



Programming with Python

Department of CE-AI/ CE-Big data

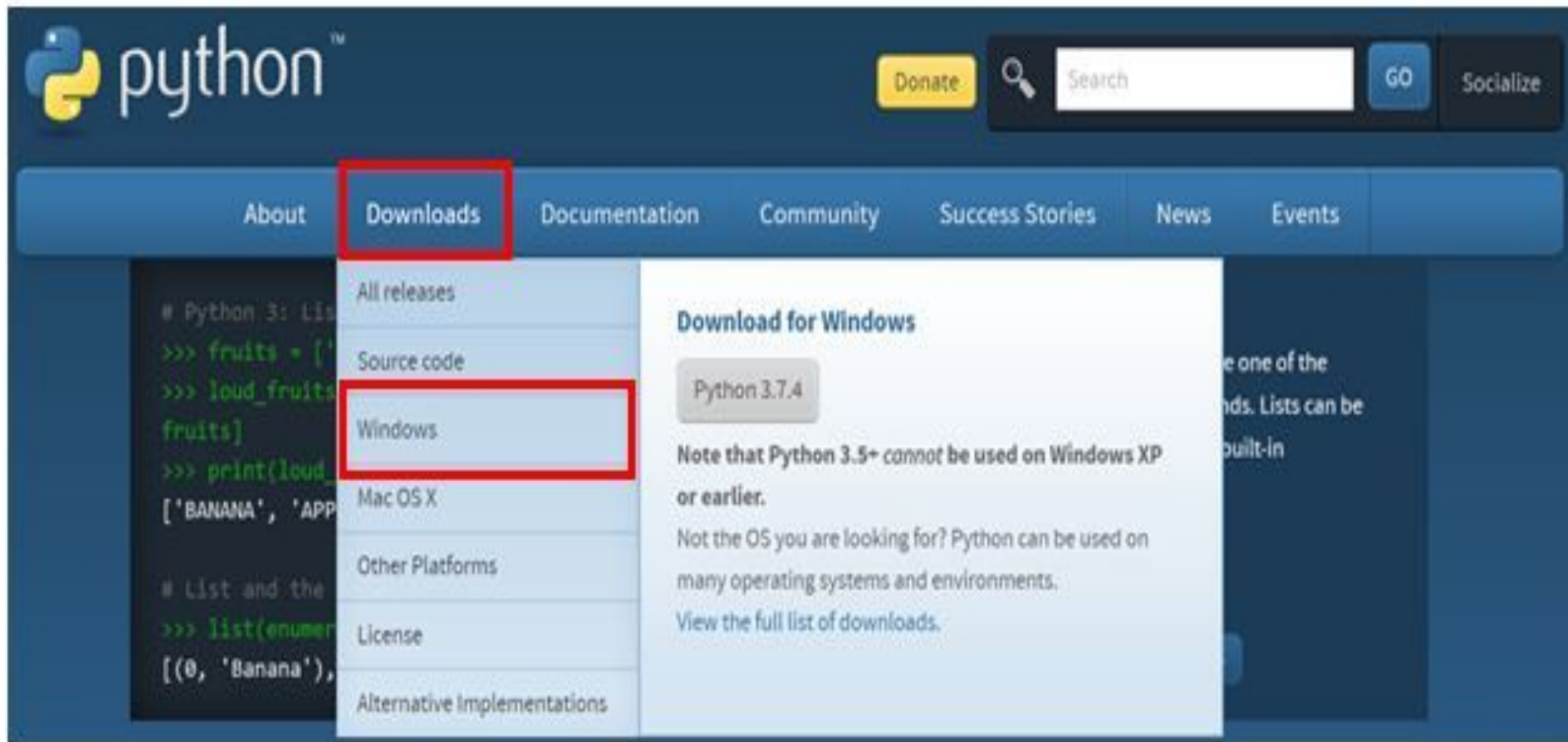
Unit no. 1 Introduction to Python

Installation and working with python

Step 1: To download Python, you need to visit www.python.org, which is the official Python website.



Click on the Downloads tab and then select the Windows option.



This will take you to the page where the different Python releases for Windows can be found. Since I am using a 64bit system, I'll select "Windows x86-64 executable installer".

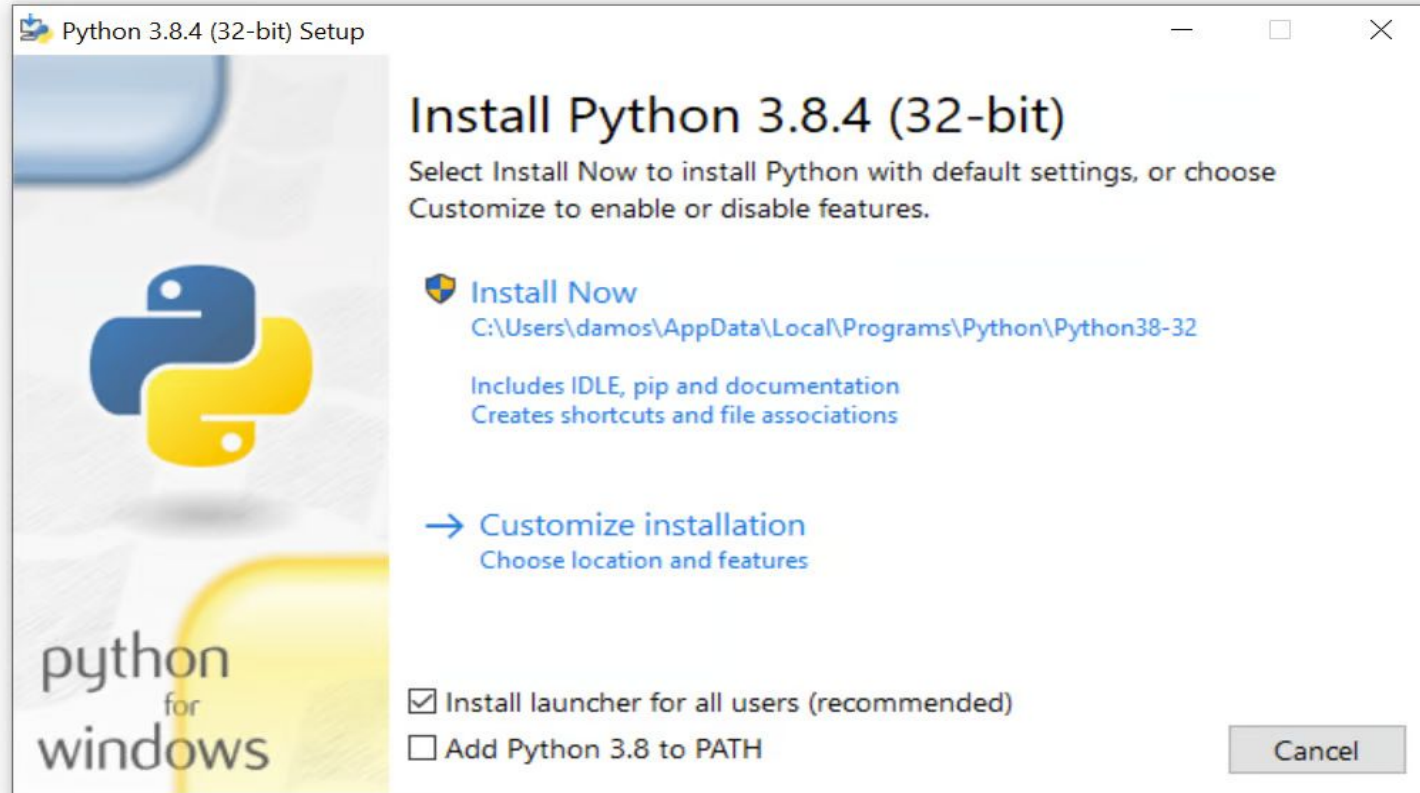
Stable Releases

- [Python 3.7.4 - July 8, 2019](#)

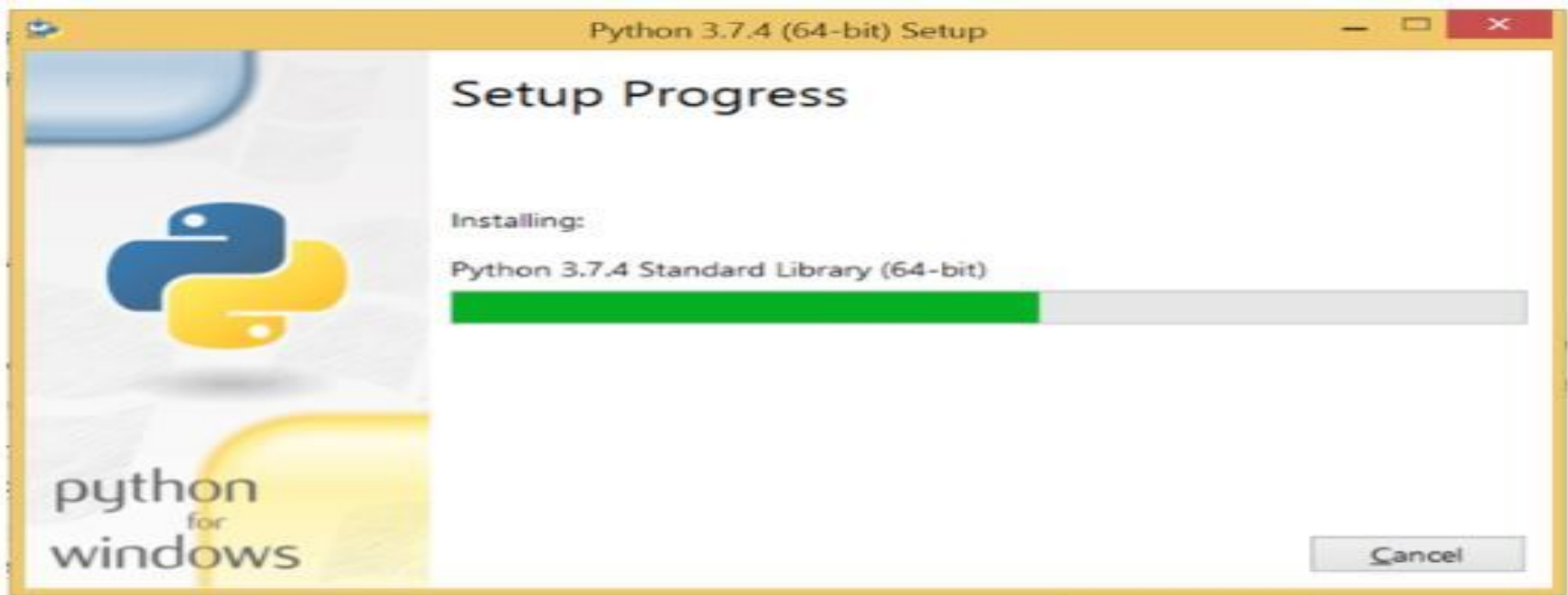
Note that Python 3.7.4 *cannot* be used on Windows XP or earlier.

- Download [Windows help file](#)
- Download [Windows x86-64 embeddable zip file](#)
- Download [Windows x86-64 executable installer](#)
- Download [Windows x86-64 web-based installer](#)
- Download [Windows x86 embeddable zip file](#)
- Download [Windows x86 executable installer](#)
- Download [Windows x86 web-based installer](#)

Click on Run, which will start the installation process.



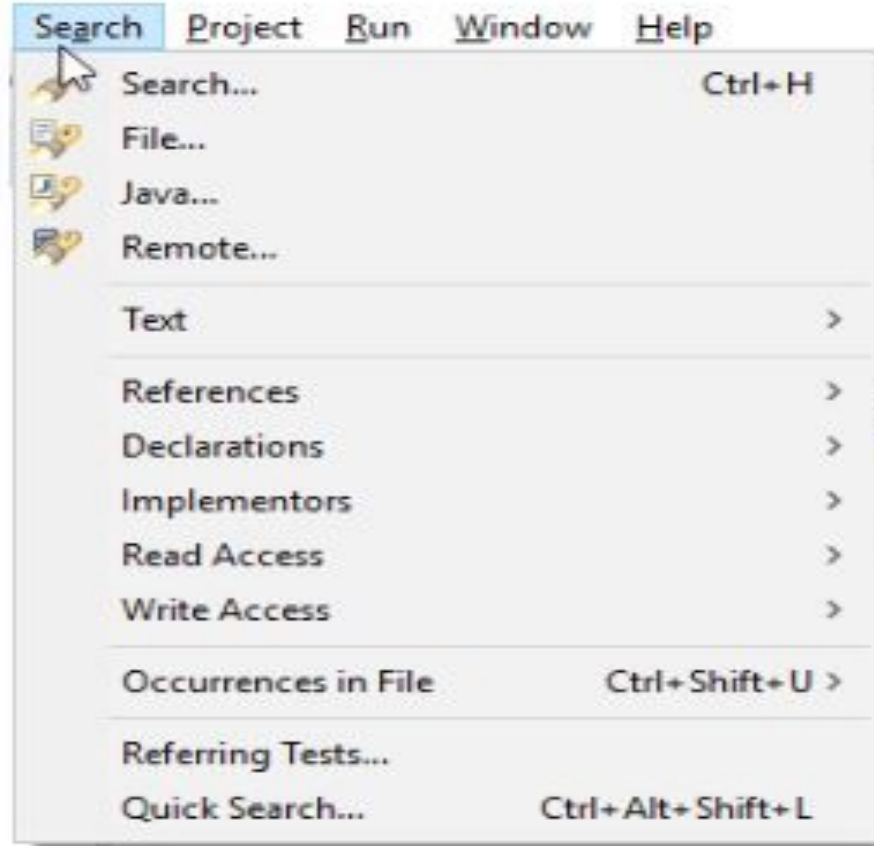
If you want to save the installation file in a different location, click on Customize installation; otherwise, continue with Install Now. Also, select the checkbox at the bottom to Add Python 3.7 to PATH.



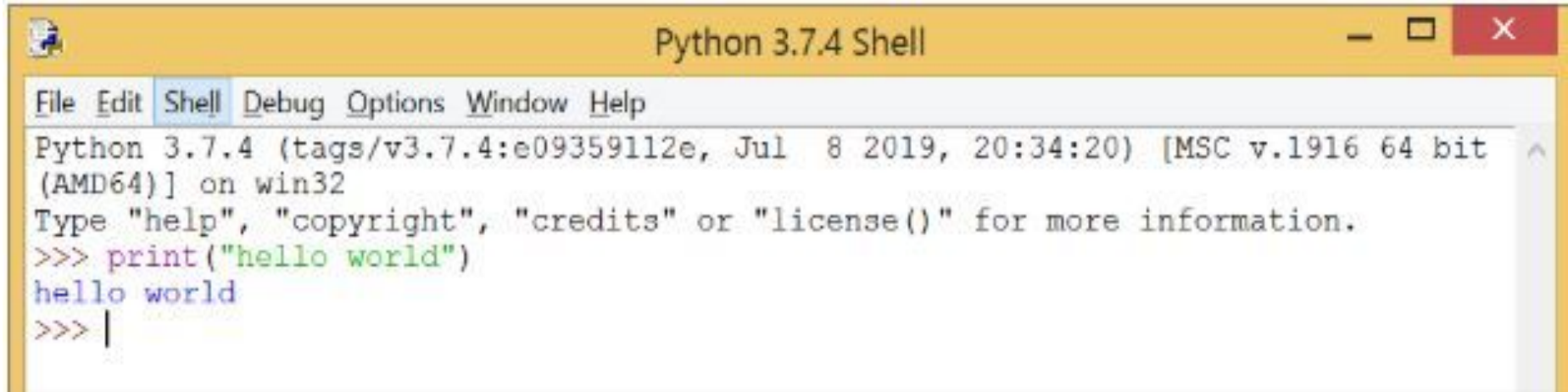
Once the installation is complete, the below pop-up box will appear:
Setup was successful.



Now that the installation is complete, you need to verify that everything is working fine. Go to Start and search for Python.



You can see Python 3.7 (64-bit) and IDLE. Let's open IDLE, which is the short form for Integrated Development Environment, and run a simple print statement.

A screenshot of the Python 3.7.4 Shell window. The window has a yellow title bar with the text "Python 3.7.4 Shell" and standard Windows window controls (minimize, maximize, close). Below the title bar is a menu bar with the following items: File, Edit, Shell, Debug, Options, Window, and Help. The main text area shows the following text:

```
Python 3.7.4 (tags/v3.7.4:e09359112e, Jul 8 2019, 20:34:20) [MSC v.1916 64 bit  
(AMD64)] on win32  
Type "help", "copyright", "credits" or "license()" for more information.  
>>> print("hello world")  
hello world  
>>> |
```

Introduction to Anaconda

Anaconda is an open-source distribution for python and R. It is used for **data science, machine learning, deep learning**, etc. With the availability of more than **300 libraries** for data science, it becomes fairly optimal for any programmer to work on anaconda for data science.

Anaconda helps in simplified package management and deployment. Anaconda comes with a wide variety of tools to easily collect data from various sources using various **machine learning and AI algorithms**. It helps in getting an easily manageable **environment setup** which can deploy any project with the click of a single button.

Introduction to jupyter notebook

Jupyter Notebook (sometimes called **IPython Notebook**) is a popular way to write and run Python code, especially for **data analysis, data science and machine learning**. Jupyter Notebooks are **easy-to-use** because they let you execute code and review the output quickly. This iterative process is central to **data analytics** and makes it easy to test hypotheses and record the results (just like a notebook).

Jupyter notebook is a **client-server application**. The application starts the server on local machine and opens the notebook interface in web browser where it can be edited and run from. The notebook is saved as ipynb file and can be exported as html, pdf and LaTeX files.

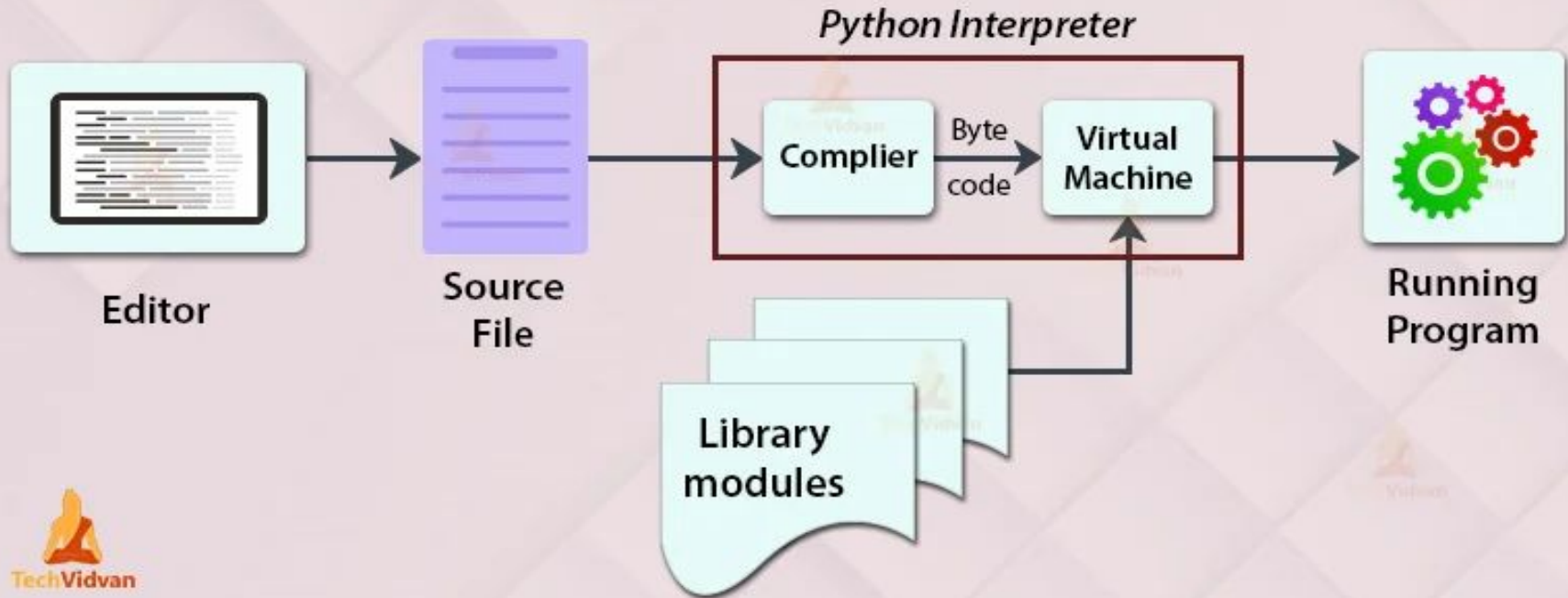
Google colab features

- Colab provides users free access to GPUs and TPUs, which can significantly speed up the training and inference of machine learning and deep learning models.
- Colab's interface is web-based, so installing any software on your local machine is unnecessary. The interface is also intuitive and user-friendly, making it easy to get started with coding.

- Colab allows multiple users to work on the same notebook simultaneously, making collaborating with team members easy. Colab also integrates with other Google services, such as Google Drive and GitHub, making it easy to share your work.
- Colab notebooks support markdown, which allows you to include formatted text, equations, and images alongside your code. This makes it easier to document your work and communicate your ideas.
- Colab comes pre-installed with many popular libraries and tools for machine learning and deep learning, such as TensorFlow and PyTorch. This saves time and eliminates the need to manually install and configure these tools.

Python Interpreter and its working,

How Python Interpreter Works?



We know that computers understand only machine code that comprises 1s and 0s. Since computer understands only machine code, it is imperative that we should convert any program into machine code before it is submitted to the computer for execution. For this purpose, we should take the help of a compiler. A compiler normally converts the program source code into machine code.

A Python compiler does the same task but in a slightly different manner. It converts the program source code into another code, called byte code. Each Python program statement is converted into a group of byte code instructions.

Python Virtual Machine (PVM) takes those byte codes converts those instructions into machine code so that the computer can execute those machine code instructions and display the final output. To carry out this conversion, PVM is equipped with an interpreter. The interpreter converts the byte code into machine code and sends that machine code to the computer processor for execution. Since interpreter is playing the main role, often the Python Virtual Machine is also called an interpreter.

Python Features

1) Easy to Learn and Use

Python is easy to learn as compared to other programming languages. Its syntax is straightforward and much the same as the English language. There is no use of the semicolon or curly-bracket, the indentation defines the code block. It is the recommended programming language for beginners.

2) Expressive Language

Python can perform complex tasks using a few lines of code. A simple example, the hello world program you simply type **print("Hello World")**. It will take only one line to execute, while Java or C takes multiple lines.

3) Interpreted Language

Python is an interpreted language; it means the Python program is executed one line at a time. The advantage of being interpreted language, it makes debugging easy and portable.

4) Cross-platform Language

Python can run equally on different platforms such as Windows, Linux, UNIX, and Macintosh, etc. So, we can say that Python is a portable language. It enables programmers to develop the software for several competing platforms by writing a program only once.

5) Free and Open Source

Python is freely available for everyone. It is freely available on its official website www.python.org. It has a large community across the world that is dedicatedly working towards make new python modules and functions. Anyone can contribute to the Python community. The open-source means, "Anyone can download its source code without paying any penny."

Object-Oriented Language

Python supports object-oriented language and concepts of classes and objects come into existence. It supports inheritance, polymorphism, and encapsulation, etc. The object-oriented procedure helps to programmer to write reusable code and develop applications in less code.

7) Extensible

It implies that other languages such as C/C++ can be used to compile the code and thus it can be used further in our Python code. It converts the program into byte code, and any platform can use that byte code.

8) Large Standard Library

It provides a vast range of libraries for the various fields such as machine learning, web developer, and also for the scripting. There are various machine learning libraries, such as Tensor flow, Pandas, Numpy, Keras, and Pytorch, etc. Django, flask, pyramids are the popular framework for Python web development.

Need of python

- Data Analysis
- Data Visualization
- Machine Learning
- Software Development
- Web Development
- Task Automation/Scripting
- Miscellaneous Python Uses

Role Of Python In Data Science

Data analysis and visualization are essential aspects of Data Science at the present time. Presently, Python has several libraries that make it easy to analyze and visualize data. Here are some of the most commonly used libraries for data analysis and visualization in Python:

- **Exploring Datasets**
- **Data Cleaning And Preprocessing**
- **Data Wrangling And Manipulation**
- **Generating Statistical Reports**
- **Graphical Representations**

Role of python in machine learning and DL

1. Python offers a rich library ecosystem

2. Python has a low entry barrier

3. It's amazingly flexible

4. It doesn't depend on any particular platform

5. Python is easy to read

6. It offers a variety of visualizations options for data scientists

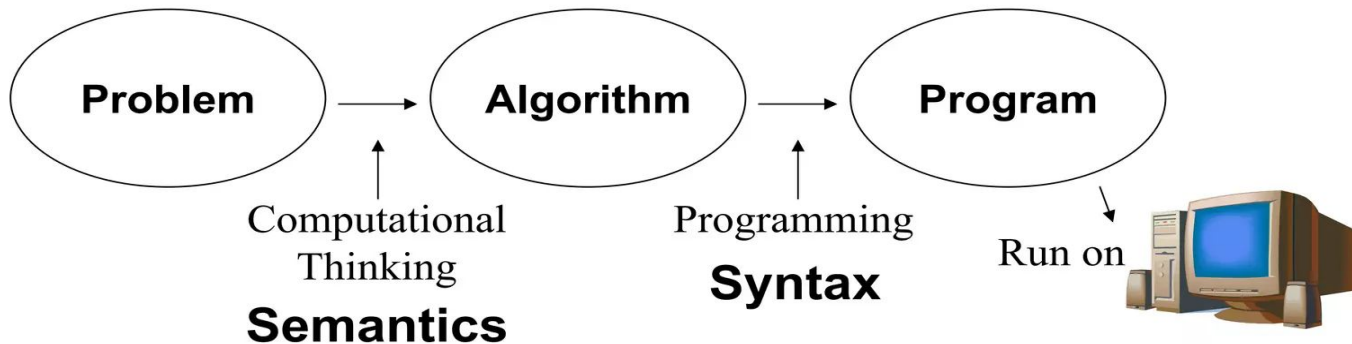
7. It's represented by a large community

8. Python's popularity keeps growing among scientists, professors, and large corporations

Semantics and Syntax

When writing programs, you have to take care of

- **Semantics** – Meaning of your program
- **Syntax** – Specifying your algorithm using a programming language



Semantics and Syntax

Just like communication with English:

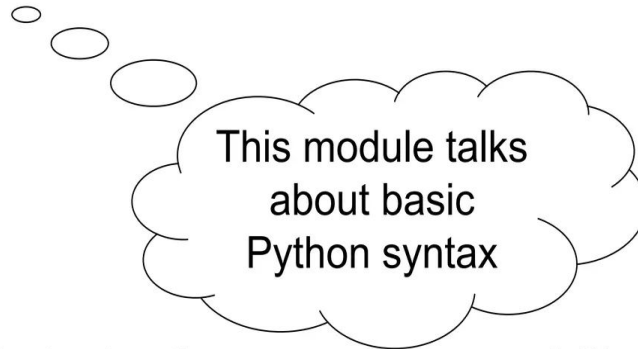
- The meaning (**semantics**) of the sentence, as well as
- The grammar (**syntax**) of the sentence, so that others can understand, e.g.,

✓ he has X he have

✓ we are X we is

Semantics and Syntax

- Different languages have different syntax; this applies to both spoken languages and computer programming languages
- To make sure the Python shell can understand your Python program, your program has to follow the **syntax** of the Python language



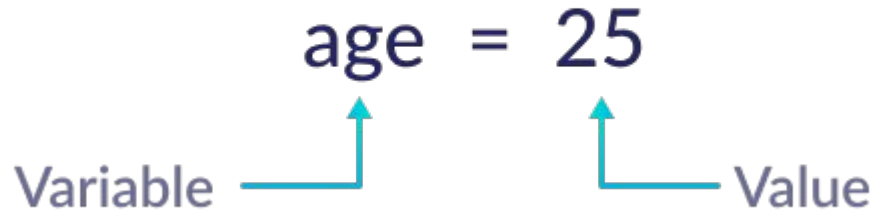
Python Variables

A variable is a container that is used to store values. For example,

Syntax-

```
age = 25
```

Assignment



The diagram illustrates the syntax of a variable assignment in Python. It shows the code `age = 25` with labels and arrows identifying its components. The word `age` is labeled as the **Variable**, the `=` symbol is the **Assignment** operator, and the number `25` is labeled as the **Value**. Blue arrows point from the labels to their respective parts in the code.

```
age = 25
```

Variable Value

Print Variables in Python

We can simply print a variable using the `print()` function. For example,

```
age = 25
```

```
# print the variable age
```

```
print(age) # 25
```

Note that if we use the quotation ' ' in the print statement, Python considers it as a string. For example,

```
age = 25
# variable with quotation
# prints the string
print('age')
# Output: age
```

Declare Variables Before Use

We cannot use a variable before defining them. For example,

```
age = 25
```

```
# use the name variable without defining it
```

```
# error
```

```
print(name)
```

```
print(age)
```

Output:

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

Rules of Naming Variables in Python

Here are the rules for naming variables:

- A variable name can consist of alphabets, digits, and an underscore.
- Variables cannot have special symbols like \$, @, #, etc.
- Variable names cannot begin with a number. For example, 1name.

By the way, we should always try to give meaningful variable names. This makes your code easier to read and understand.

Good, Bad and Illegal Variable Names

Illegal Variable	Bad Variable	Good Variable
date time	dati	dateTime
city-name	cn	city_name
20n	numb	number
s@lary	sal	salary

Immutable Variables

1. Numbers

Integers, floats, and complex numbers are immutable - their values cannot be changed after creation.

2. Strings

Strings are also immutable, meaning individual characters cannot be modified once the string is created.

3. Tuples

Tuples are ordered, immutable collections of values that cannot be changed after assignment.

Blocks in Python

1. Indentation or Whitespaces

Python uses indentation to define code blocks, rather than curly braces or other delimiters.

2. Scope

Variables defined within a block have local scope, while those outside have global scope.

3. Statements

Blocks can contain various statements like if, for, while, def, and class to control program flow.

In C

```
#include <stdio.h>
int main()
{ //Beginning of Block 1
int a;
printf("Enter the easiest programming language: ");
scanf("%d", &a);
if (a == "python")
{ //Beginning of Code Block 2
printf("Yes! You are right");
} //End of Code Block 2
else
{ //Beginning of Code Block 3
printf("Nope! You are wrong");
} //End of Code Block 3
return 0;
} //End of Block 1
```

In Python


```
a = input('Enter the easiest programming language: ')
```

```
if a == 'python':
```

```
    print('Yes! You are right')
```

```
else:
```

```
    print('Nope! You are wrong')
```



Additional space before print is used to indicate a new block of code. Here, both the print are indented 2 spaces.

Python uses Indentation

```
if a==1:
    print(a)
    if b==2:
        print(b)
print('end')
```



Indentation Tips

- Use 4 spaces per indentation level
- Spaces are the preferred indentation method
- Use spaces for indentation; not tabs
 - 1 tab == 1 indent level
- Python 3 explicitly disallows mixing the use of tabs and spaces for indentation

Comments

Comments are hints that we can add to our program to make our code easier to understand. The Python compiler ignores everything after the # symbol.

Let's take an example.

```
# declare a variable
```

```
name = 'John'
```

```
# print name
```

```
print(name)    # John
```

Python Multiline comments

We can use triple quotes (""" """) to achieve a similar effect. For Example:

```
'''This program takes an input from the user  
and prints it'''  
  
name = input('Enter your name: ')  
  
print(name)
```

Why Use Comments?

We should use comments for the following reasons:

- Comments make our code readable for future reference.
- Comments are used for debugging purposes.
- We can use comments for code collaboration as it helps peer developers to understand our code.

