FPT Education
FPT UNIVERSITY

Python For Engineer

Bùi Văn Hiệu
0933189292
hieubv10@fe.edu.vn

Course contents

- 1: Python Variables and Operators
- 2: Python Data Types
- 3: Python Conditions and Loops
- 4: Python Functions
- 5: Python Input and Output Handling
- 6: Python Class/Object, and Modules/Packages
- 7: Python Visualization

Grading Policy

- Attendance, In-class Lab and Assignment: 20%
- Quiz: 20%
- Exercise: 20%
- Assignment (Project Presentation): 40%

Class code-google classroom: 4x45gle

Link:https://classroom.google.com/c/NzM2MjAxNjg0ODQw?cjc=4x45gle

References

eBooks:

[1] Severance, C. R. (2016). *Python for everybody: Exploring data in Python*

[2] Andrew N. and Harrington Hands on Python Tutorial

Online courses:

[1] Python for Everybody Specialization www.coursera.org

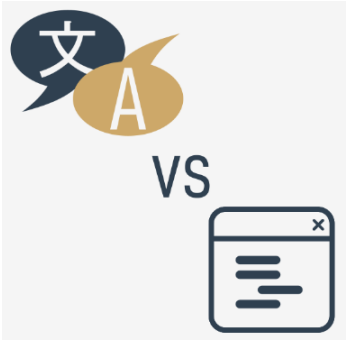
[2] Python Essentials - Part 1,2 https://edube.org/

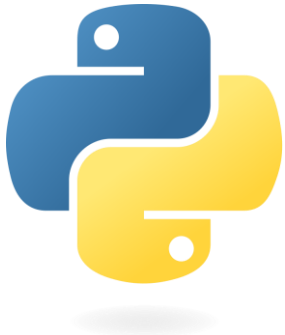
Introduction to Python and computer programming

- The fundamentals of computer programming, i.e.,
  - how the computer works, how the program is executed,
  - how the programming language is defined and constructed;
- The difference between compilation and interpretation
- What Python is, how it is positioned among other programming languages, and what distinguishes the different versions of Python.

Python as a Language

- Natural languages vs. programming languages
  - A language is a means (and a tool) for expressing and recording thoughts.
  - Language you use each day is your mother tongue, which you use to manifest your will and to ponder reality.
  - Computers have their own language, too, called machine language, which is very rudimentary.





What makes a language?

- An alphabet: a set of symbols used to build words of a certain languag
- A lexis: (aka a dictionary) a set of words the language offers its users
- A syntax: a set of rules (formal or informal, written or felt intuitively) used to determine if a certain string of words forms a valid sentence
- Semantics: a set of rules determining if a certain phrase makes sense

- The IL is the alphabet of a machine language. This is the simplest and most primary set of symbols we can use to give commands to a computer

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

shutterstock.com · 1375463840

Compilation vs. interpretation

FPT Education
FPT UNIVERSITY

- There are two different ways of transforming a program from a high-level programming language into machine language:
- COMPILATION - the source program is translated once by getting a file containing the machine code; now you can distribute the file worldwide; the program that performs this translation is called a compiler or translator;
- INTERPRETATION - you can translate the source program each time it has to be run; the program performing this kind of transformation is called an interpreter, as it interprets the code every time it is intended to be executed; it also means that you cannot just distribute the source code as-is, because the end-user also needs the interpreter to execute it.

Compilation vs. interpretation

FPT Education
FPT UNIVERSITY

COMPILATION	INTERPRETATION
• the execution of the translated code is usually faster; • only the user has to have the compiler - the end-user may use the code without it; • the translated code is stored using machine language - as it is very hard to understand it, your own inventions and programming tricks are likely to remain your secret.	• you can run the code as soon as you complete it - there are no additional phases of translation; • the code is stored using programming language, not machine language - this means that it can be run on computers using different machine languages; you don't compile your code separately for each different architecture.
• the compilation itself may be a very time-consuming process - you may not be able to run your code immediately after making an amendment; • you have to have as many compilers as hardware platforms you want your code to be run on.	• don't expect interpretation to ramp up your code to high speed - your code will share the computer's power with the interpreter, so it can't be really fast; • both you and the end user have to have the interpreter to run your code.

What is Python?

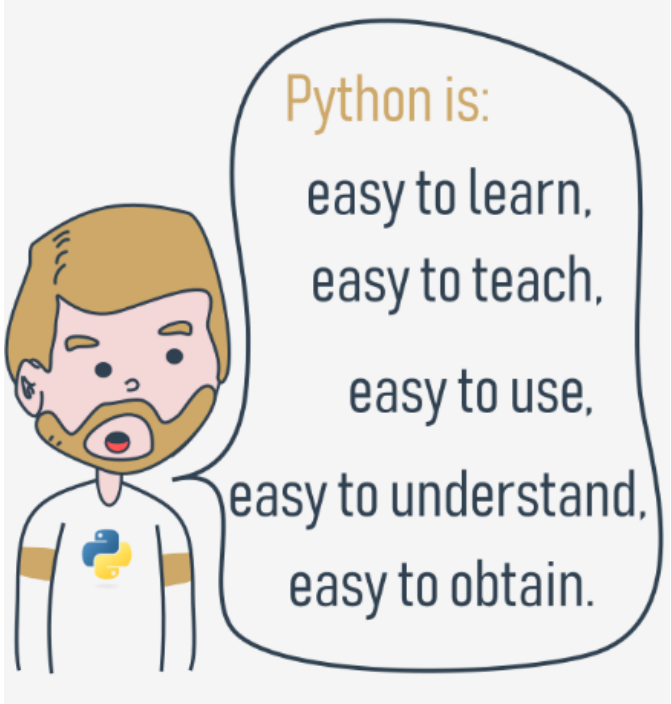
- Python is a widely-used, interpreted, object-oriented, and high-level programming language with dynamic semantics, used for general-purpose programming.
- Name of the Python programming language comes from an old BBC television comedy sketch series called Monty Python's Flying Circus.
- One of the amazing features of Python is the fact that it is actually one person's work.
- Python was created by Guido van Rossum, born in 1956 in Haarlem, the Netherlands
- The speed with which Python has spread around the world is a result of the continuous work of thousands (very often anonymous) programmers, testers, users (many of them aren't IT specialists) and enthusiasts, - Guido's.

Python goals

> In December 1989, I was looking for a "hobby" programming project that would keep me occupied during the week around Christmas. My office (...) would be closed, but I had a home computer, and not much else on my hands I decided to write an interpreter for the new scripting language I had been thinking about lately: a descendant of ABC that would appeal to Unix/C hackers. I chose Python as a working title for the project, being in a slightly irreverent mood (and a big fan of Monty Python's Flying Circus).
>
> — *Guido van Rossum*

- In 1999, Guido van Rossum defined his goals for Python:
  - An easy and intuitive language just as powerful as those of the major competitors;
  - Open source, so anyone can contribute to its development;
  - Code that is as understandable as plain English;
  - Suitable for everyday tasks, allowing for short development times.

What makes Python special?



- Python has its drawbacks:
  - it's not a speed demon – Python does not deliver exceptional performance;
  - in some cases it may be resistant to some simpler testing techniques – this may mean that debugging Python code can be more difficult than with other languages; fortunately, making mistakes is also harder in Python.

Why Python, why not?

FPT Education
FPT UNIVERSITY

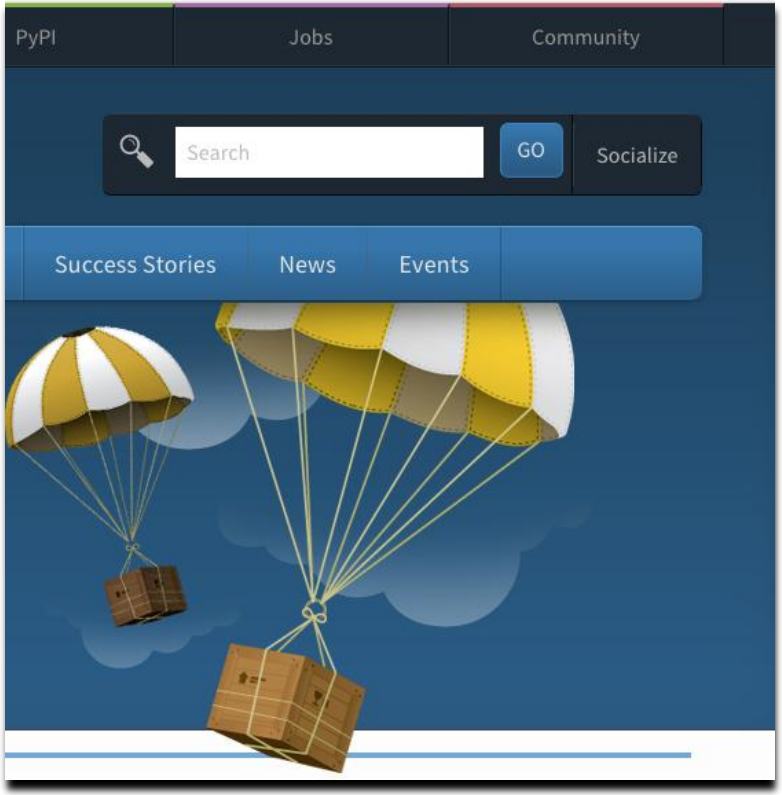
- Python rivals?
  - Perl – a scripting language originally authored by Larry Wall;
  - Ruby – a scripting language originally authored by Yukihiro Matsumoto.
- Where can we see Python in action?
  - We see it every day and almost everywhere. It's used extensively to implement complex Internet services like search engines, cloud storage and tools, social media and so on.
  - Lots of scientists have abandoned expensive proprietary tools and switched to Python. Lots of IT project testers have started using Python to carry out repeatable test procedures
- Why not Python? low-level programming, applications for mobile devices

CPython and Cython

- Python aka CPython
  - Pythons which are maintained by the people gathered around the PSF (Python Software Foundation), a community that aims to develop, improve, expand, and popularize Python and its environment
  - Guido van Rossum used the "C" programming language to implement the very first version of his language and this decision is still in force. All Pythons coming from the PSF are written in the "C" language.
- Cython
  - This is what Cython is intended to do – to automatically translate the Python code (clean and clear, but not too swift) into "C" code (complicated and talkative, but agile).
- Jython, PyPy, RPython

Downloading and installing Python

- Linux users most probably have Python already installed - this is the most likely scenario, as Python's infrastructure is intensively used by many Linux OS components.
  - Python3
- Windows user, Downloading and installing Python
  - look at the checkbox named Add Python 3.x to PATH and check it.
- MacOS user download and install the relevant .pkg file from the Python site



Starting your work with Python

- IDLE
  - an editor which will support you in writing the code (it should have some special features, not available in simple tools); this dedicated editor will give you more than the standard OS equipment;
  - a console in which you can launch your newly written code and stop it forcibly when it gets out of control;
  - a tool named a debugger, able to launch your code step-by-step, which will allow you to inspect it at each moment of execution.

How to spoil and fix your code

```
>>>
================== RESTART: C:/Users/Snake/Desktop/snake.py ==================
Traceback (most recent call last):
  File "C:/Users/Snake/Desktop/snake.py", line 1, in <module>
    prin("Hisssssss...")
NameError: name 'prin' is not defined
>>>
```

- the traceback (which is the path that the code traverses through different parts of the program - you can ignore it for now, as it is empty in such a simple code);
- the location of the error note: the number may be misleading, as Python usually shows the place where it first notices the effects of the error, not necessarily the error itself;
- the content of the erroneous line; note: IDLE's editor window doesn't show line numbers, but it displays the current cursor location the name of the error and a short explanation.
- the name of the error and a short explanation.

TOOLS

FPT Education
FPT UNIVERSITY

Colab : Google Colaboratory is a hosted Jupyter notebook environment that is free to use and requires no setup. It's a free cloud service and it supports free GPU!

You can simply improve your Python programming language coding skills OR develop deep learning applications using popular libraries such as Keras, TensorFlow, PyTorch, and OpenCV

TOOLS

- Anaconda : Anaconda is a python and R distribution. It aims to provide everything you need (python wise) for data science tasks.
- Anaconda is a set of binaries that includes Scipy, Numpy, Pandas along with all their dependencies.

How to know where you are?

- your code wants to create a file, so it invokes one of Python's functions;
- Python accepts the order, rearranges it to meet local OS requirements, which is like putting the stamp "approved" on your request, and sends it down (this may remind you of a chain of command)
- the OS checks if the request is reasonable and valid (e.g., whether the file name conforms to some syntax rules) and tries to create the file; such an operation, seemingly very simple, isn't atomic – it consists of many minor steps taken by...
- the hardware, which is responsible for activating storage devices (hard disk, solid state devices, etc.) to satisfy the OS's needs.