PythonLife

Basic to Advance Python Programming

Core Python

Basics For Programming:

1. What is Programming?
2. What is Coding?
3. Compiler
4. Internal Structure of Compiler
5. Interpreter
6. Compiler vs Interpreter
7. Byte code
8. High Level Language
9. What is Low Level Language

What is Programming?

Programming is the process of creating a set of instructions that tell a computer how to perform a task.

Programming can be done using a variety of computer programming languages such as JavaScript, Python, and C++.

What is Coding?

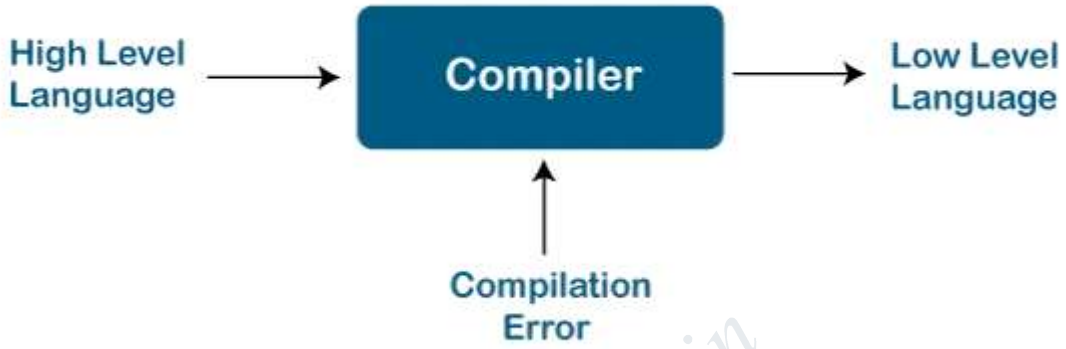
Coding is a list of step by step instructions that get computers to do what you want them to do.

Coding makes it possible for us to create computer software,games,apps and websites.Coders or Programmers behind everything we see and do on a computer.

Compiler:

The language processor that reads the complete source program written in high-level language as a whole in one go and translates it into an equivalent program in machine language is called a Compiler.

Example: C, C++, C#, Java.



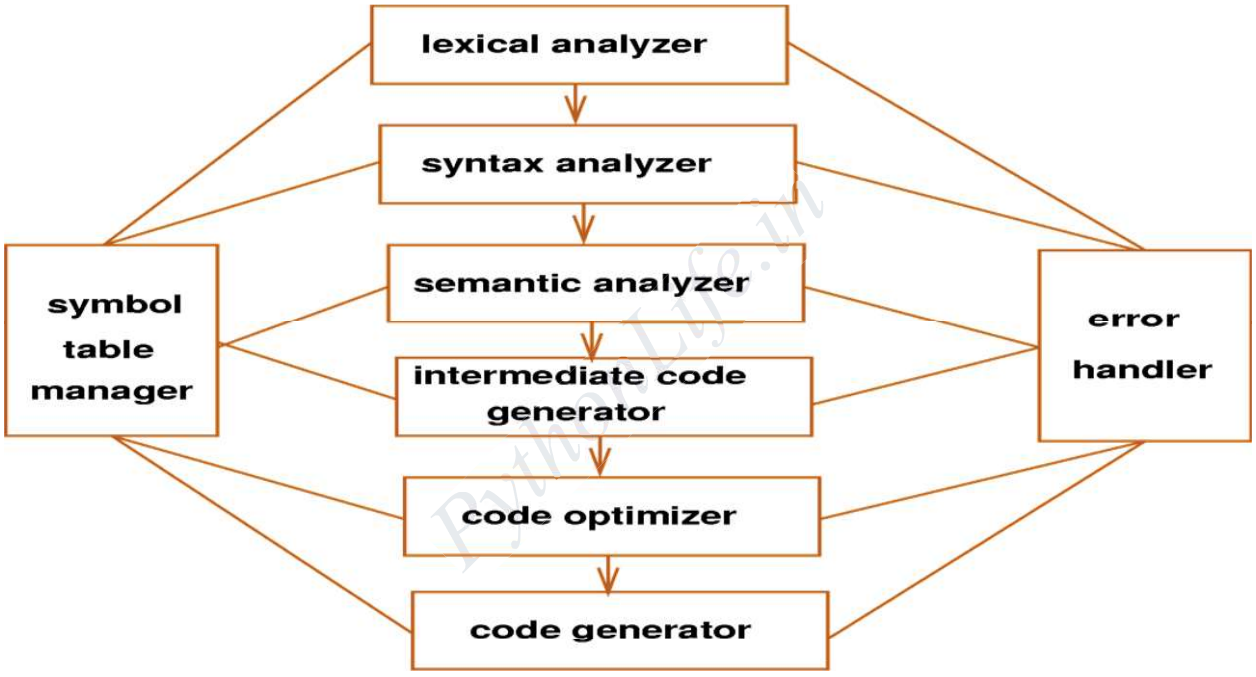
Internal Structure of Compiler:

Compiler operates in various phases each phase transforms the source program from one representation to another.

Every phase takes inputs from its previous stage and feeds its output to the next phase of the compiler.

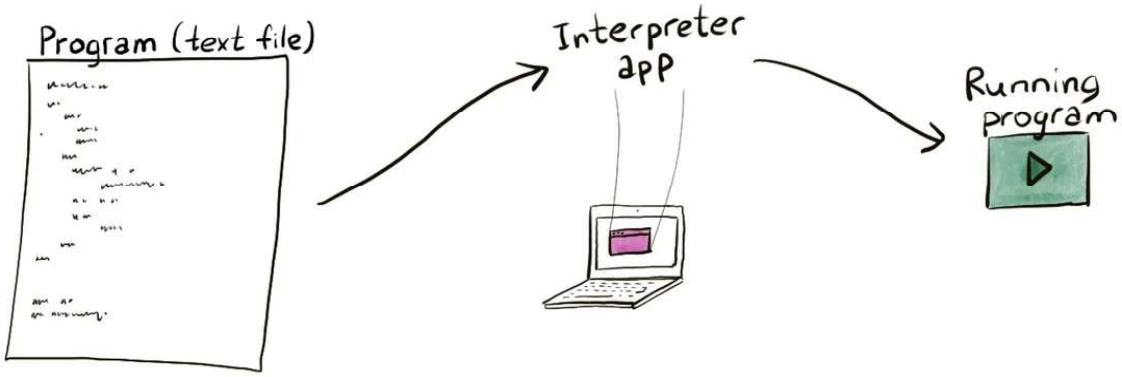
There are 6 phases in a compiler. Each of this phase help in converting the high-level language the machine code.

The phases of a compiler are: Lexical analysis Syntax analysis Semantic analysis Intermediate code generator Code optimizer, Code generator



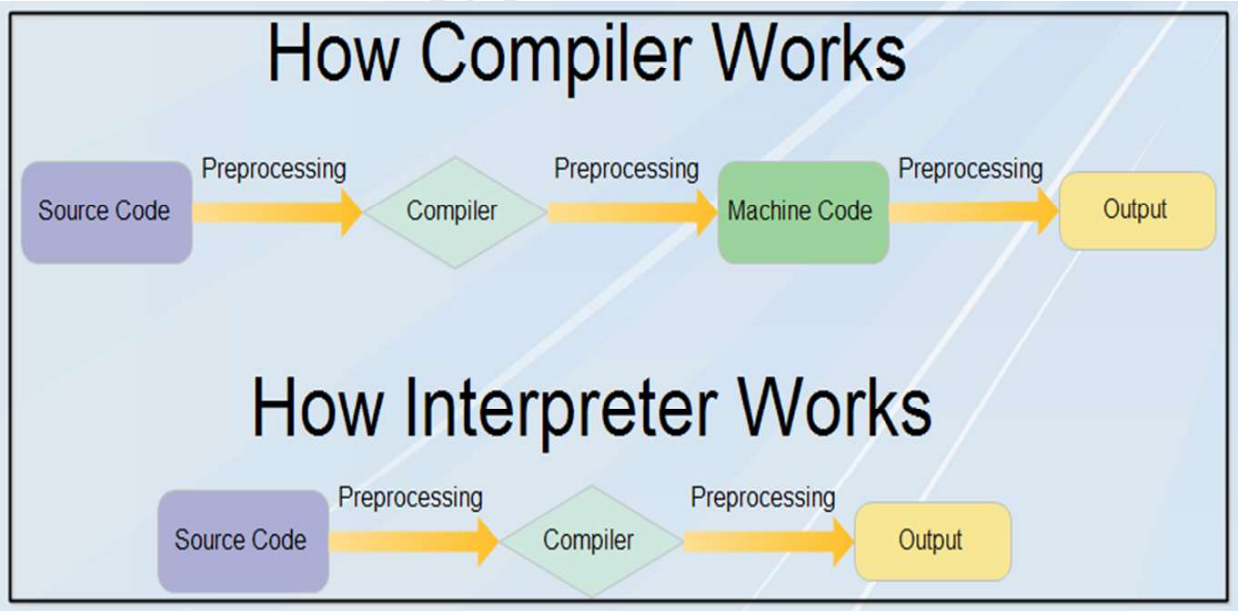
Interpreter:

The Python interpreter is a virtual machine, meaning that it is software that emulates a physical computer.
This particular virtual machine is a stackmachine: it manipulates several stacks perform its operations (as contrasted with a register machine, which writes to and reads from particular memory locations).



Compiler vs Interpreter:

Compiler	Interpreter
Compiler scans the entire program and translates the whole of it into machine code at once.	Interpreter translates just one statement of the program at a time into machine code
A compiler takes a lot of time to analyze the source code. However, the overall time taken to execute the process is much faster.	An interpreter takes very less time to analyze the source code. However, the overall time to execute the process is much slower.
A compiler always generates an intermediary object code. It will need further linking. Hence more memory is needed.	An interpreter does not generate an intermediary code. Hence, an interpreter is highly efficient in terms of its memory.
A compiler generates the error message only after it scans the complete program and hence debugging is relatively harder while working with a compiler. It is used in languages like C and C++ for example.	Keeps translating the program continuously till the first error is confronted. If any error is spotted, it stops working and hence debugging becomes easy. It is used in languages like Ruby and Python for example.



Byte code:

The byte code is a low-level platform independent representation of your source code. However ,it is not the binary machine code and cannot be run by the target machine directly.In fact,it is a set of instructions for a virtual machine which is called the Python Virtual Machine(PVM)

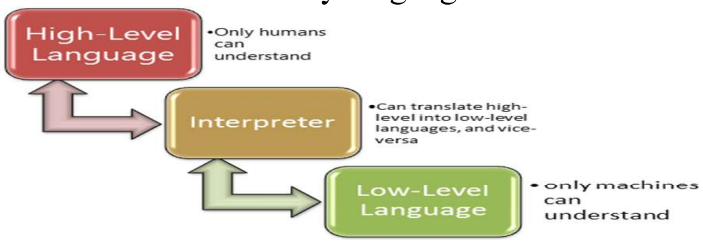
High Level Language:

A high-level is any programming language that enables development of a program in a much more user-friendly programming context and is generally independent of a computer's hardware architecture.

Low Level Language:

Low-level languages are languages that sit close to the computer's instruction set . An instruction set is the set of instructions that the processor understands.

Two types of low-level language are: machine code. assembly language.



Introduction to Python

- **What is Python?**
- **Why Python?**
- **History**
- **Features – Dynamic, Interpreted, Object oriented, Embeddable, Extensible, Large standard libraries, Free and Open source**
- **What is PVM?**
- **Python applications**
- **Python versions**
- **Python in real time industry**
- **Flavors of Python.**
- **Different Domains After Python.**
- **Python Module**
- **Python Package**
- **Why Python is More Popular**
- **Companies Offering Python Jobs**
- **Job Roles in Python**
- **Python Library**
- **Python Framework**
- **After Python**
- **Python Syntax compared to other programming languages**

What is Python ?

- Python is an interpreted, object-oriented, high-level programming language with dynamic semantics.
- Its high-level built in data structures, combined with dynamic typing and dynamic binding, make it very attractive for Rapid Application Development, as well as for use as a scripting or glue language to connect existing components together.

- Python's simple, easy to learn syntax emphasizes readability and therefore reduces the cost of program maintenance.
- Python supports modules and packages, which encourages program modularity and code reuse.

Why Python?

REASONS WHY PYTHON IS ONE OF THE BEST PROGRAMMING LANGUAGES:

- Python is Easy to Learn and Use
- Python is Handy for Web Development Purposes
- The Language is Extensively used in Data Science
- Has Multiple Libraries and Frameworks
- Python can be used in ML tool
- Flexibility and Reliability
- Python Automates Tasks

Python History:

- Python laid its foundation in the late 1980s.
- The implementation of Python was started in December 1989 by Guido Van Rossum at CWI in Netherland.
- In February 1991, Guido Van Rossum published the code labeled version 0.9.0 to alt.sources.
- In 1994, Python 1.0 was released with new features like lambda, map, filter, and reduce.
- Python 2.0 added new features such as list comprehensions, garbage collection systems.
- On December 3, 2008, Python 3.0 (also called "Py3K") was released. It was designed to rectify the fundamental flaw of the language.
- ABC programming language is said to be the predecessor of Python language, which was capable of Exception Handling and interfacing with the Amoeba Operating System.
- The following programming languages influence Python:

  1.ABC language 2.Modula-3

Features of Python:

- Python is a general-purpose high-level programming language whose design philosophy emphasizes code readability.
- Python supports multiple programming paradigms, primarily but not limited to object oriented, imperative and, to a lesser extent, functional programming styles.
- Python aims to combine “remarkable power with very clear syntax", and its standard library is large and comprehensive.
- Its use of indentation for block delimiters is unusual among popular programming languages.

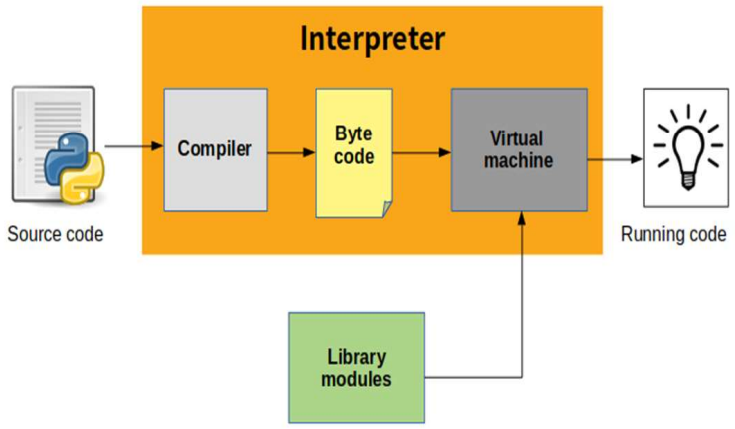
Features in Python :

There are many features in Python, some of which are discussed below as follows:

- Free and Open Source
- Easy to code
- Easy to Read
- Object-Oriented Language
- GUI Programming Support
- High-Level Language
- Extensible feature
- Easy to Debug
- Python is a Portable language
- Python is an Integrated language
- Interpreted Language
- Large Standard Library
- Dynamically Typed Language

What is PVM?

Python Virtual Machine (PVM) is a program which provides programming environment. The role of PVM is to convert the byte code instructions into machine code so the computer can execute those machine code instructions and display the output. Interpreter converts the byte code into machine code and sends that machine code to the computer processor for execution.



Python applications :

1. Web Applications
2. Enterprise Applications
3. Software Development Applications
4. Desktop GUI Applications
5. 3D CAD Applications
6. Scientific and Numeric Applications
7. Image Processing Applications
8. Audio or Video-based Applications
9. Console-based Applications
10. Business Applications

Python versions:

Python version hon Ve	RReleasedDate eleased
Python 1.0	January 1994
Python 1.5	December 31, 1997
Python 1.6	September 5, 2000

Python 2.0	October 16, 2000
Python 2.1	April 17, 2001
Python 2.2	December 21, 2001
Python 2.3	July 29, 2003
Python 2.4	November 30, 2004
Python 2.5	September 19, 2006
Python 2.6	October 1, 2008
Python 2.7	July 3, 2010
Python 3.0	December 3, 2008
Python 3.1 **Python version**	June 27, 2009 Released date
Python 3.2	February 20, 2011
Python 3.3	September 29, 2012
Python 3.4	March 16, 2014
Python 3.5	September 13, 2015
Python 3.6	December 23, 2016
Python 3.7	June 27, 2018
Python 3.8	October 14, 2019

Flavors of Python

Flavors of Python simply refers to the different Python compilers. These flavors are useful to integrate various programming languages into Python. Let us look at some of these flavors :

1. **CPython** : CPython is the Python compiler implemented in C programming language. In this, Python code is internally converted into the **byte code** using standard C functions. Additionally, it is possible to run and execute programs written in C/C++ using CPython compiler.
2. **Jython** : Earlier known as **JPython**. Jython is an implementation of the Python programming language designed to run on the Java platform. Jython is extremely useful because it provides the productivity features of a mature scripting language while running on a JVM.
3. **PyPy** : This is the implementation using Python language. PyPy often runs faster than CPython because PyPy is a just-in-time compiler while CPython is an interpreter.
4. **IronPython** : IronPython is an open-source implementation of the Python programming language which is tightly integrated with the .NET Framework.
5. **RubyPython** : RubyPython is a bridge between the Ruby and Python interpreters. It embeds a Python interpreter in the Ruby application's process using FFI (Foreign Function Interface).
6. **Pythonxy** : Python(x,y) is a free scientific and engineering development software for numerical computations, data analysis and data visualization based on Python.
7. **StacklessPython** : Stackless Python is a Python programming language interpreter. In practice, Stackless Python uses the C stack, but the stack is cleared between function calls
8. **AnacondaPython** : Anaconda is a free and open-source distribution of the Python and R programming languages for scientific computing, that aims to simplify package management and deployment. Package versions are managed by the package management system conda.

Different Domains After Python:

- Web Development
- Testing
- Web Scraping
- Data Analysis
- Computer Graphics
- Machine Learning
- Big Data
- Internet of Things
- Data Science
- Deep Learning
- Artificial intelligence

Python Module :

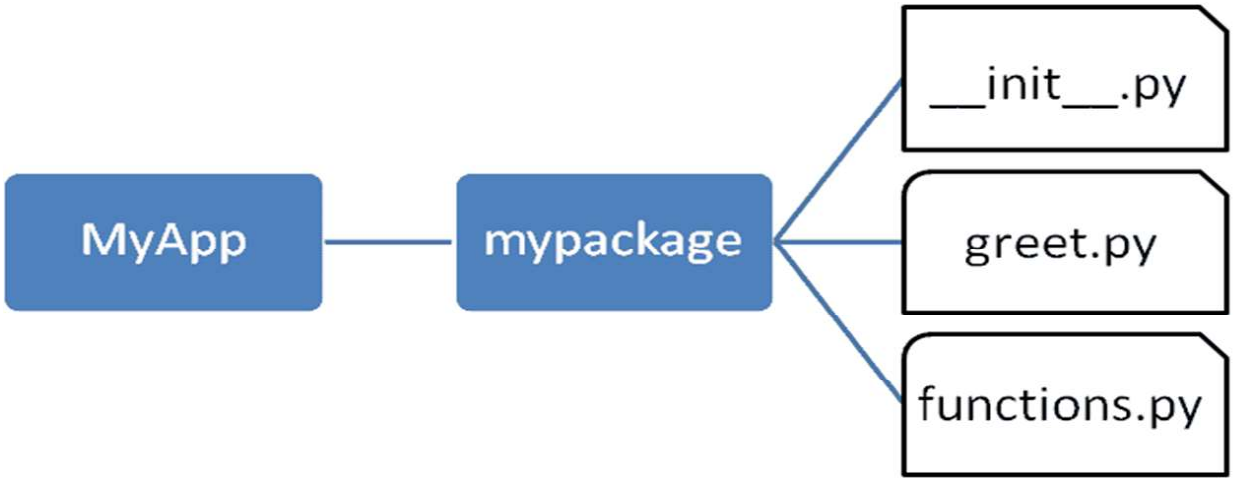
1. Python modules are nothing but files that consist of different statements and functions defined inside.
2. A module can define functions, classes, and variables. Modules help in organizing the code making it easier to use and understand. Modules provide reusability of the code.
3. Any file with extension .py can be referred as a module and the functions defined inside the module can be used in another program by simply using the import statement.
4. Suppose we need a function to find factorial in many programs. So, instead of defining a function to find the factorial in each program, what we can do is create a module with a function to find factorial and use that function in every program by simply importing the module.

Python Package:

A python package creates a hierarchical directory structure with numerous modules and sub-packages to give an application development environment. They are simply a collection of modules and sub-packages.

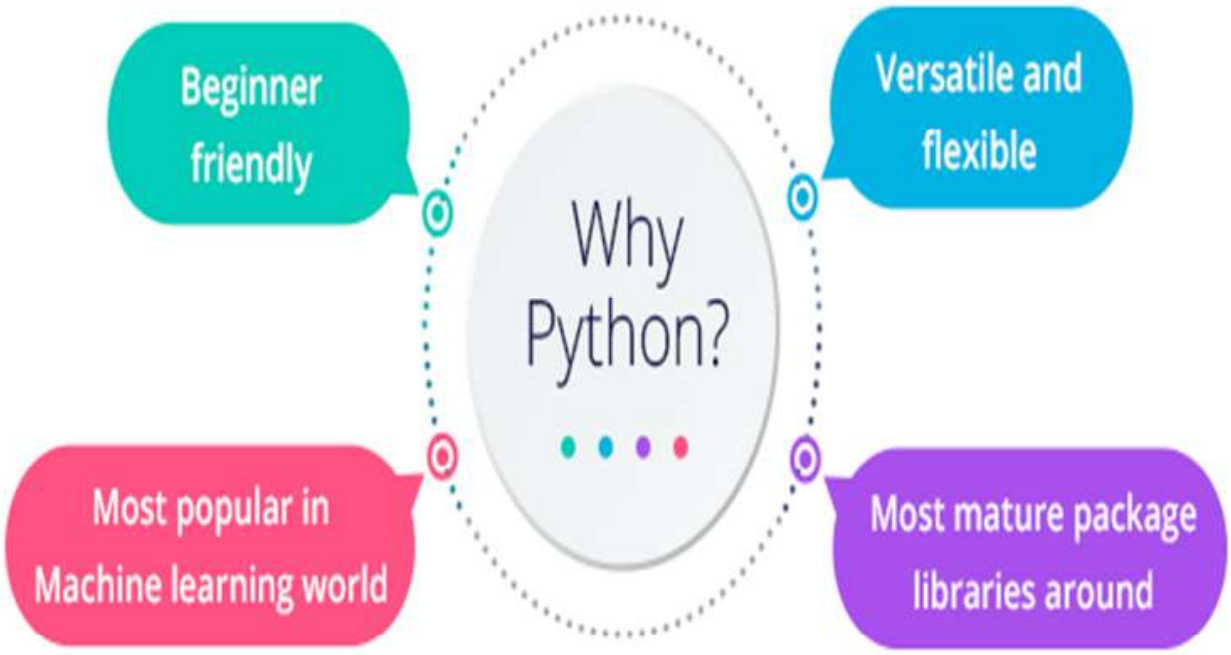
Importance of Python Packages :

- When working on a large or complex project, we frequently wind up with multiple modules. Using, maintaining, and organising so many modules is challenging.
- Fortunately, Python delivers us a simple solution in the form of packages. Using packages, we can easily group, organise, and use multiple modules.
- Packages also allow us to access all functions from all modules in a package with just one import statement.



Why Python is More Popular:

- Python is transparent.
- Python is open source.
- Python provides great support for developers.
- Python has a style guide for its code.
- Python reduces development time.
- Easy to learn and use.
- Supports multiple programming paradigms.
- Hundreds of Python and Frameworks Available Online.
- Growing Market Demand.



Companies Offering Python Jobs :

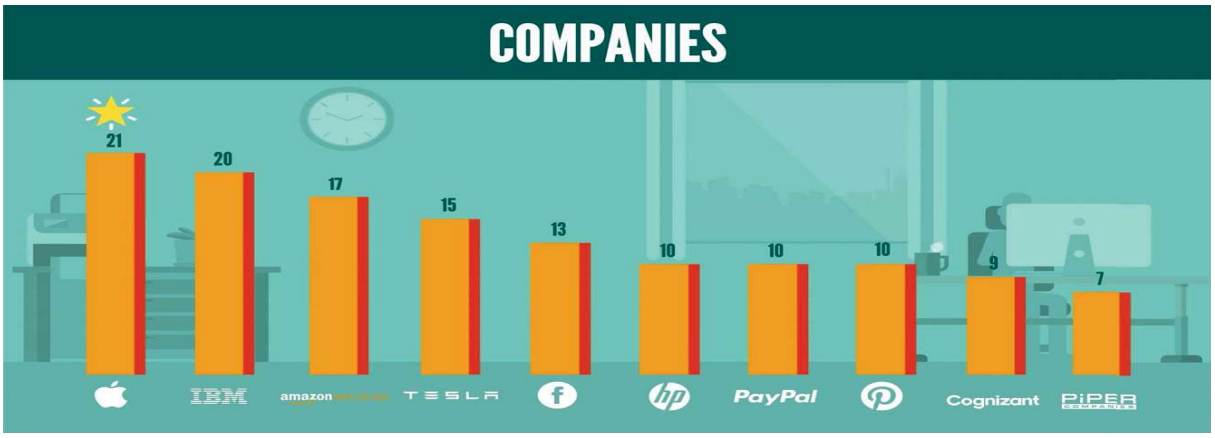
Python is a popular programming language among both smaller and larger companies – below are a few of the many big names which use the Python language to develop their projects.

Google – Python is one of the few official Google server-side languages. The plan for Google's backend was to use Python whenever possible. Python's main aim was to ensure undemanding maintenance and fast delivery.

Spotify – this music streaming platform uses Python for data analysis and backend. Python was chosen because it accelerates the development process. The coding in Python makes the development pipeline faster.

Reddit – half a year after deployment, the whole site was rewritten into Python from the Lisp language. In this case, the variety of libraries is as important as the readability of the

code.



Job Roles in Python:

1.Software Engineer

2.Web Developer/Front-end Developer

3.Python Developer

4.DevOps Engineer

5.Research Analyst

6.Data Analyst

7.Consultant

8.Product Manager

9.Data Scientist

10.Machine Learning Engineer

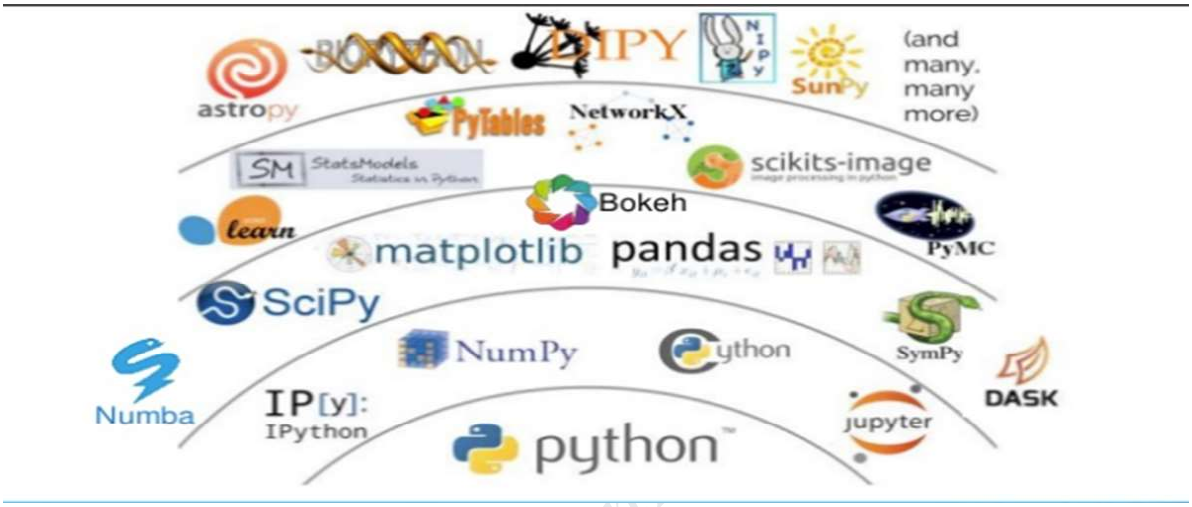
Python Library :

A Python library is a collection of related modules.

It contains bundles of code that can be used repeatedly in different programs.

It makes Python Programming simpler and convenient for the programmer.

As we don't need to write the same code again and again for different programs.



Important Python Libraries:

- TensorFlow
- Scikit-Learn
- Numpy
- Keras
- PyTorch
- LightGBM
- Eli5
- SciPy
- Theano
- Pandas
- Matplotlib
- Requests
- SQLAlchemy
- Pyglet

Python Frameworks:

Python is one of the most acceptable languages among web and application developers because of its strong emphasis on efficiency and readability.

There are numerous outstanding Python web frameworks, each with their own specialities and features.

Types of Python Framework:

Python majorly has three categories of frameworks and those are full-stack framework, micro-framework, and asynchronous framework. Now, let's understand what each category offers:

1. **Full Stack Framework**: As the name suggests this kind of framework provides a complete solution for web development like form generator, form validation, template layouts, etc. This type of framework can be utilized for any type of application. It is a little bit complex to use.
2. **Micro Framework**: It's a lightweight, easy-to-use framework and doesn't provide any extra features like data abstraction layer, form validation, etc. Developers have to put in lots of effort in adding code manually to get additional features and functionalities. This is useful for small applications.
3. **Asynchronous Framework**: This framework is gaining popularity in recent times and it uses asyncio library to work. This kind of framework mainly helps in running concurrent connections in huge amounts.

There are various frameworks of python like:

1. Bottle
2. Flask
3. Django
4. Web2py
5. AIOHTTP
6. CherryPy
7. Dash
8. Falcon
9. Growler
10. UvLoop
11. Pyramid
12. Sanic
13. CubicWeb

14. TurboGears
15. Hug
16. MorePath

After Python :

1. Teach Python to Beginners
2. Web Development with Python
3. Quality Assurance Engineer
4. Python Full Stack Developer
5. GIS Analyst
6. Data Scientist
7. Machine Learning Engineer
8. DevOps Engineer
9. Game Developer

Python Syntax compared to other programming languages :

Python vs PHP

	Python	**php**
Popularity	**Very popular**	**Very popular**
Frameworks	**A lot of frameworks**	**A few frameworks**
Learning	**Easy to learn**	**Harder to learn**

Python vs Java

	python	**java**
Learning	Easy to learn	Harder to learn
Cross-platform apps	Possible	Not Possible
Compatibility with operating systems	Possible	Possible
Network based apps	Not Possible	Possible

Python vs C#

	Python	C#
Simplicity	Possible	Not Possible
Script writing	In any environment	Only in IDE
Libraries	A lot of libraries	Few libraries
Performance	Low	High

Python vs Ruby

	python	**Ruby**
Approach to a problem	One solution	A lot of solutions
Community	Large	Large
Syntax	Very simple	More complex

Package After The Python Course
4LPA (Fresher)