Beginning with Python

What is Python?

Python is a general-purpose interpreted, interactive, object-oriented, and high-level popular programming language. It was created by Guido van Rossum during 1985-1990 and released in 1991 well before the Java programming language (1995).

We may know the python as a large snake but the name of the Python programming language comes **from an old BBC television comedy sketch series named "Monty Python's Flying Circus", to which the developer** Guido van Rossum was a great fan.

It is the fastest growing programming language in terms of developers and companies using it, in terms of libraries, in terms of application fields e.g. web development, software development, machine learning, GUI design etc., it is used etc.

It is used for all type of application including:

- web development (server-side),
- software development,
- Mathematical computations,
- Artificial intelligence,
- system scripting.

What can Python do?

- Python can be used on a server to create web applications.
- Python can be used alongside software to create workflows.
- Python can connect to database systems. It can also read and modify files.
- Python can be used to handle big data and perform complex mathematics.
- Python can be used for rapid prototyping, or for production-ready software development.

Why Python?

- Python works on different platforms (Windows, Mac, Linux, Raspberry Pi, etc).
- Python has a simple syntax similar to the English language.
- Python has syntax that allows developers to write programs with fewer lines than some other programming languages.
- Python runs on an interpreter system, meaning that code can be executed as soon as it is written. This means that prototyping can be very quick.
- Python can be treated in a procedural way as well as in an object-orientated way or a functional way.

Characteristics

- **Easy-to-learn** − Python has few keywords, simple structure, and a clearly defined syntax. This allows the student to pick up the language quickly.
- **Easy-to-read** − Python code is more clearly defined and visible to the eyes.
- **Easy-to-maintain** − Python's source code is fairly easy-to-maintain.
- **A broad standard library** − Python's bulk of the library is very portable and cross-platform compatible on UNIX, Windows, and Macintosh.
- **Interactive Mode** − Python has support for an interactive mode which allows interactive testing and debugging of snippets of code.
- **Portable** − Python can run on a wide variety of hardware platforms and has the same interface on all platforms.
- **Extendable** − You can add low-level modules to the Python interpreter. These modules enable programmers to add to or customize their tools to be more efficient.
- **Databases** − Python provides interfaces to all major commercial databases.
- **GUI Programming** − Python supports GUI applications that can be created and ported to many system calls, libraries and windows systems, such as Windows MFC, Macintosh, and the X Window system of Unix.
- **Scalable** − Python provides a better structure and support for large programs than shell scripting.

Good to know

- The most recent major version of Python is Python 3, which we shall be using in this tutorial. However, Python 2, although not being updated with anything other than security updates, is still quite popular and these two version are completely different.
- In this tutorial Python will be written in a text editor. It is possible to write Python in an Integrated Development Environment, such as Thonny, Pycharm, Netbeans or Eclipse which are particularly useful when managing large collections of Python files.

Python Syntax compared to other programming languages

- Python was designed for readability, and has some similarities to the English language with influence from mathematics.
- Python uses new lines to complete a command, as opposed to other programming languages which often use semicolons or parentheses.
- Python relies on indentation, using whitespace, to define scope; such as the scope of loops, functions and classes. Other programming languages often use curly-brackets for this purpose.

Installing Python in Windows

To install Python on a Windows machine follow the instructions given in the link below.
https://www.howtogeek.com/197947/how-to-install-python-on-windows/

Installing Python in Ubuntu Linux

To install python on a Ubuntu Linux machine follow the steps given in the link below.
https://linuxize.com/post/how-to-install-python-3-7-on-ubuntu-18-04/

For other linux you can search the Internet for the steps.

What is Colaboratory?

Colaboratory is a **Google** research project created to help disseminate machine learning education and research. It's a Jupyter notebook environment that requires no setup to use and runs entirely in the cloud.

Colaboratory, or "Colab" for short, allows you to write and execute Python in your browser, with

- Zero configuration required
- Free access to GPUs
- Easy sharing

Whether you're a **student**, a **data scientist** or an **AI researcher**, Colab can make your work easier.

To start in Google colab open the link https://colab.research.google.com
And sign in with your college Email or gmail account. At the beginning create a notebook from File menu, where all your code will be saved. To write a new code section Click +code section. To run the code Click on the Play Button Present left of that Code section.

Python Quickstart

Python is an interpreted programming language, this means that as a developer you write Python (.py) files in a text editor and then put those files into the python interpreter to be executed.

The way to run a python file is like this on the command line:

```
$ python helloworld.py
```

Where "helloworld.py" is the name of the python file.

Let's write our first Python file, called helloworld.py, which can be done in any text editor.

```
Helloworld.py
print("Hello, World!")
```

Simple as that. Save your file. Open your command line, navigate to the directory where you saved your file, and run:

```
$ python helloworld.py
The output should read:
Hello, World!
```

Congratulations, you have written and executed your first Python program.

The Python Command Line

To test a short amount of code in python sometimes it is quickest and easiest not to write the code in a file. This is made possible because Python can be run as a command line itself.

Type the following on the Windows, Mac or Linux command line:

```
$ python
```

Or, if the "python" command did not work, you can try "py":

```
$ py
```

From there you can write any python, including our hello world example from earlier in the tutorial: print("Hello, World!")

Which will write "Hello, World!" in the command line:

```
$ python
```

```
Python 3.6.4 (v3.6.4:d48eceb, Dec 19 2017, 06:04:45) [MSC v.1900 32 bit (Intel)] on win32
Type "help", "copyright", "credits" or "license" for more information.
Adding two numbers
>>>3+8
11
Subtracting two numbers
>>> 3-8
-5

Multiplying two numbers
>>> 3*8
24
Division of two numbers (Floating division)
>>> 8/3
2.6666666666666665
Division of two numbers (Integer division)
>>> 8//3
2
Modulo division
>>> 8%3
2

Exponential operation
>>> 4**3
64
Complex operation
>>> 4+5*9
49
Complex operation with brackets
>>> (4-5)*9
```

```
-9


String handling
>>> CEMK
Traceback (most recent call last):
  File "<stdin>", line 1, in <module>
NameError: name 'CEMK' is not defined
>>> 'CEMK'
'CEMK'
>>> print('CEMK')
CEMK
>>> 'CEMK'+' College'
'CEMK College'

>>> 10*'CEMK'
'CEMKCEMKCEMKCEMKCEMKCEMKCEMKCEMKCEMKCEMK'
>>> 10*' CEMK'
' CEMK CEMK CEMK CEMK CEMK CEMK CEMK CEMK CEMK CEMK'
>>> print(10*' CEMK')
 CEMK CEMK CEMK CEMK CEMK CEMK CEMK CEMK CEMK CEMK
>>> print('my path is c:\adc\notr')
my path is c:dc
otr
>>> print('my path is c:\\adc\\notr')
my path is c:bc\notr
Also work with double quotation “ “
>>> print("Hello, World!")
Hello, World!
```

Whenever you are done in the python command line, you can simply type the following to quit the python command line interface:

```
exit()
```

Block Indentation

Python uses indentation to define control and loop constructs. This contributes to Python's readability, however, it requires the programmer to pay close attention to the use of whitespace. Thus, editor miscalibration could result in code that behaves in unexpected ways.
Python uses the colon symbol (:) and indentation for showing where blocks of code begin and end. That is, blocks in Python, such as functions, loops, if clauses and other constructs, have no ending identifiers. All blocks start with a colon and then contain the indented lines below it.

For example:

```
def my_function():  # This is a function definition.Note the colon (:)
     a = 2  # This line belongs to the function because it's indented
     return a  # This line also belongs to the same function
print(my_function())   # This line is OUTSIDE the function block
```

OR

```
if a > b:        # If block starts here
     print(a)    # This is part of the if block
else:            # else must be at the same level as if
     print(b)    # This line is part of the else block
```

Blocks that contain exactly one single-line statement may be put on the same line, though this form is generally not considered good style:

```
     if a > b: print(a)
          else: print(b)
```

Attempting to do this with more than a single statement will not work:

```
     if x > y: y = x
          print(y)     # IndentationError: unexpected indent
```

```
if x > y: while y != z: y -= 1
# SyntaxError: invalid syntax
```

Comments

Comments can be used to explain Python code.

Comments can be used to make the code more readable.

Comments can be used to prevent execution when testing code.

Creating a Comment

Comments starts with a #, and Python will ignore them:

Example

```
#This is a comment
print("Hello, World!")
```

Comments can be placed at the end of a line, and Python will ignore the rest of the line:

Example

```
print("Hello, World!") #This is a comment
```

Multi Line Comments

Python does not really have a syntax for multi line comments.

To add a multiline comment you could insert a # for each line:

Example

```
#This is a comment
#written in
#more than just one line
print("Hello, World!")
```

Or, not quite as intended, you can use a multiline string.

Since Python will ignore string literals that are not assigned to a variable, you can add a multiline string (triple quotes) in your code, and place your comment inside it:

Example

```
"""
This is a comment written in
more than just one line
"""
print("Hello, World!")
```

Python help

To get help in python type help() in the command line and in the ‘help> ‘prompt type any function to get help about this.

```
>>> help>
help> <command>
To exit from help type quit
help> quit
```