Python Programming

Programming Languages

Computer Hardware

- Hardware components: CPU, RAM, Disk, Keyboard, Monitor.
- Program is set of instructions executed by the CPU.

Computer Language

- What is Computer Language?
  - helps us to interact with hardware
  - medium of communication with hardware
  - implementation of algorithm i.e. program

Types of Languages

- Based on the level
  - **low level**
    - binary (0s and 1s)
    - interacts with CPU
    - architecture dependent
    - e.g. Machine language and Assembly language
    - made up of opcodes and operands e.g. ADD A, B
  - **high level**
    - developer can write human understable code
    - compiler or interpreter convers the human understandable to machine (CPU) understandable (ASM)
    - highly portable
    - e.g. C++, Java, Python
- Based on build/execution process

- **compiled language**
  - compile: converting human understable to machine (CPU) understandable
  - compiler: program which does compilation
  - executable generated: program which contains only machine understandable instructions
  - native applications
  - always platform (OS) dependent
  - faster than interpreted program
  - requires compiler
  - the entire program gets converted into executable
  - if program contains error, compiler detects these error at compilation time
  - e.g. C, C++
- **interpreted language**
  - interpretation: which converts the human understandable to machine (CPU) understandable line by line
  - interpreter: program which does interpretation
  - no executable gets generated
  - if there is any error, it will get detected at the run time
  - program will be always platform (OS) independent
  - programs will be always slower than native applications
  - e.g. html/CSS, JS, bash script
- **mixed language**
  - shows behavior from both worlds (compiled as well as interpreted)
  - uses compiler as well as interpreter
  - e.g. Java, **Python**

Python

- General-purpose High-level language which shows behavior from both compiled as well as interpreted languages
- Python is a
  - Scripting language
  - OOP language
  - Functional programming language
  - Aspect oriented programming language

- Developed by Guido Rossum
- can be used for
    - console application
    - web application
    - ML application
    - GUI application
- Features
    - Highly readable
    - Portable (High-level)
    - Dynamically typed
    - Garbage collected
    - Free and Open-source
    - Predefined packages/libraries
- PyPI - Python Package Index
    - Alone Python is of no use.
    - Used with predefined Python packages.
    - Hosts 5,30,000+ packages (May 2024)
    - Popular packages
        - Data science, Machine Learning
        - Web programming
        - Application testing, Automation
        - Image processing

Python Versions

- By Van Guido Rossum - as successor to the ABC language.
- Development started in Dec 1989.
- First release: Feb 1991.
- Van Guido Rossum annouce his vacation from July 12, 2018 for Python project lead developer role
- Python Software Foundation (PSF) chosen a team of five members was developed in Jan 2019 to lead the project.
- Version 0.9.0 [Feb 1991] - Deprecated
    - Having features like classes with inheritance, exception handling, functions etc.

    - One of the major versions of python
- Version 1 [Jan 1994] - Deprecated
    - The major new features included in this release were the functional programming tools like labda, map, filter, reduce
    - The last version released was 1.6 in 2000
- Version 2 [Oct 2000] - Deprecated
    - Introduced features like list comprehension, garbage collection, generators etc.
    - Introduced its own license known as Python Software Foundation License (PSF)
    - The last version released was 2.7.16 in Mar 2019
- Version 3 [Dec 2008]
    - Python 3.0 is also called "Python 3000" or "Py3K"
    - It was designed to rectify fundamental design flaws in the language
    - Python 3.0 had an emphasis on removing duplicative constructs and modules

Python Installation

- To install python on ubuntu

```
sudo apt-get install python3 python3-pip
```

- to install python on centos/redhat

```
sudo yum install python3 python3-pip
```

- to install on Windows/Macos

```
https://www.python.org/downloads/
```

    - Download and follow installer instructions.
    - Add to "PATH" variable.
- Test on Python REPL/Shell
    - REPL = Read Evaluate Print Loop

```
python
```

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

IDE Installation

- **PyCharm**
  - "Community Edition"
  - https://www.jetbrains.com/pycharm/download/
- Spyder
- Jupiter Notebook
- Visual Studio Code

Python implementations

- Python is a specification and have various implementations.
- i.e. Its interpreter/compiler is implemented with different languages
  - CPython -- default/standard C implementation
  - JPython -- Java based
  - IronPython -- C# based
  - MicroPython -- Lightweight C implementation for micro-controllers.
  - PyPy - Python based

CPython

- CPython is the default and most widely used implementation of the language
- One of the reference implementation of Python language
- It is written in C and python
- Can be referred as both, compiler and interpreter
- It comes with a whole range of tools, libraries, and components

Python Virtual Machine

- Written in C
- Compiles the bytecode into machine language
- It emulates the machine or CPU
- Executes bytecodes similar to the way a CPU executes the machine instructions
- Python memory manager (Garbage collector)
- Computation Stacks

Hello World application

- demo01.py (create using any text editor)

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

- Python application does NOT require any entry point function
  - python is one of the scripting languages
  - the code starts execution from from top to bottom i.e. first line to last line.
- Execution

```
python demo01.py
```

Execution process

1. demo01.py --> Python compiler --> (Lexing + Parsing) Python byte code
2. Byte code --> Python Virtual Machine (Python interpreter) --> Machine language code
3. Machine code --> CPU

pyc file

- When a python file is imported using "import" statement, it is compiled and python byte code is created (.pyc file).
- However, when a python script is executed, bytecode creation process is done implicitly. No .pyc file is created.
- To explicitly create .pyc file, one can use py_compile module.

```
python -m py_compile demo.py
```

- This command will create demo.pyc in "**pycache**" folder.
- To execute the .pyc file,

```
python pycache/demo.pyc
```

- To compile all python files into current directory, use compileall module.

```
python -m compileall
```

Python Fundamentals

Statement

- unit of execution (the one which executes)
- semicolon (😉 is used to terminate a statement
- one statement per line does not require semicolon (😉
- multiple statements on one line must be separated by semicolon (😉
- Statement types
  - Single-line statements
    - Ends with newline character (optional 😉
  - Multi-line statement
    - Extends over multiple lines with the line continuation character (\ )
    - Line continuation is implied inside parentheses ( ), brackets [ ], and braces { }
  - Comments
    - For programmer's reference
    - Ignored by compiler/interpreter while execution of a program
    - In Python, we use the hash (#) symbol to start writing a comment
- Statement examples
  - assignment statement
    - e.g. PI = 3.14
    - e.g. num += 3

    - declaration statement
        - e.g. global var
    - function call
        - e.g. function()
    - control statements
        - e.g. if, for, while, match, etc.
    - comment statements
        - e.g. #comment

Block

- group of statements
- use space(s)/tab(s) [indentation] to create a block
- e.g. function, if, else, while, etc.

Error reporting

- If the code has any synctical error, the python compiler will not generate the byte codes [the syntactical errors will be detected at the time compilation]

```
print("hello 1")
print("hello 2"
# this code will generate SyntaxError
# even the first line will NOT get executed
```

- If the code has any run time error, the compilation will not detect any error and program will execute till the line where the error is detected

```
print("hello 1")
printf("hello 2")
# this code will generate NameError
# the first line will get executed and code will stoop on the line 2
```