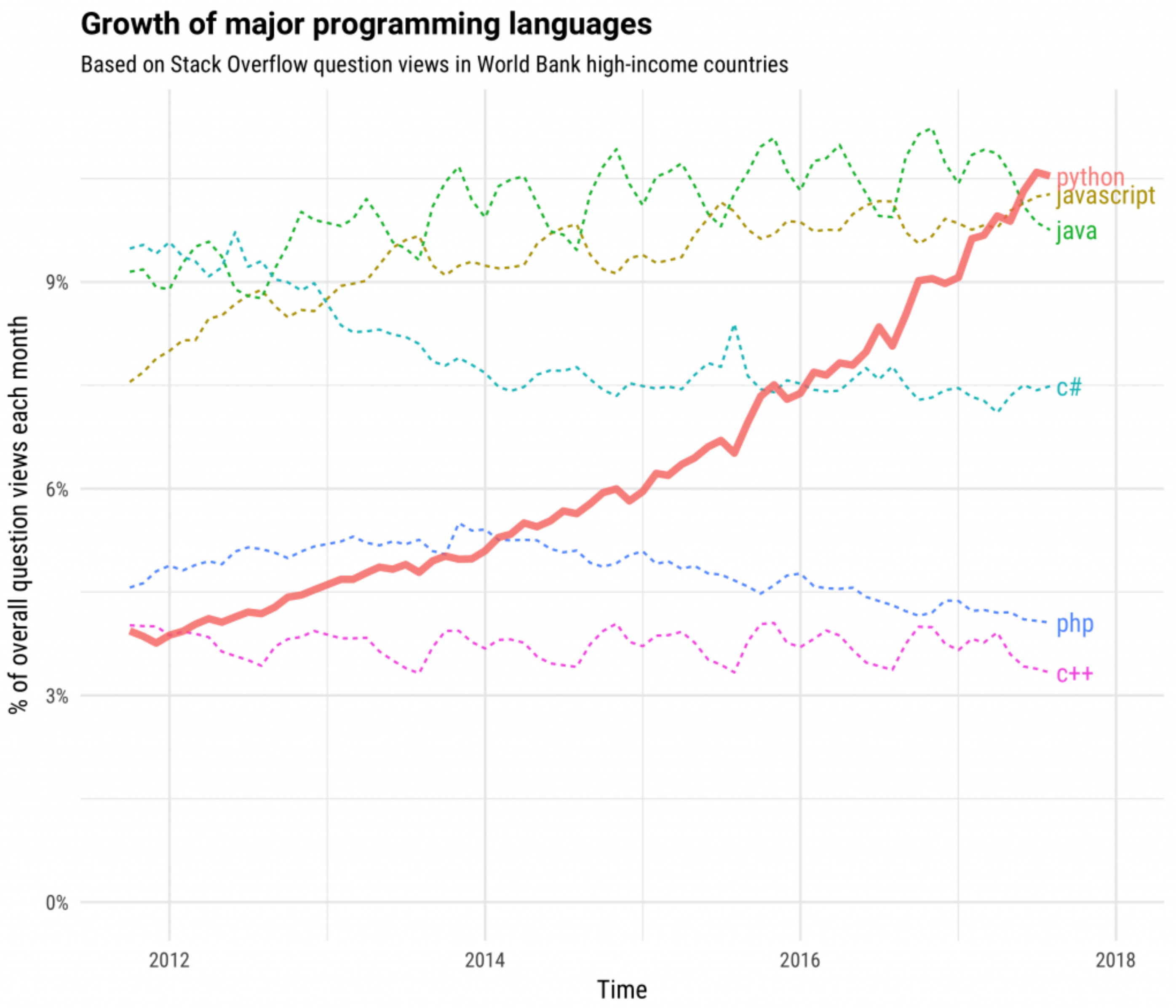
Getting Started

A byte of Python

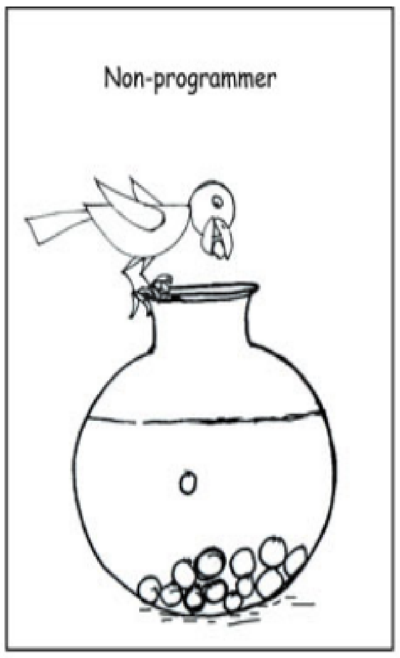
- Interpreted high-level programming language for general-purpose programming
- Created by Guido van Rossum and first released in 1991
- Design philosophy that emphasizes code readability
- Extraordinary growth of the Python programming language in the last five years, as seen by Stack Overflow traffic within high-income countries



source : https://stackoverflow.blog/2017/09/06/incredible-growth-python/

Why Python ?

- High level programming language
- Designed to be easy to learn and master
- Clean and clear syntax
- Highly portable
- Lots of libraries
- Extremely readable code



Topics covered in Python Tutorials

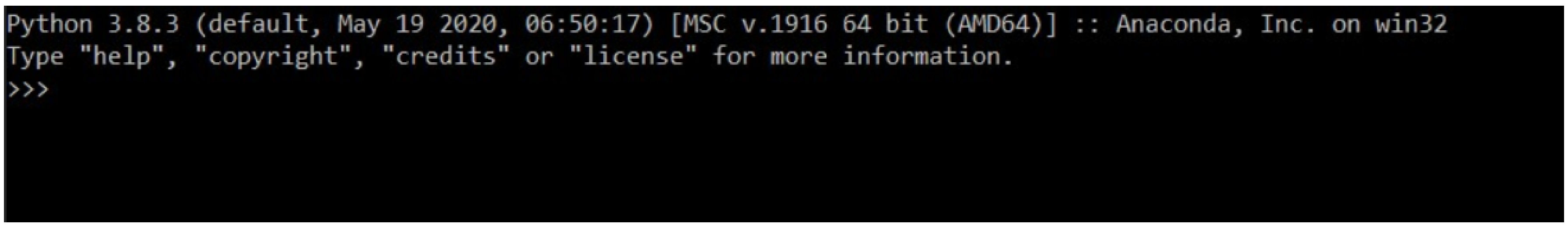
- Writing a Python script
- Print and input
- Basic data types
- Strings
- Lists
- Dictionaries
- Tuples
- Conditional statements
- Control structures
- Loops
- Functions
- Modules

Topics covered in this tutorial

- How to run an interactive python session in terminal
- Understanding difference between Compiler and Interpreter
- Running Python script
- print statement
- Escape Sequence
- input - return string,
- reserved keywords
- Code Comments

Interactive Python session in Terminal

- Useful for testing small python code
- Open Terminal -> type `python` and press Enter
- Python prompt will open as shown in figure below



```
Python 3.8.3 (default, May 19 2020, 06:50:17) [MSC v.1916 64 bit (AMD64)] :: Anaconda, Inc. on win32
Type "help", "copyright", "credits" or "license" for more information.
>>>
```

- To exit from prompt -> type `exit()` and press Enter

Running Python Program

- Python program are saved a with a **.py** extension. Example: filename.py
- Python program can be run from terminal by typing `python filename.py`

How to print on console ?

In [1]:

```
print ("Hello world")
```

```
Hello world
```

In [2]:

```
# %s, %d and %f are called format specifiers
print ("%s is a string, %d is a decimal integer and %.3f is floating number upto 3 decimal place" %("Python",4,3.33))
```

```
Python is a string, 4 is a decimal integer and 3.330 is floating number upto 3 decimal place
```

In [2]:

```
# Avoiding new line with multiple prints

print ("1", end="")
print ("2", end="")
print ("3")
```

```
123
```

In [4]:

```
print ("Welcome to e-Yantra Robotics Competition({})-{}!".format('eYRC',2020))
```

```
Welcome to e-Yantra Robotics Competition(eYRC)-2020!
```

How to input from user ?

In [5]:

```
name = input("Type your name and press enter: ")     # input reads a line from input and return it as string
print ("Welcome '%s' to e-Yantra tutorials." %name)
# type of name
# typecasting into other datatypes
```

```
Type your name and press enter: Alice
Welcome 'Alice' to e-Yantra tutorials.
```

Understanding difference between Compiler and Interpreter

Compiler coverts the code as a whole from high level language to machine code whereas, Interpreter converts the code line by line from high language to machine code.

In [6]:

```
test=int(input("This is an incorrect statement "))
test=test1+1
test
```

```
This is an incorrect statement 25

---------------------------------------------------------------------------
NameError                                 Traceback (most recent call last)
<ipython-input-6-8892d010c5c5> in <module>
      1 test=int(input("This is an incorrect statement "))
----> 2 test=test1+1
      3 test

NameError: name 'test1' is not defined
```

Keywords

- Keywords are reserved words which are used by language
- Keywords can not be used as variable name

List of Keywords in python

- `and` , `exec` , `not` , `assert` , `finally` , `or` , `break` , `for` , `pass` , `class` , `from` , `print` , `continue` , `global` , `raise` , `del` ,
- `if` , `return` , `import` , `try` , `elif` , `in` , `while` , `else` , `is` , `with` , `except` , `lambda` , `yield`

Comments

Single Line comment

use `#` for single line comment

Multi Line comment

use `'''` (three single quotes)

Note: This is not PEP-8 supported coding standard. Use multiple single line comment to conform to PEP-8.

for example:

In [7]:

```
'''
This is an example to demonstrate single and multi-line comment

Multi-line comment
'''
# single line comment

print ("hello commenting")
```

```
hello commenting
```