободный
3. Переносимый
4. Он мощный
5. Смешивающийся
6. Относительно прост в использовании (rapid turnaround - быстрый, питон - выполняемый псевдокод)
7. Относительно прост в изучении
8. it’s Named After Monty Python

Powerful:

* dynamic typing: код на питоне не ограничивает типы данных
* automatic memory management: питон автоматически выделяет объекты и реклеймит их (reclaim - собирает мусор)
* programming-in-the-large support: питон включает модули, классы и экспепшены. Эта возможность позволяет организовать системы в компоненты.
* built-in object types: списки, словари, строки
* built-in tools: операции для работы с object types - конкатенация (коллеций), slicing (извлечение секций- части колеций), сортировка, мапинг и другое.
* library utilities: большая часть действий приложений происходит именно с библиотекой (в ней есть всё от матчинга регулярных выражений до инструментов работы по сети)
* third-party utilities: разрабы могут добавлять еще свои инструменты кроме тех, что уже модульные (built-ins)

The result is a powerful programming tool with all the usability of a scripting language.

## Интерпретатор Python

1. Что такое интерпретатор Python?

Интерпретатор Python – это программа, которая выполняет программы на

языке Python.

2. Что такое исходный программный код?

Исходный программный код – это инструкции, составляющие программу. Он состоит из текста в текстовых файлах, имена которых обычно имеют

расширение .py.

3. Что такое байт-код?

Байт-код – это низкоуровневое представление программы после ее компиляции. Python автоматически сохраняет полученный байт-код в файлах .pyc

4. Что такое PVM?

PVM – это Python Virtual Machine (виртуальная машина Python) – механизм Python, который интерпретирует компилируемый программный код.

5. Назовите две разновидности стандартной модели выполнения Python.

Psyco, Shedskin и фиксированные двоичные файлы – все это разновидности модели выполнения.

6. В чем заключаются различия между CPython, Jython и IronPython?

CPython – это стандартная реализация языка. Jython и IronPython реализуют поддержку программирования на языке Python в среде Java и .NET соответственно они являются альтернативными компиляторами языка Python.

## Запуск программ

1. Интерактивный режим - полезен для написания небольших программ и их тестирования на лету
2. Системная командная строка
3. Щелчок мыши
4. IDLE
5. SublimeREPL

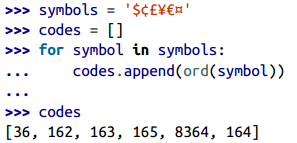
## Built-in sequences

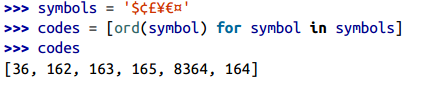
Container sequences: list, tuple & collections.deque (все они содержат элементы различных типов)

Flat sequences: str, bytes, bytearray, memoryview & array.array содержат элементы только одного типа.

### list comprehensions (списковые выражения или генераторы списков)

Самый быстрый способ построить sequence - это использовать **list comprehension** (если цель - список) или **generator expression** (для других видов последовательностей).

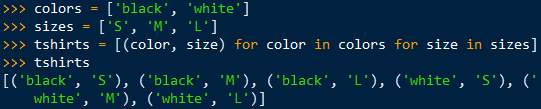
 - наращивает (building up) существующий список

- создает новый список

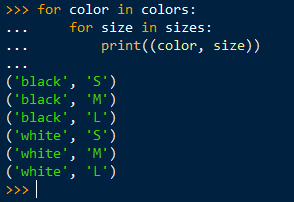
**listcomps** быстрее чем **map & filter**

Cartesian Product using List Comprehensions:

listcomp здесь создаёт список кортежей. Элементы списка сортируются по color (так как он идёт первым, если size - первый, то по нему)



Получится аналогичное при использовании обычных циклов for (сортировка по внешнему условию for):

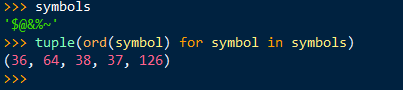


### generator expressions

Как и с помощью генераторов списков, с помощью генераторов выражений можно инициализировать кортежи, массивы и другие типы последовательностей.

Преимущество genexp: он экономит память, так как возвращает элементы один за другим, используя итераторы. Это лучше чем генераторы списков, так как зачем строить целый список чтобы накормить другой конструктор? :)

genexp заключаются в круглые скобки (are enclosed in parentheses) :



Итак, генераторы довольно полезны когда нужно выводить выражения (без сохранения в памяти) или для инициализации других выражений (ни списков :) )

### tuples