Estd. 2000

1.1 Getting started with Python

General Guideline

Topics Covered

Day 1	Day 2	Day 3	Day 4
1.1 Getting started with Python • 1.1.1 Introduction to programming and coding • 1.1.2 Why choose Python • 1.1.3 Scope of Python • 1.1.4 Python History • 1.1.5 Python Features	1.1 Getting started with Python • 1.1.6 Advantages of Python • 1.1.7 Disadvantages of Python • 1.1.8 Applications of Python • 1.1.9 Different Flavors of Python • 1.1.10 Different Python Frameworks • 1.1.11 Python in contrast with other programming languages	1.2 Python Installation Guide • 1.2.1 Introduction to python IDE – IDLE • 1.2.2 Setting Up Your Environment • 1.2.3 Installation of Python and Anaconda Navigator • 1.2.4 Quick Tour of Jupyter Notebook • 1.2.5 Python vs. IPython • 1.2.6 Online compilation support • 1.2.7 Running python script using command prompt	1.3 Basics of Python • 1.3.1 Python keywords • 1.3.2 Python Statement and Comments • 1.3.3 Python Literals • 1.3.4 Data Types • 1.3.5 Variables • 1.3.6 type (), dir (),ID command

Topics Covered

Day 5	Day 6		
1.3 Basics of Python • 1.3.7 Type conversion: implicit and explicit • 1.3.8 Basic I/O Operations: input (), print **()**	1.3 Basics of Python • 1.3.9 Operators • 1.3.10 Precedence and associativity • 1.3.11 Python 2 vs Python 3		

Introduction

Python is a Popular

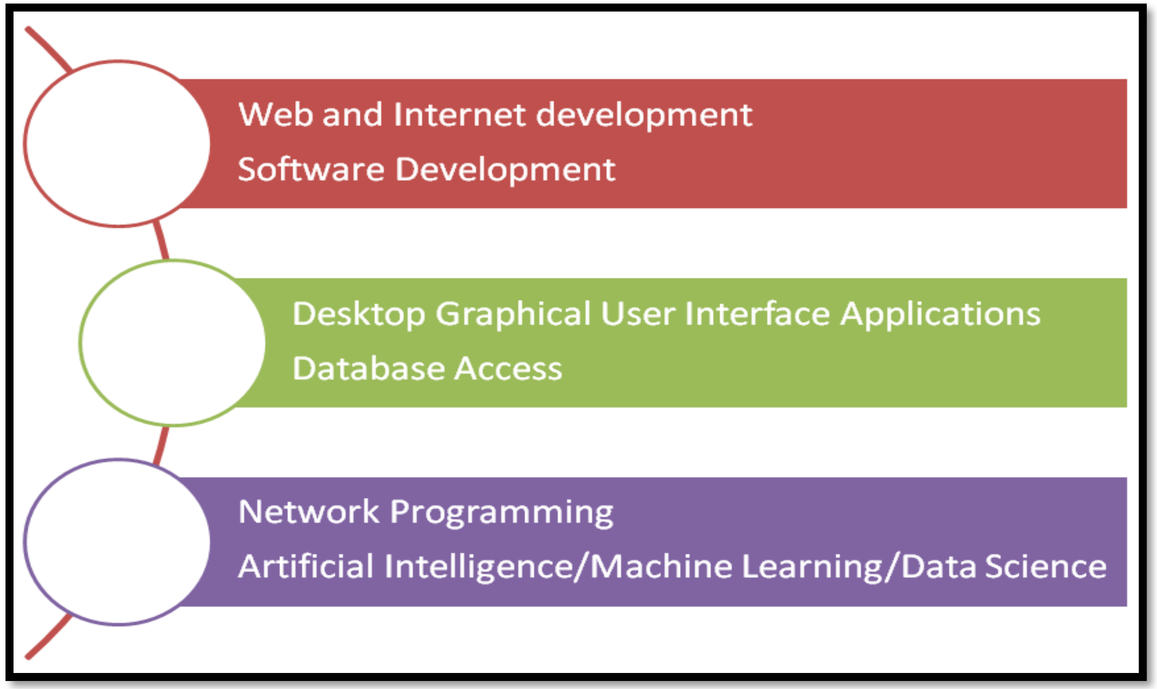
Programming language

?

https://www.wordclouds.com/

Some Areas where it is Popular

- ❑Application Development
- ❑Web Development
- ❑Artificial Intelligence
- ❑Data Science


Web and Internet development
Software Development
Desktop Graphical User Interface Applications
Database Access
Network Programming
Artificial Intelligence/Machine Learning/Data Science

Why choose Python : Lets Discuss More

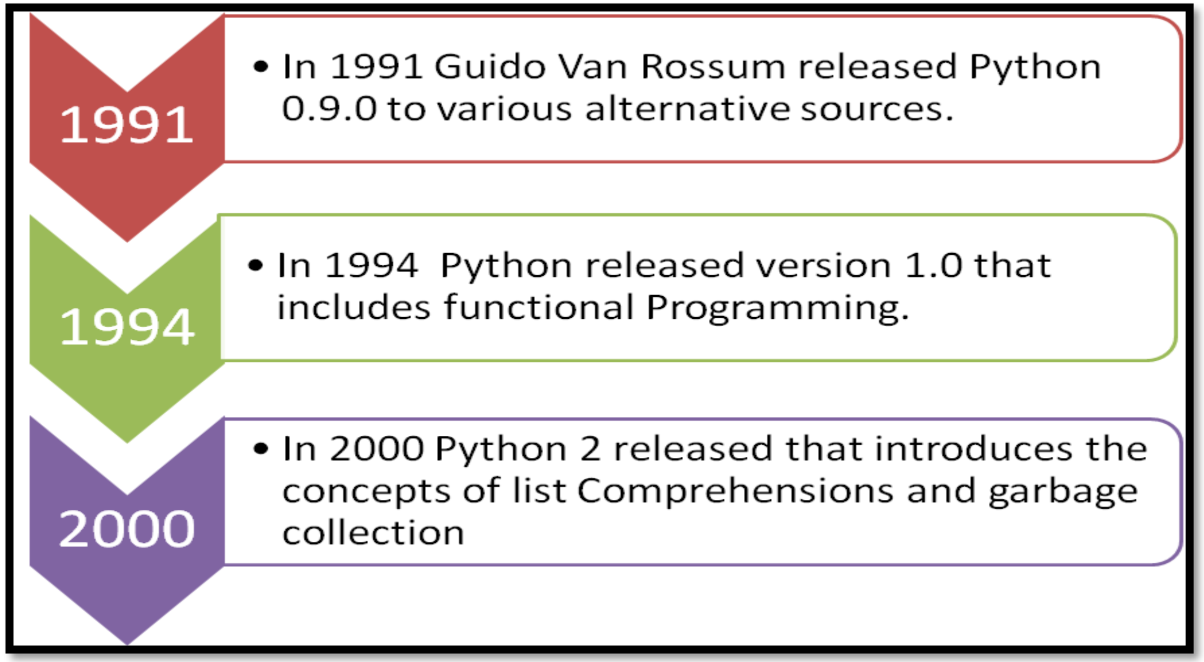
- ❑Simple and Easier to learn
- ❑Good Readability
- ❑Free and Open Source
- ❑Python is a platform-independent language
- ❑High Level and Interpreted language
- ❑Extensive libraries: Python has a vast number of libraries.

Scope of Python

- Python Language provides promising and rewarding career in the IT industry
- Various Job roles advanced in Python with high paying jobs:
  - Research Analyst
  - DevOps Engineer
  - Python Developer
  - Data Analyst
  - Software Developer
  - Game Developer
  - Web Scrapper

Python History

Python was written in the late 1980s by Guido van Rossum at Centrum Wiskunde & Informatica (CWI)

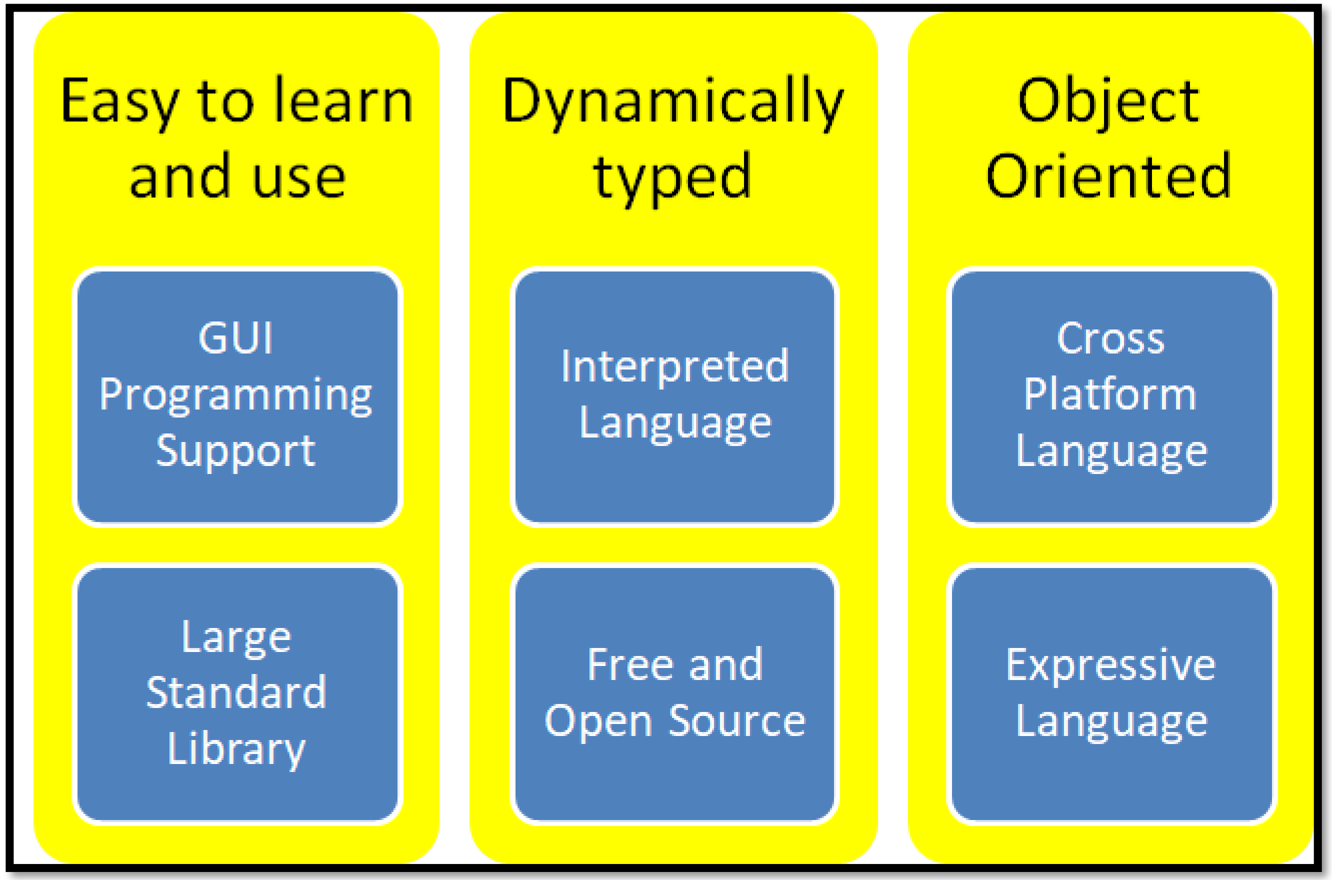


1991
- In 1991 Guido Van Rossum released Python 0.9.0 to various alternative sources.

1994
- In 1994 Python released version 1.0 that includes functional Programming.

2000
- In 2000 Python 2 released that introduces the concepts of list Comprehensions and garbage collection

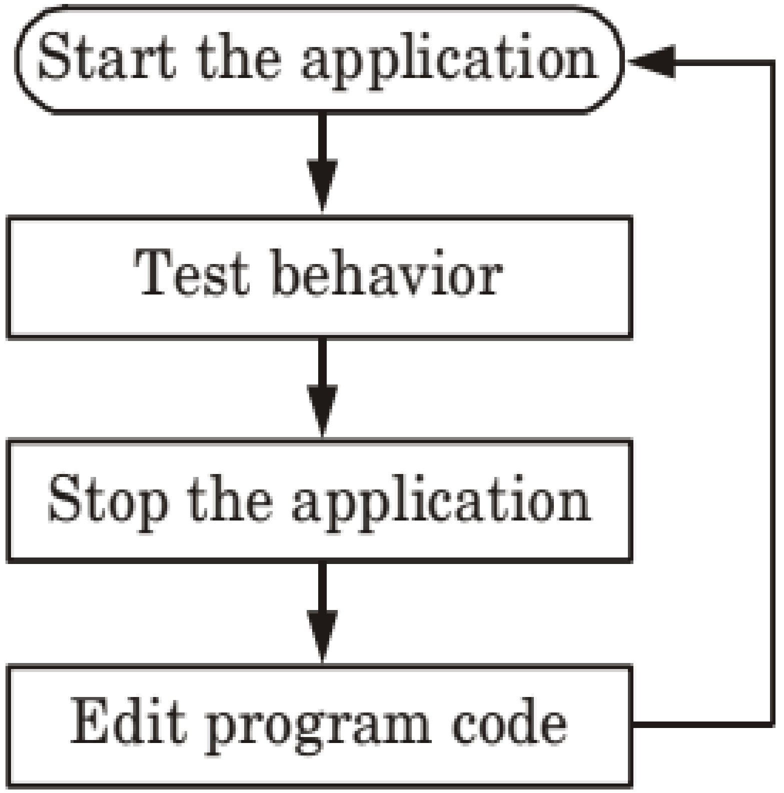
Python Features



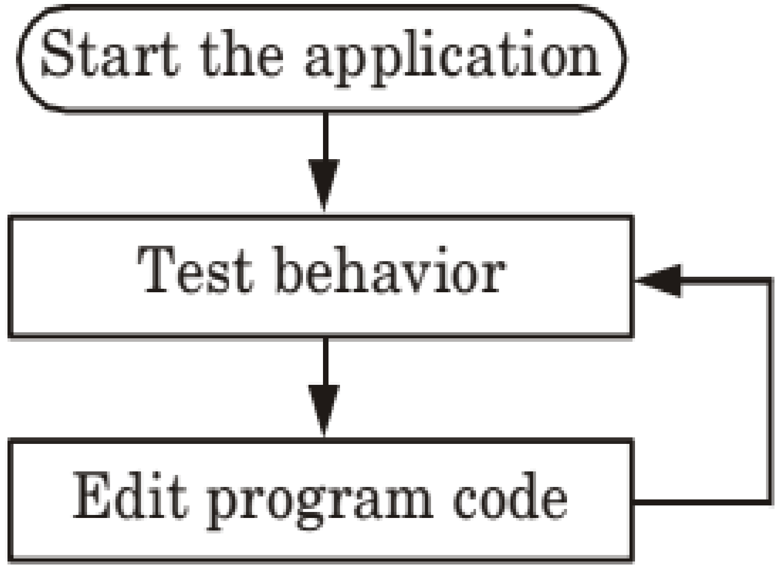
Python Programming Cycle

- Python’s programming cycle is dramatically shorter than that of traditional programming cycle.
- In Python, there are no compile or link steps.
- Python programs simply import modules at runtime and use the objects they contain. Because of this, Python programs run immediately after changes are made.
- In cases where dynamic module reloading can be used, it is even possible to change and reload parts of a running program without stopping it at all.

Python Programming Cycle



(a) Python's programming cycle



(b) Python's programming cycle with module reloading

Review Questions

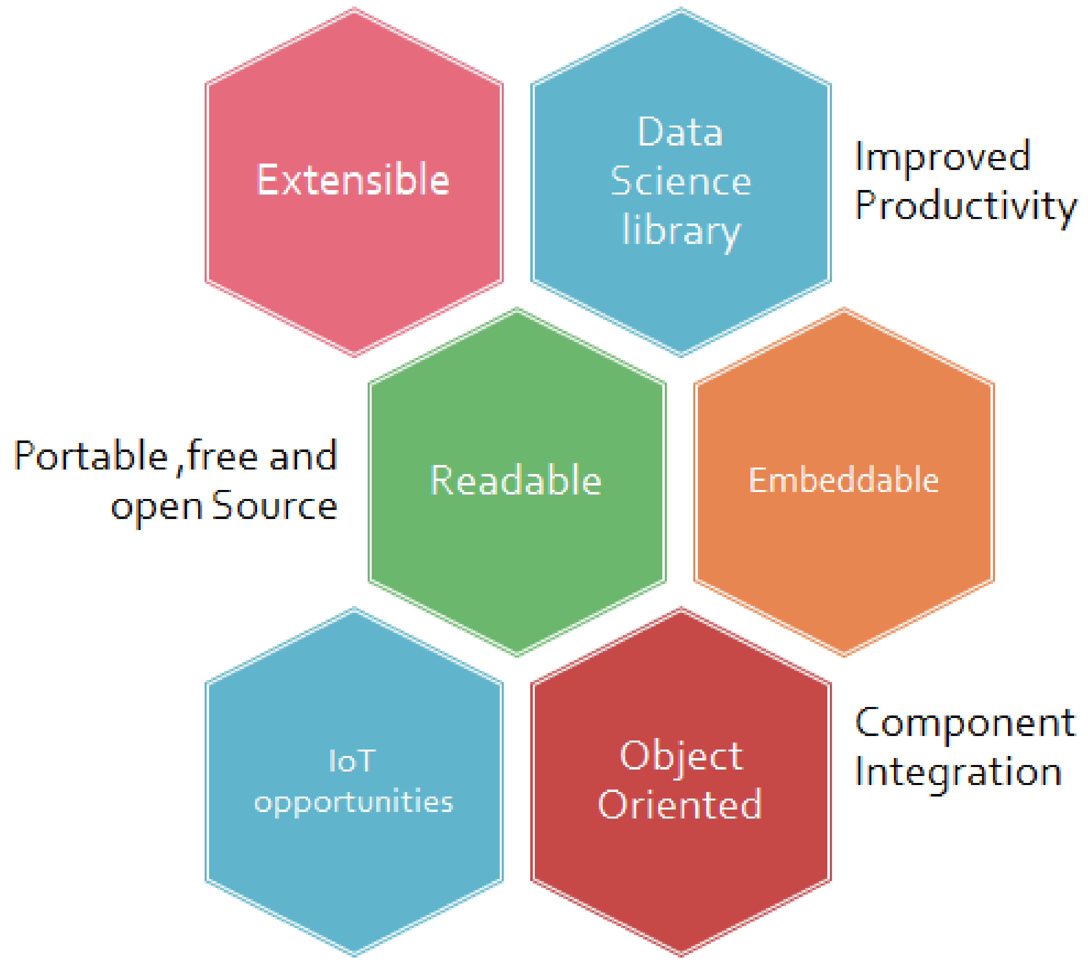
- In which year was the Python language developed?
  - 1995
  - 1972
  - 1981
  - 1989

- Who developed the Python language?
  - Zim Den
  - Guido van Rossum
  - Niene Stom
  - Wick van Rossum

Review Questions

- How many keywords are there in python 3.7?
  - 32
  - 33
  - 35
  - 30
- Which one of the following is the correct extension of the Python file?
  - .py
  - .python
  - .p
  - None of these

Advantages of Python



Contd..

Write a Program using any language to print “ HELLO WORLD “.

Java Program :

```
public class Hello
        {
            public static void main(String argv[])
          {

              System.out.println(“Hello, World!”);

                  }
         }
```

C++ Program :

```
#include <iostream><br>
   int main()
 {
      cout << "Hello World" << endl;
      return 0;
 }
```

Contd..

In Python

print ("Hello World")

Disadvantages of Python

- Python code is executed line by line since Python is interpreted as **slower in runtime** than other programming languages like C++, Java, and PHP.
- Python takes a **lot of memory** due to the flexibility of the data types, so it is not a desirable choice for memory-intensive tasks.
- Although Python serves as an excellent server-side programming language, it is **less commonly used to build intelligent phone-based Applications**.
- Python is **rarely used in Enterprise development** because Python has some limitations with Database Access compared to primarily other used technologies like JDBC (Java Database Connectivity and ODBC (Open Database Connectivity) as Python Language Database layers are underdeveloped.

Your Future….!!!

Top Companies Using Python
NETFLIX
JPMorganChase
NASA
YouTube
pandora
Bitbucket
shutterstock
PayPal
facebook
Prezi
Vine
yelp
Bank of America
UBER
Google
Udemy
asana
redhat
freelancer
Quora
twilio
IBM
@mailgun
moz://a
amazon
Expedia
Spotify
DISQUS
SurveyMonkey
glassdoor
reddit
hike
YAHOO!
9GAG
Massachusetts Institute of Technology
NOKIA
SendGrid
redis
the ONION
trivago

Applications of Python

- Web and Internet Development
- Game Development
- Desktop GUI Applications
- Artificial Intelligence and Machine Learning
- Data Science and Data Visualization
- Web Scraping Applications
- Desktop Applications
- Business Applications
- Image Processing and Computer Graphics
- Language Development
- Popular Applications Built on Python

Different Flavors of Python

- ❑Cpython
- ❑Jpython
- ❑Active Python
- ❑Anaconda Python
- ❑PyPy
- ❑Win Python
- ❑Python Portable

Different Python Frameworks

- ❑Full-Stack Frameworks
- ❑Non-Full Stack Frameworks
- ❑Asynchronous Frameworks

CherryPy
A Minimalist Python Web Framework
django
Flask
web development, one drop at a time
pyramid
Tornado
Real Time Web Server

Full Stack Frameworks
- Also known as Enterprise framework
- Gives full support to Developers
- Basic Functionality URL Routing and Form Validation

Non full Stack Frameworks
- Also known as Micro framework
- Micro Framework usually do one thing at a time and do that thing very well.

Asynchronous Frameworks
- It is a Framework for handling a huge amount of Concurrent Connections
- Asynchronous I/O or asyncio is a library to write concurrent code using the async/await syntax.

Review Questions

- How to output the string “May the odds favor you” in Python?
  - print(“May the odds favor you”)
  - echo(“May the odds favor you”)
  - System.out(“May the odds favor you”)
  - printf(“May the odds favor you”)

- In which year was the Python 3.0 version developed?
  - 2005
  - 2000
  - 2010
  - 2008

Review Questions

- Python is often described as a:
  - Batteries excluded language
  - Gear included language
  - Batteries included language
  - Gear excluded language

- What do we use to define a block of code in Python language?
  - Indentation
  - Key
  - Brackets
  - None of these

1.2 Python Installation Guide

Before you start, you will need Python on your computer.

Installing Python on your computer is the first step to becoming a Python programmer.

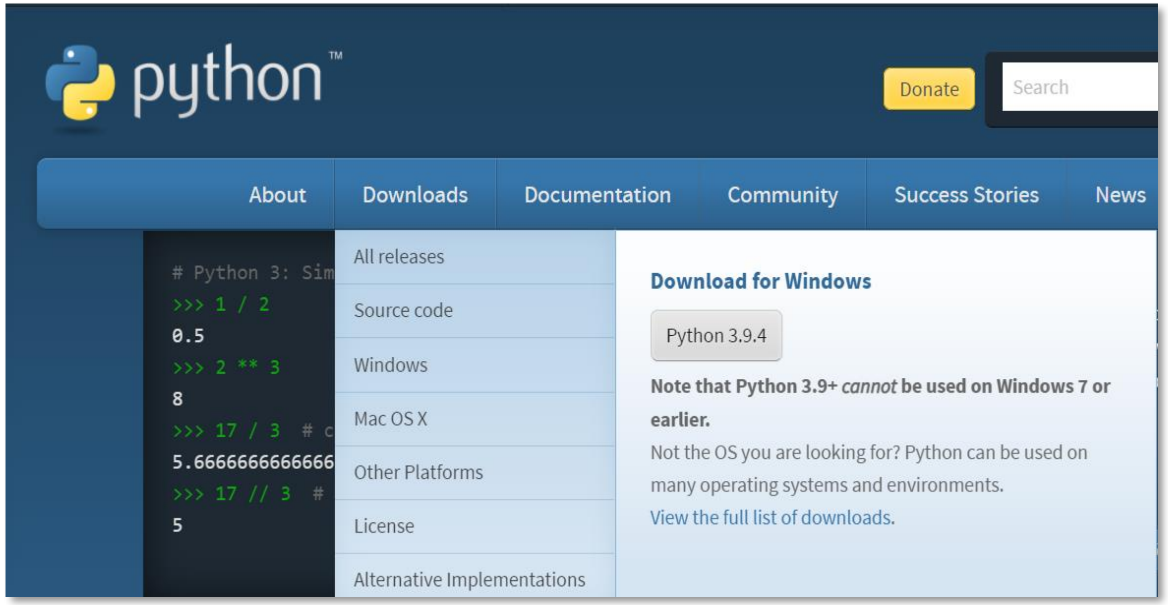
Python has two main versions:

- Python 2
- Python 3

However, Python installation differs among different operating systems. The use of Python 3 is highly preferred over Python 2.

Contd..

For the installation process, go to the official website of Python, i.e., *www.python.org.* Refer to the current stable version 3.9.4 as of date 13 April 2021.You will get the installer for Python 3.7 or Python 3.9. (at the time of writing). You may even have it with a *32-bit or 64-bit* processor versions.



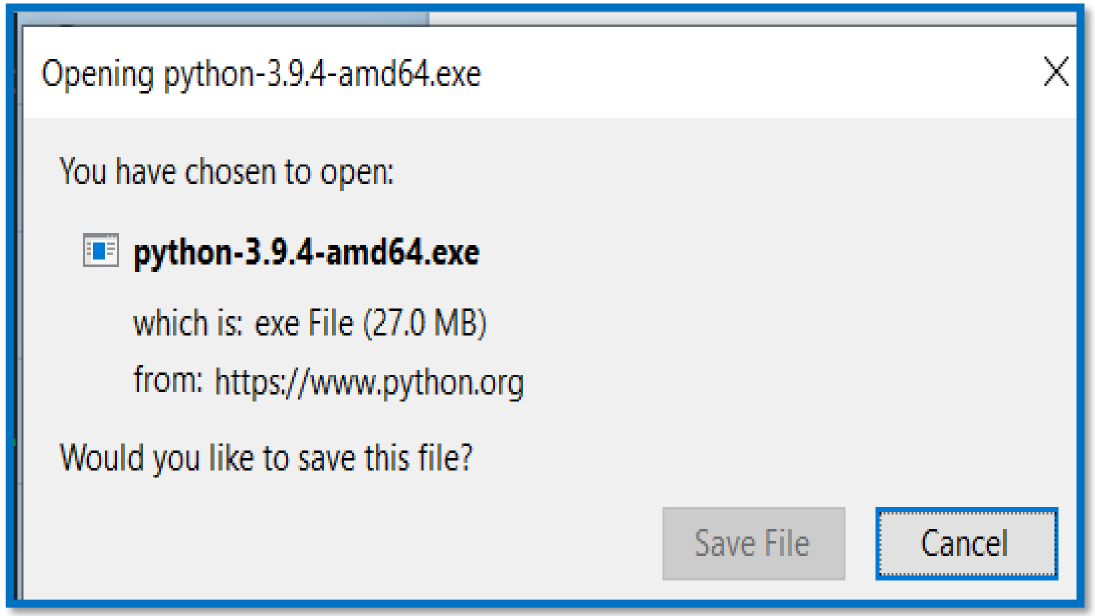
```
# Python 3: Sim
>>> 1 / 2
0.5
>>> 2 ** 3
8
>>> 17 / 3  # c
5.666666666666
>>> 17 // 3  #
5
```

Download for Windows

Note that Python 3.9+ *cannot* be used on Windows 7 or earlier.

Not the OS you are looking for? Python can be used on many operating systems and environments.

View the full list of downloads.



Opening python-3.9.4-amd64.exe

You have chosen to open:

python-3.9.4-amd64.exe

which is: exe File (27.0 MB)

from: https://www.python.org

Would you like to save this file?

1.2.1 Introduction to python IDE – IDLE

If you have recently installed Python on your computer, you might have seen a new *IDLE* program.

"What is this software doing on my computer?" you might be curious. I did not download that!"

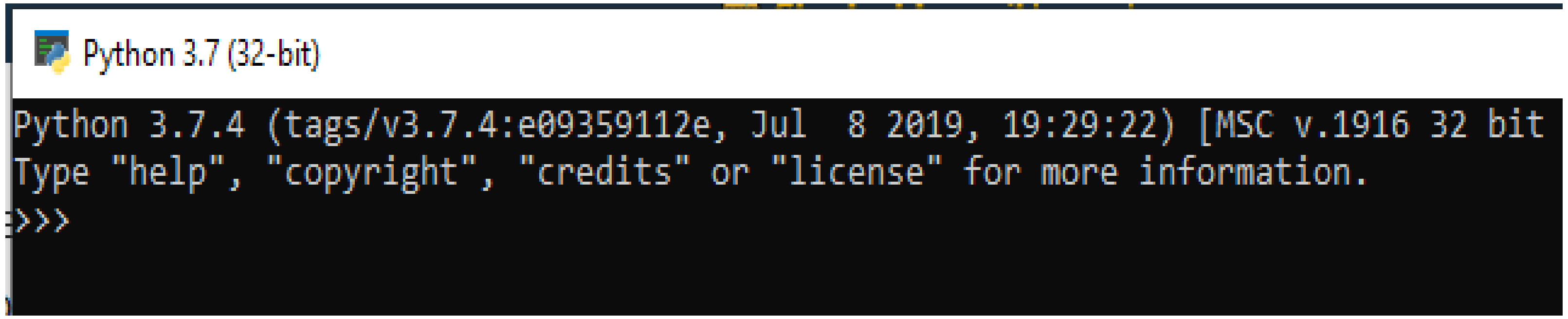
Though you might not have downloaded IDLE on your own, it is included with any Python installation. It is there to help you get acquainted with the language right away.

Any Python installation includes an *Integrated Development and Learning Environment*, abbreviated IDLE or even IDE.

Contd..

In a graphical user interface (GUI) desktop environment, the installation process puts an icon on the desktop or an object in the desktop menu system that launches Python.

In Windows, for example, there will be a category in the Start menu called Python 3.7. Under it, a menu item labeled Python 3.7.4 (32-bit).


Python 3.7 (32-bit)

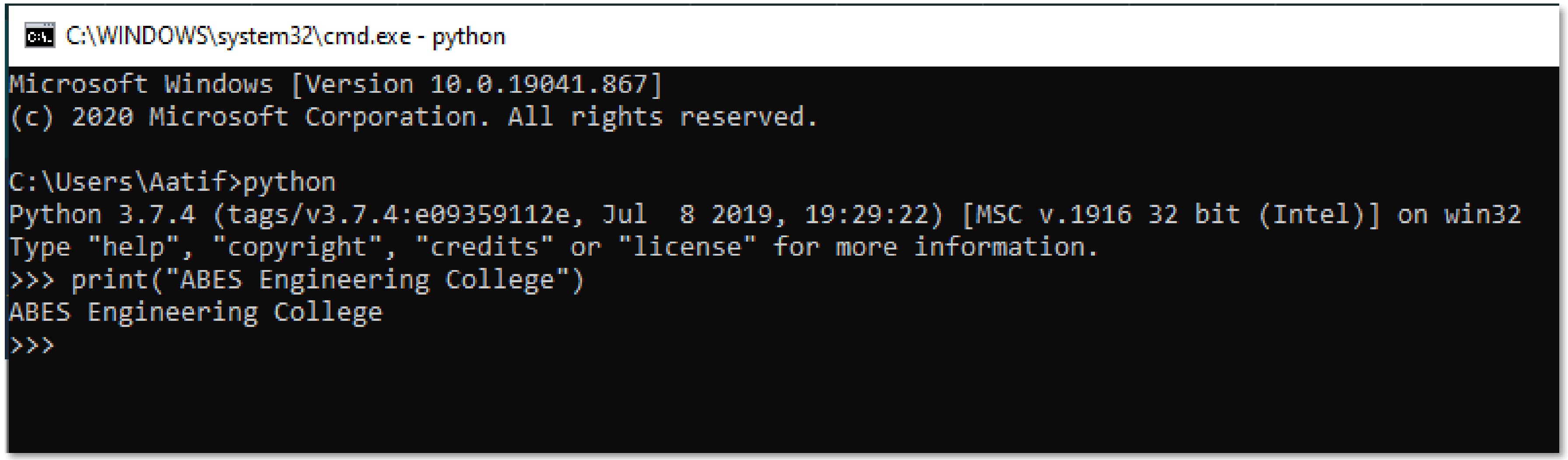
```
Python 3.7.4 (tags/v3.7.4:e09359112e, Jul  8 2019, 19:29:22) [MSC v.1916 32 bit
Type "help", "copyright", "credits" or "license" for more information.
>>>
```

Contd..

Alternate way: You can also open a terminal window and run the interpreter from the command line.

It is known as **Command Prompt** in Windows. It can be renamed **terminal** in macOS or Linux.

You can type **Windows key+ R** and type **cmd** to open Command Prompt. Then, type python to execute python programs.

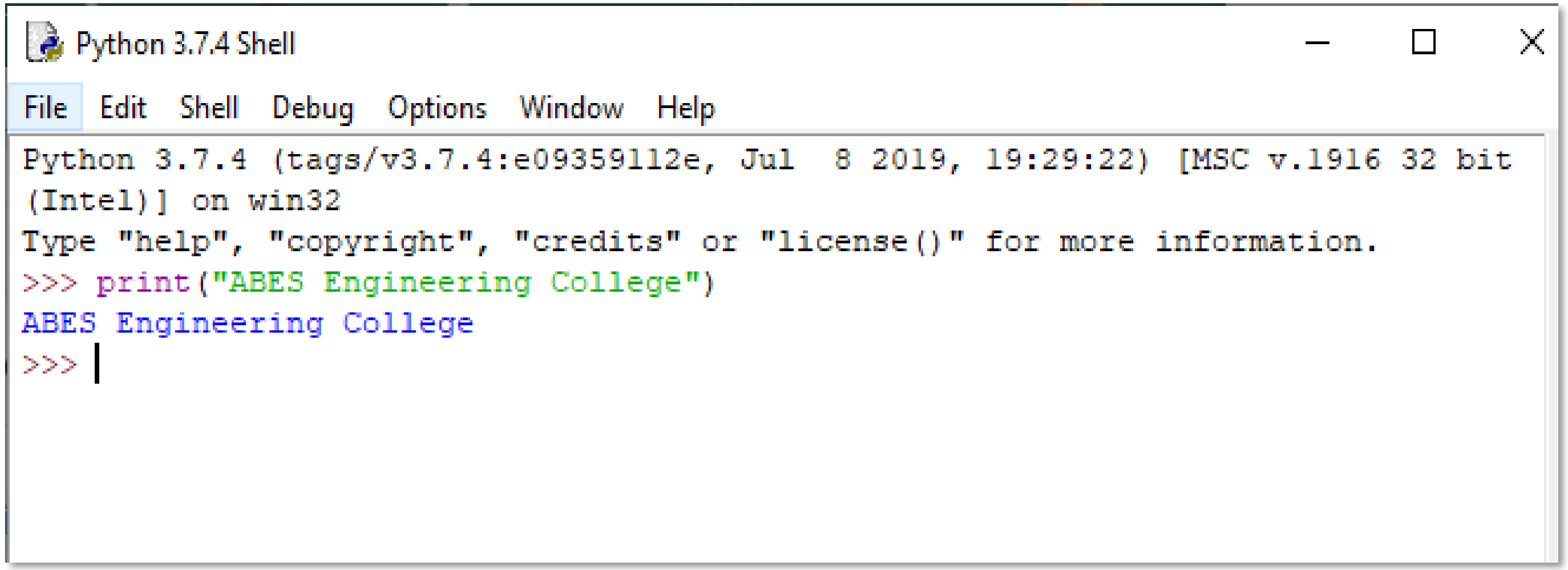


```
Microsoft Windows [Version 10.0.19041.867]
(c) 2020 Microsoft Corporation. All rights reserved.

C:\Users\Aatif>python
Python 3.7.4 (tags/v3.7.4:e09359112e, Jul  8 2019, 19:29:22) [MSC v.1916 32 bit (Intel)] on win32
Type "help", "copyright", "credits" or "license" for more information.
>>> print("ABES Engineering College")
ABES Engineering College
>>>
```

Contd..

Start working with Python shell.

To print () to display the string "ABES Engineering College" on your computer. Enter the command one at a time, and Python returns the results of each command.

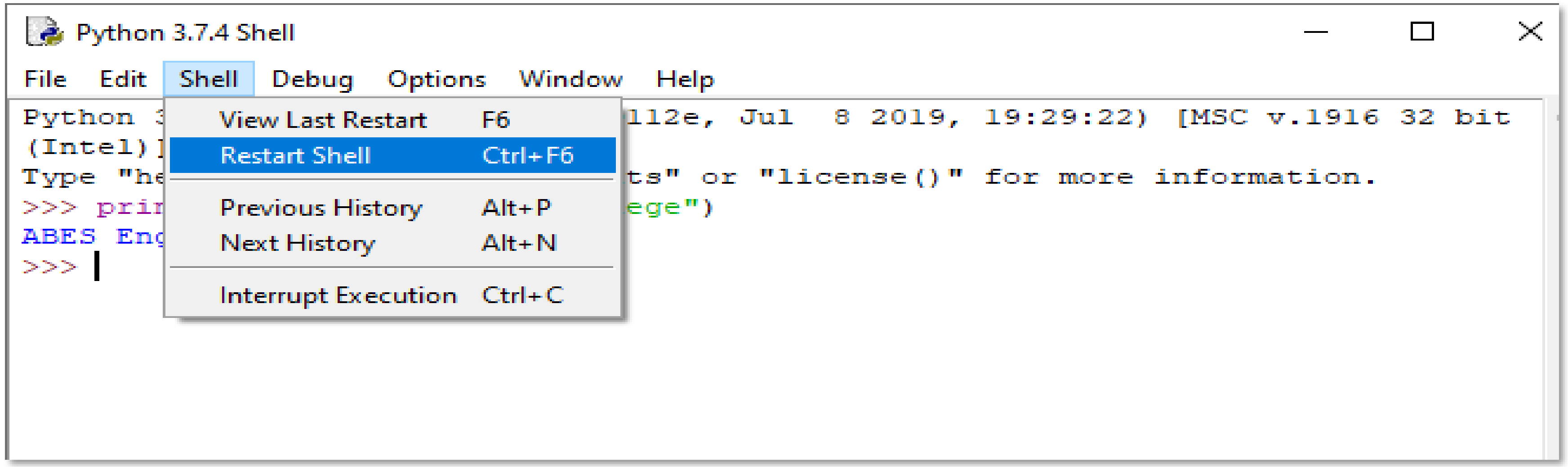


```
Python 3.7.4 (tags/v3.7.4:e09359112e, Jul  8 2019, 19:29:22) [MSC v.1916 32 bit
(Intel)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>> print("ABES Engineering College")
ABES Engineering College
>>> 
```

Contd..

From this menu bar, you can restart the shell. It will behave as if you had launched a new instance of Python IDLE. The shell will forget anything from its former state.

If you want to exit the interpreter, then type exit () and press Enter.

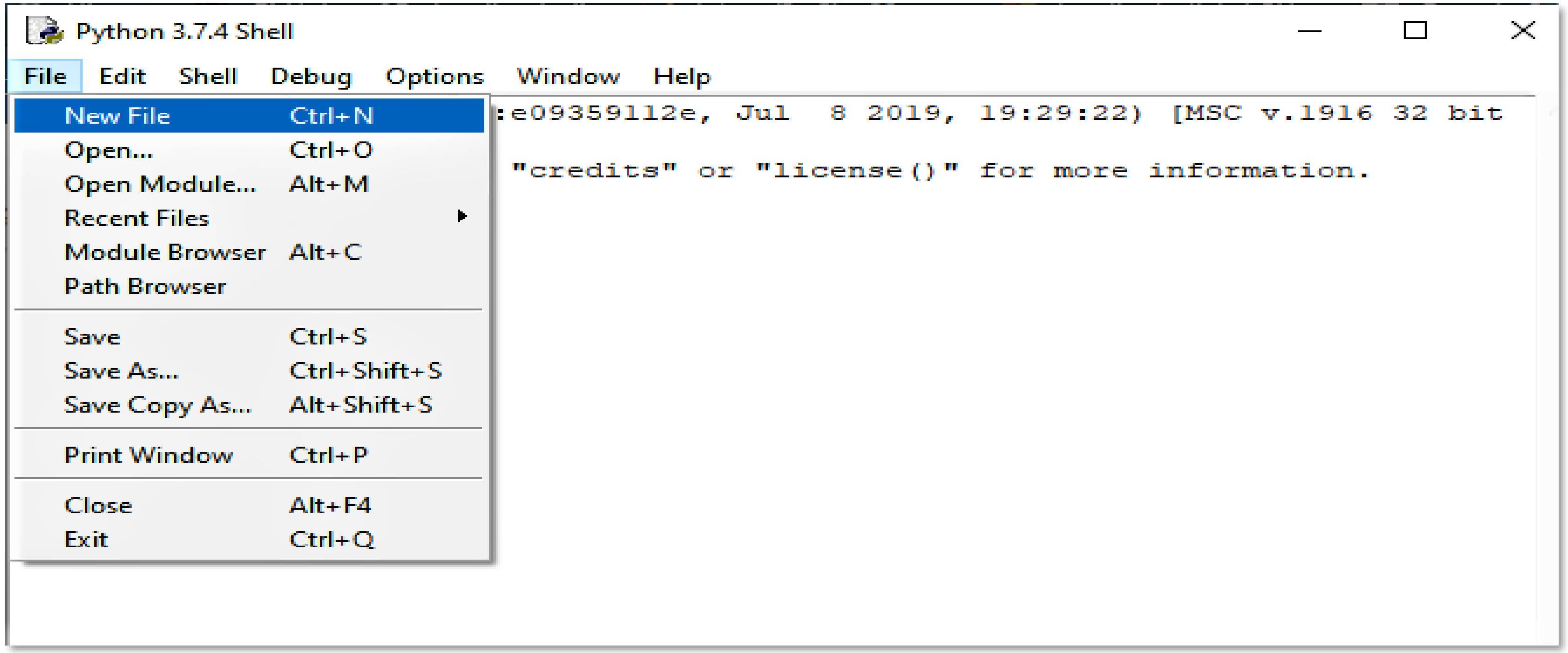


```
Python 3.7.4 Shell
File  Edit  Shell  Debug  Options  Window  Help
Python 3                         112e, Jul  8 2019, 19:29:22) [MSC v.1916 32 bit
(Intel)]
Type "he                         ts" or "license()" for more information.
>>> prin                         ege")
ABES Eng
>>> |
```

Python IDLE Editor

Python IDLE includes a full-featured file editor, allowing you to write and run Python programs. The built-in file editor also supplies many tools to speed up your coding workflows, such as code completion and automated indentation.

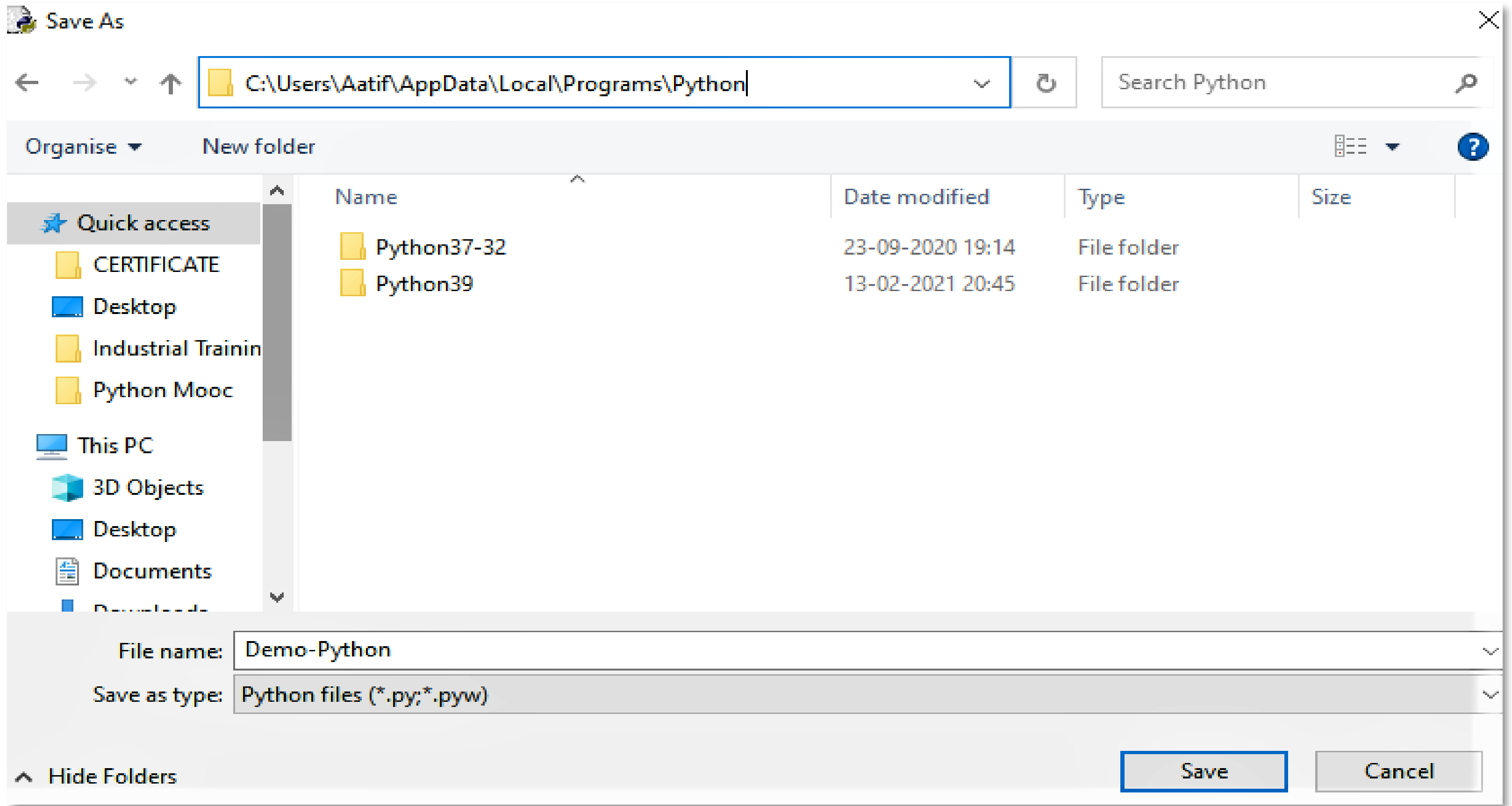
Select File “New File” from the menu bar to begin a new Python file



```
:e09359112e, Jul  8 2019, 19:29:22) [MSC v.1916 32 bit

 "credits" or "license()" for more information.
```

Contd..

When you are ready to work on a file, click the Edit button. You can save the file as Demo-Python.py (.py is an extension to save python scripts files) in the specified default location "C:\Users\Aatif\AppData\Local\Programs\Python."



Save As

C:\Users\Aatif\AppData\Local\Programs\Python

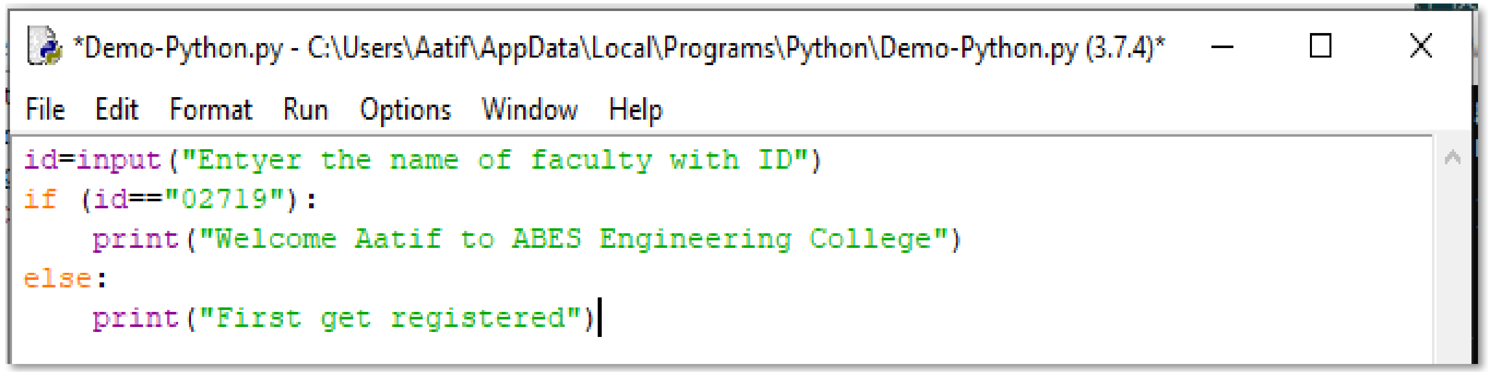
Name	Date modified	Type	Size
Python37-32	23-09-2020 19:14	File folder	
Python39	13-02-2021 20:45	File folder	

File name: Demo-Python

Save as type: Python files (*.py;*.pyw)

Contd..

When you want to run a file, you must first ensure that it has been saved, remember to check for asterisks * around the filename at the top of the file editor window to see whether the file was correctly saved.


Demo-Python.py - C:\Users\Aatif\AppData\Local\Programs\Python\Demo-Python.py (3.7.4)
File Edit Format Run Options Window Help

```
id=input("Entyer the name of faculty with ID")
if (id=="02719"):
    print("Welcome Aatif to ABES Engineering College")
else:
    print("First get registered")
```

But do not panic if you forget! When you want to run an unsaved file in Python IDLE, *it will prompt you to save it.*

Click the F5 key on your keyboard to run a file in IDLE. You can also use the menu bar to choose *Run Module*

1.2.2 Setting Up Your Environment

The route*(path that lists the directories in which the OS (Operating System) looks for executables*) is saved in an environment variable called a string.This variable holds knowledge that the command shell and other programs can use.

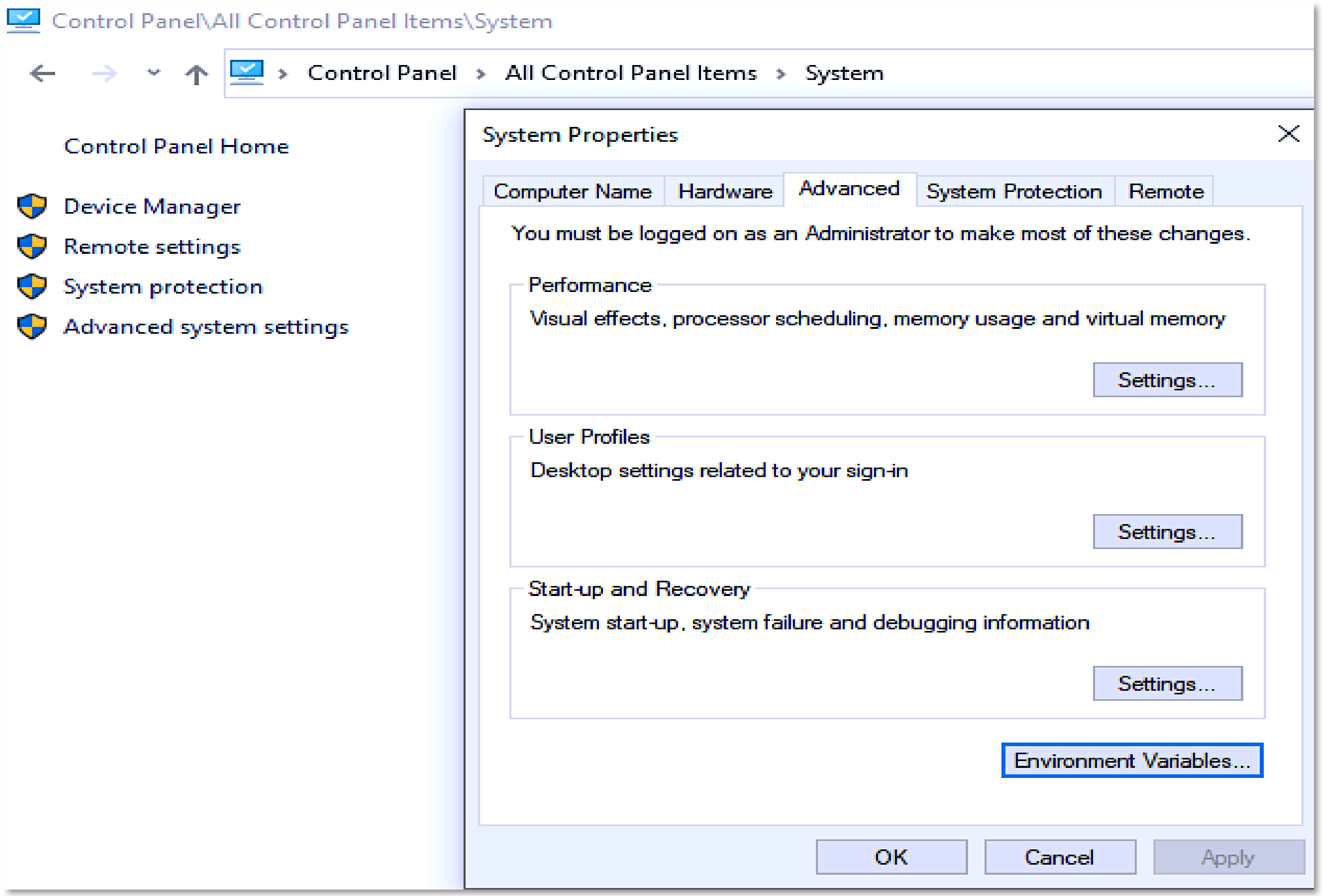
In Unix, the path variable is known as PATH, and in Windows, it is known as Path *(Unix is case sensitive; Windows is not).*

Following are the steps are taken for setting up the environment:

- *Step 1 – Install Python 3.7 (Latest Version) from python.org.*
- *Step 2-- Add the Python 3.7 Directory to your System Path Environment Variable*

Contd..

So, navigate to *Control Panel –> System –> Advanced System Settings–> Environment Variables* and choose the PATH variable



Control Panel\All Control Panel Items\System

Control Panel › All Control Panel Items › System

Control Panel Home

Device Manager
Remote settings
System protection
Advanced system settings

System Properties

Computer Name | Hardware | Advanced | System Protection | Remote

You must be logged on as an Administrator to make most of these changes.

Performance
Visual effects, processor scheduling, memory usage and virtual memory
Settings...

User Profiles
Desktop settings related to your sign-in
Settings...

Start-up and Recovery
System start-up, system failure and debugging information
Settings...

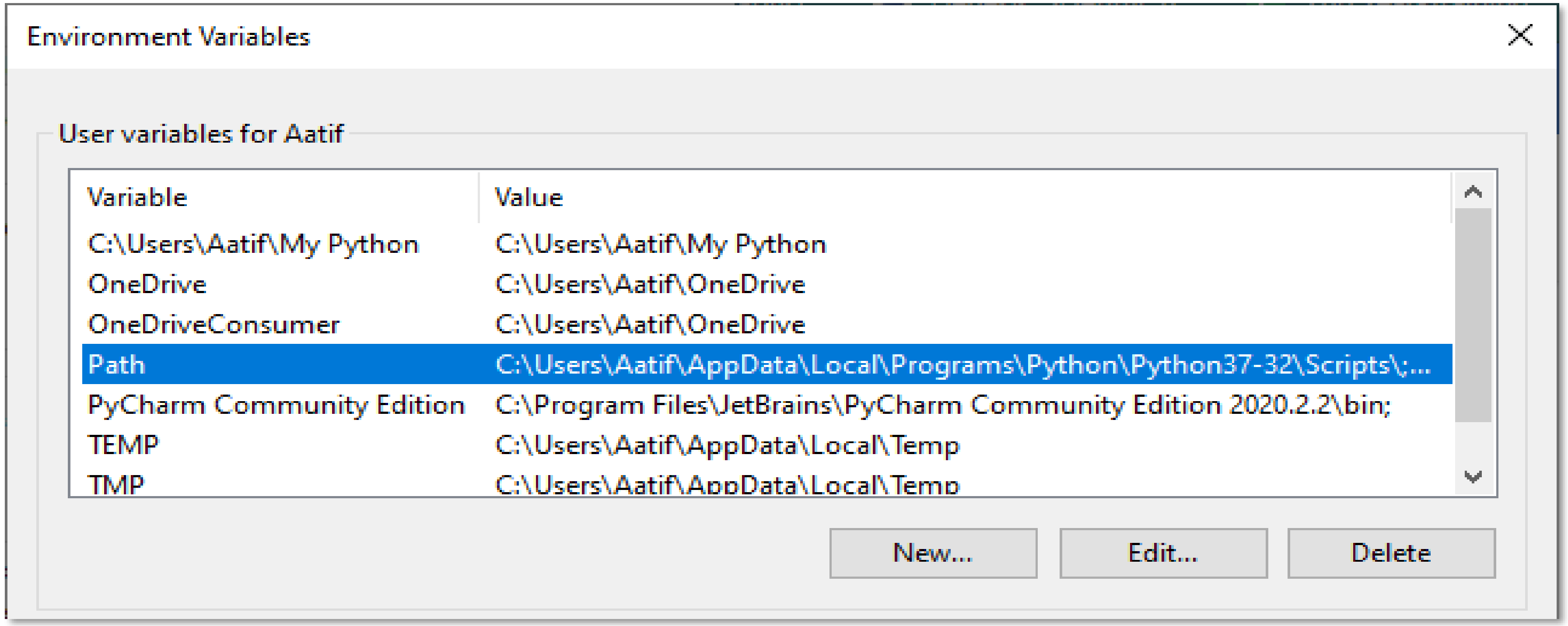
Environment Variables...

OK | Cancel | Apply

Contd..

Add the Python path to the end of the string, this is where the package management software, unit testing tools, and other command line-accessible Python programs can live.

C:\Users\Aatif\AppData\Local\Programs\Python\Python37-32\Scripts\



Variable	Value
C:\Users\Aatif\My Python	C:\Users\Aatif\My Python
OneDrive	C:\Users\Aatif\OneDrive
OneDriveConsumer	C:\Users\Aatif\OneDrive
Path	C:\Users\Aatif\AppData\Local\Programs\Python\Python37-32\Scripts\;...
PyCharm Community Edition	C:\Program Files\JetBrains\PyCharm Community Edition 2020.2.2\bin;
TEMP	C:\Users\Aatif\AppData\Local\Temp
TMP	C:\Users\Aatif\AppData\Local\Temp

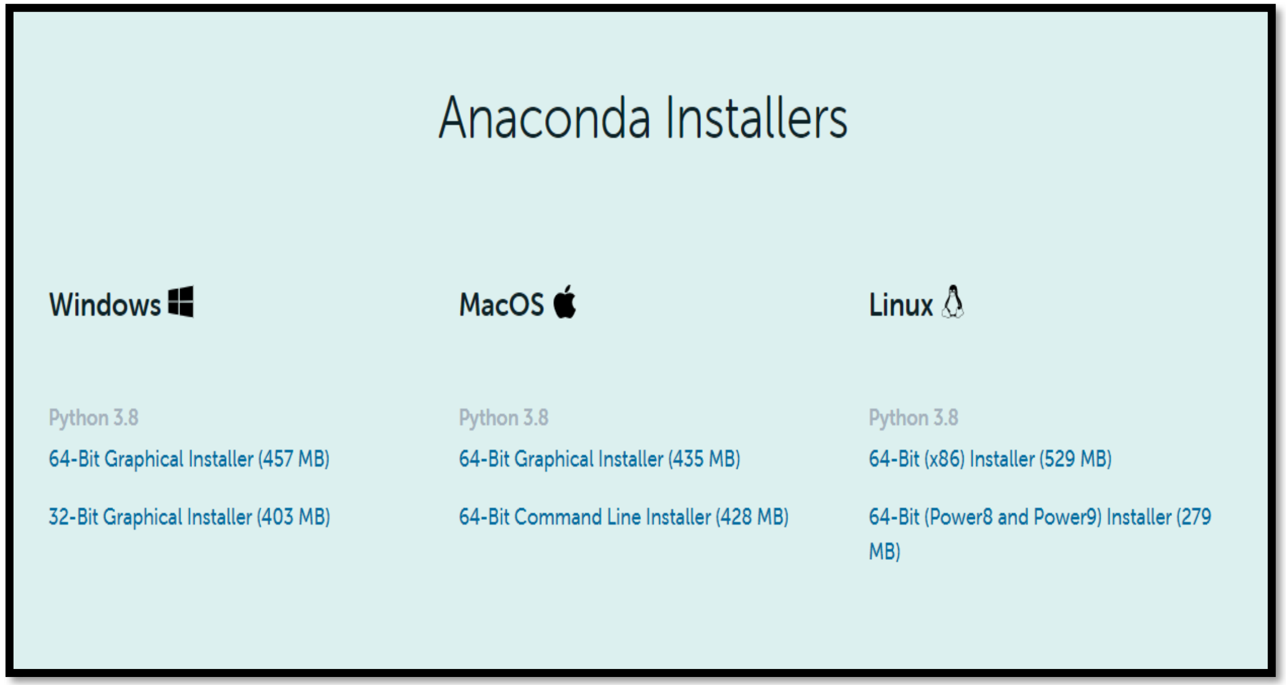
1.2.3 Installation of Anaconda Navigator

Anaconda Navigator is a desktop graphical user interface that comes with Anaconda, which allows you to open programs and control conda packages, environments, and networks without needing to use a command-line interface.

For the installation process, go to the official website of anaconda navigator https://docs.anaconda.com/anaconda/navigator/.

Latest version:

Anaconda3-2020.11-Windows-x86_64.exe

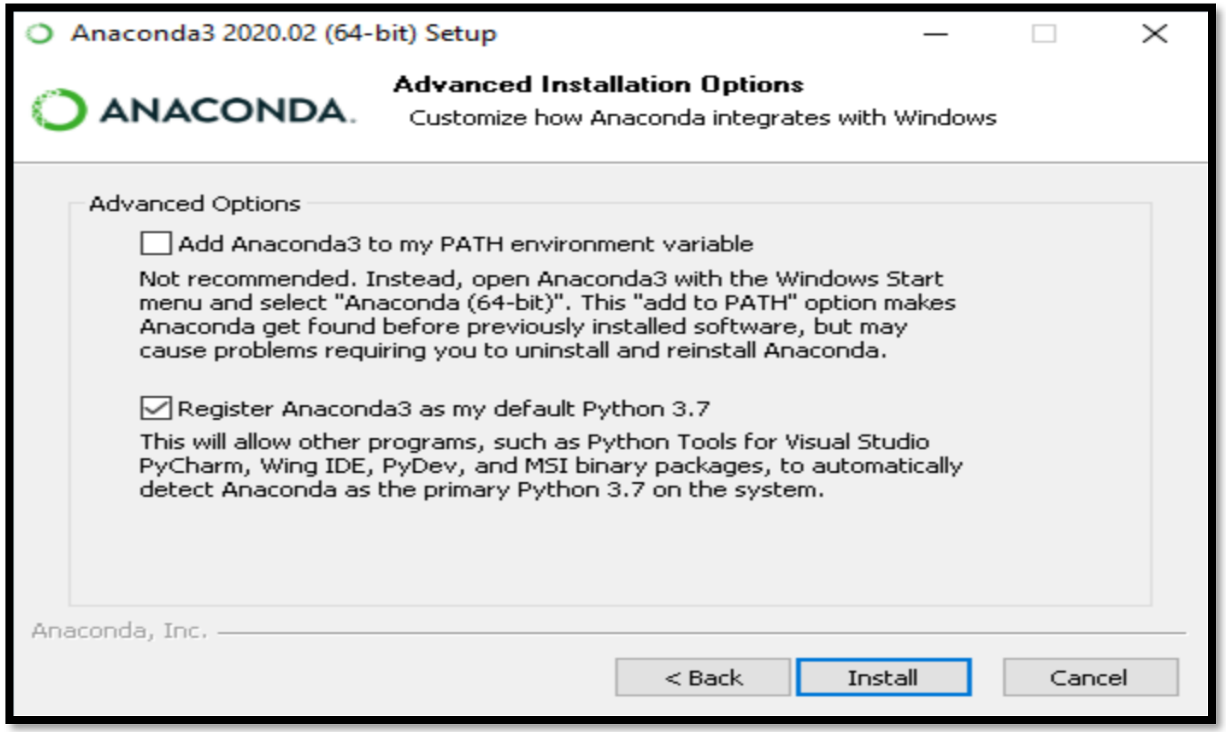


Anaconda Installers

Windows	MacOS	Linux
Python 3.8	Python 3.8	Python 3.8
64-Bit Graphical Installer (457 MB)	64-Bit Graphical Installer (435 MB)	64-Bit (x86) Installer (529 MB)
32-Bit Graphical Installer (403 MB)	64-Bit Command Line Installer (428 MB)	64-Bit (Power8 and Power9) Installer (279 MB)

Contd..

Click on install and Choose destination folder for installation.



Anaconda3 2020.02 (64-bit) Setup

Advanced Installation Options

Customize how Anaconda integrates with Windows

Advanced Options

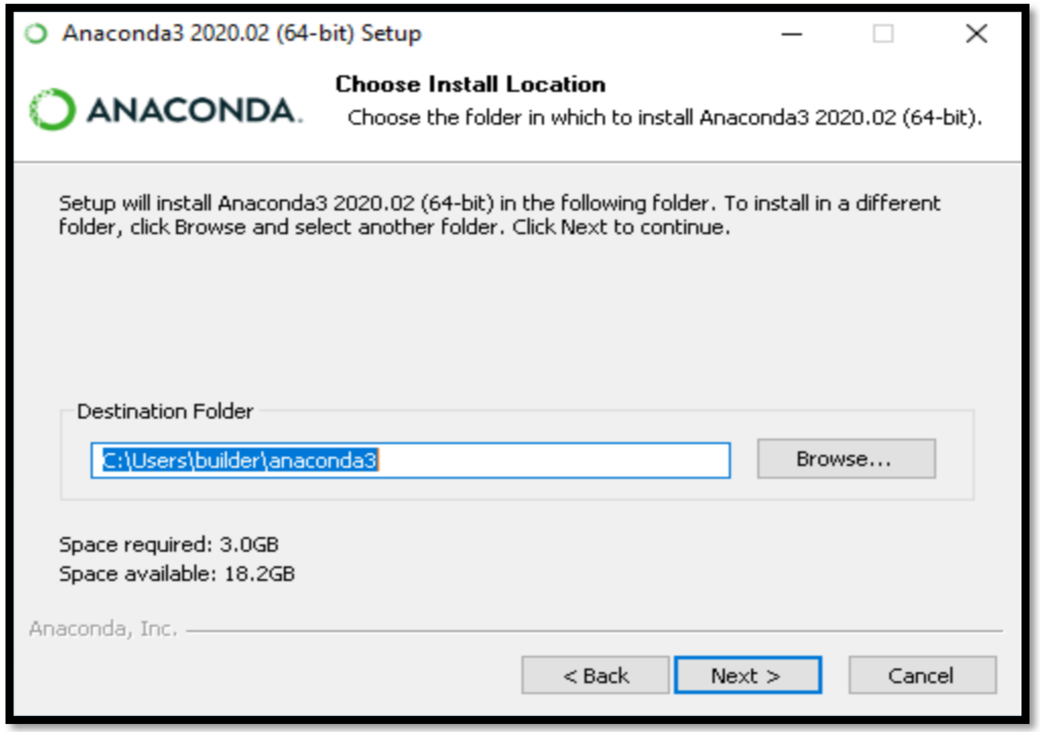
☐ Add Anaconda3 to my PATH environment variable

Not recommended. Instead, open Anaconda3 with the Windows Start menu and select "Anaconda (64-bit)". This "add to PATH" option makes Anaconda get found before previously installed software, but may cause problems requiring you to uninstall and reinstall Anaconda.

☑ Register Anaconda3 as my default Python 3.7

This will allow other programs, such as Python Tools for Visual Studio PyCharm, Wing IDE, PyDev, and MSI binary packages, to automatically detect Anaconda as the primary Python 3.7 on the system.

Anaconda, Inc.



Anaconda3 2020.02 (64-bit) Setup

Choose Install Location

Choose the folder in which to install Anaconda3 2020.02 (64-bit).

Setup will install Anaconda3 2020.02 (64-bit) in the following folder. To install in a different folder, click Browse and select another folder. Click Next to continue.

Destination Folder: C:\Users\builder\anaconda3

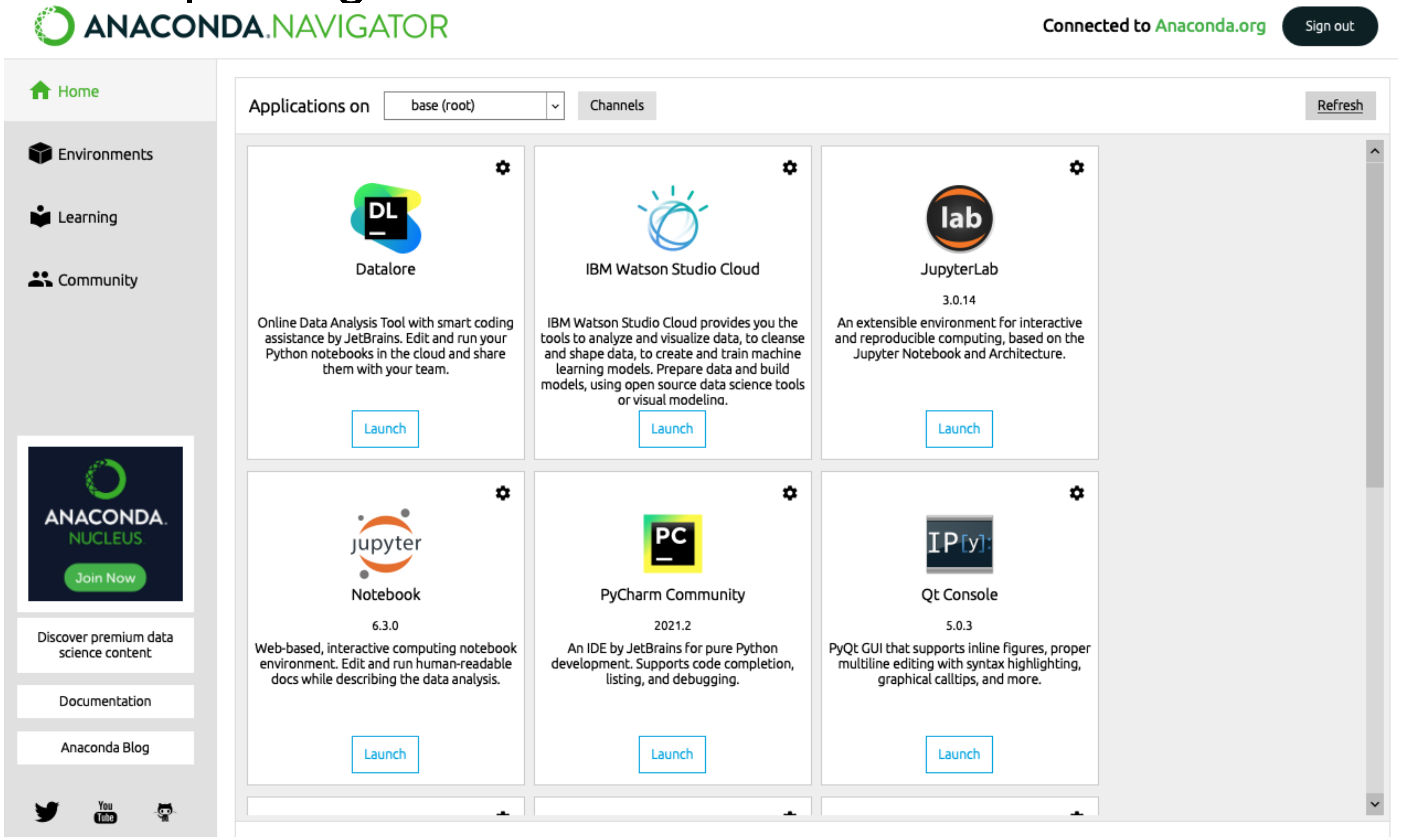
Space required: 3.0GB
Space available: 18.2GB

Anaconda, Inc.

1.2.4 Quick Tour of Jupyter Notebook

In an earlier topic, we have seen the installation of the Anaconda package. Jupyter comes by default with this package

The *Jupyter Notebook* is a fantastic platform for creating basic and advanced level programs.



Datalore

Online Data Analysis Tool with smart coding assistance by JetBrains. Edit and run your Python notebooks in the cloud and share them with your team.

IBM Watson Studio Cloud

IBM Watson Studio Cloud provides you the tools to analyze and visualize data, to cleanse and shape data, to create and train machine learning models. Prepare data and build models, using open source data science tools or visual modeling.

JupyterLab

3.0.14

An extensible environment for interactive and reproducible computing, based on the Jupyter Notebook and Architecture.

Notebook

6.3.0

Web-based, interactive computing notebook environment. Edit and run human-readable docs while describing the data analysis.

PyCharm Community

2021.2

An IDE by JetBrains for pure Python development. Supports code completion, listing, and debugging.

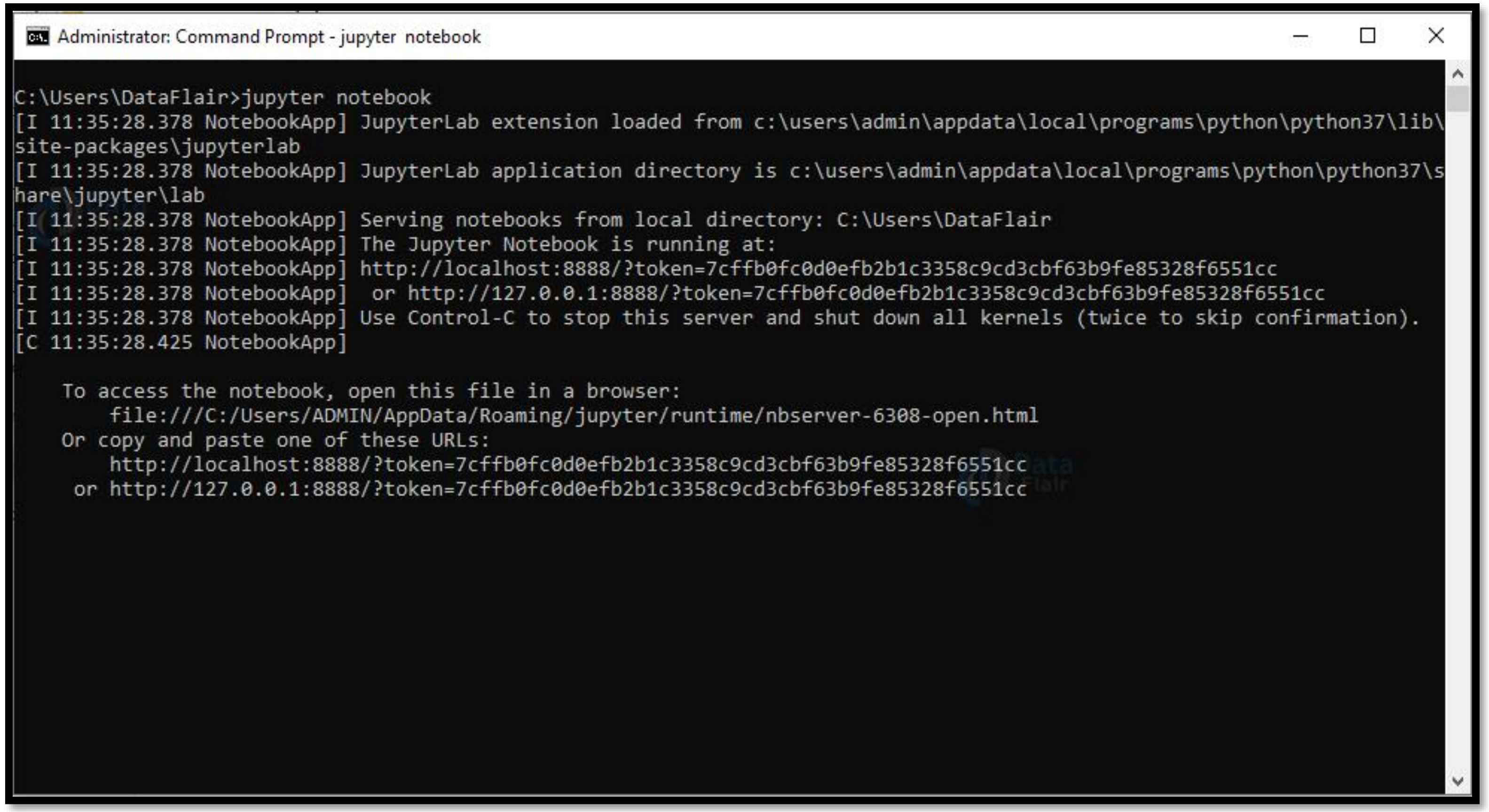
Qt Console

5.0.3

PyQt GUI that supports inline figures, proper multiline editing with syntax highlighting, graphical calltips, and more.

Contd..

Alternate way: If you type 'Jupyter notebook' into your command prompt, it will open the Jupyter dashboard for you

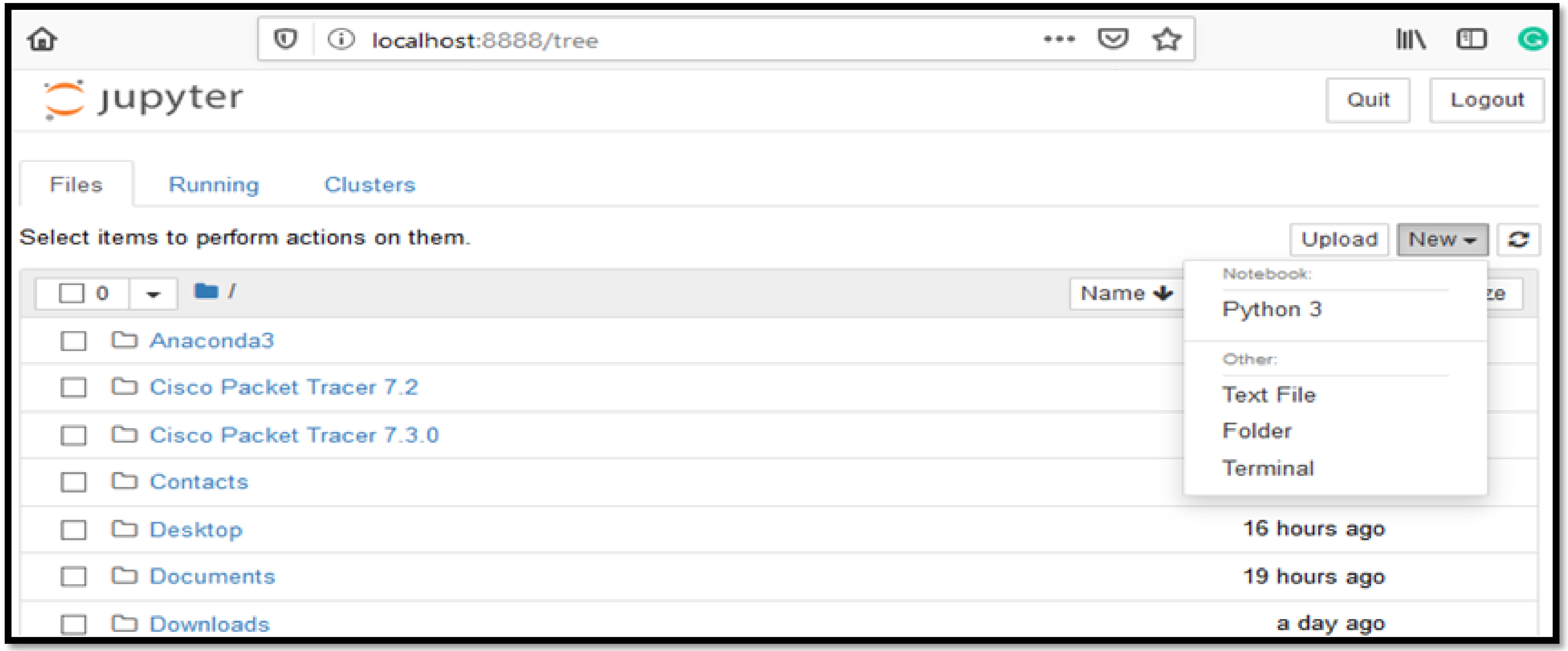


Administrator: Command Prompt - jupyter notebook

```
C:\Users\DataFlair>jupyter notebook
[I 11:35:28.378 NotebookApp] JupyterLab extension loaded from c:\users\admin\appdata\local\programs\python\python37\lib\
site-packages\jupyterlab
[I 11:35:28.378 NotebookApp] JupyterLab application directory is c:\users\admin\appdata\local\programs\python\python37\s
hare\jupyter\lab
[I 11:35:28.378 NotebookApp] Serving notebooks from local directory: C:\Users\DataFlair
[I 11:35:28.378 NotebookApp] The Jupyter Notebook is running at:
[I 11:35:28.378 NotebookApp] http://localhost:8888/?token=7cffb0fc0d0efb2b1c3358c9cd3cbf63b9fe85328f6551cc
[I 11:35:28.378 NotebookApp]  or http://127.0.0.1:8888/?token=7cffb0fc0d0efb2b1c3358c9cd3cbf63b9fe85328f6551cc
[I 11:35:28.378 NotebookApp] Use Control-C to stop this server and shut down all kernels (twice to skip confirmation).
[C 11:35:28.425 NotebookApp]

    To access the notebook, open this file in a browser:
        file:///C:/Users/ADMIN/AppData/Roaming/jupyter/runtime/nbserver-6308-open.html
    Or copy and paste one of these URLs:
        http://localhost:8888/?token=7cffb0fc0d0efb2b1c3358c9cd3cbf63b9fe85328f6551cc
     or http://127.0.0.1:8888/?token=7cffb0fc0d0efb2b1c3358c9cd3cbf63b9fe85328f6551cc
```

Contd..

You might have found that the URL for the dashboard is *https://localhost:8888/tree* while Jupyter Notebook is open in your window. The term "localhost" does not refer to a website, but to the fact that the content is served from your own devices.


localhost:8888/tree

jupyter

Quit Logout

Files Running Clusters

Select items to perform actions on them.

Upload New

0 /

Name

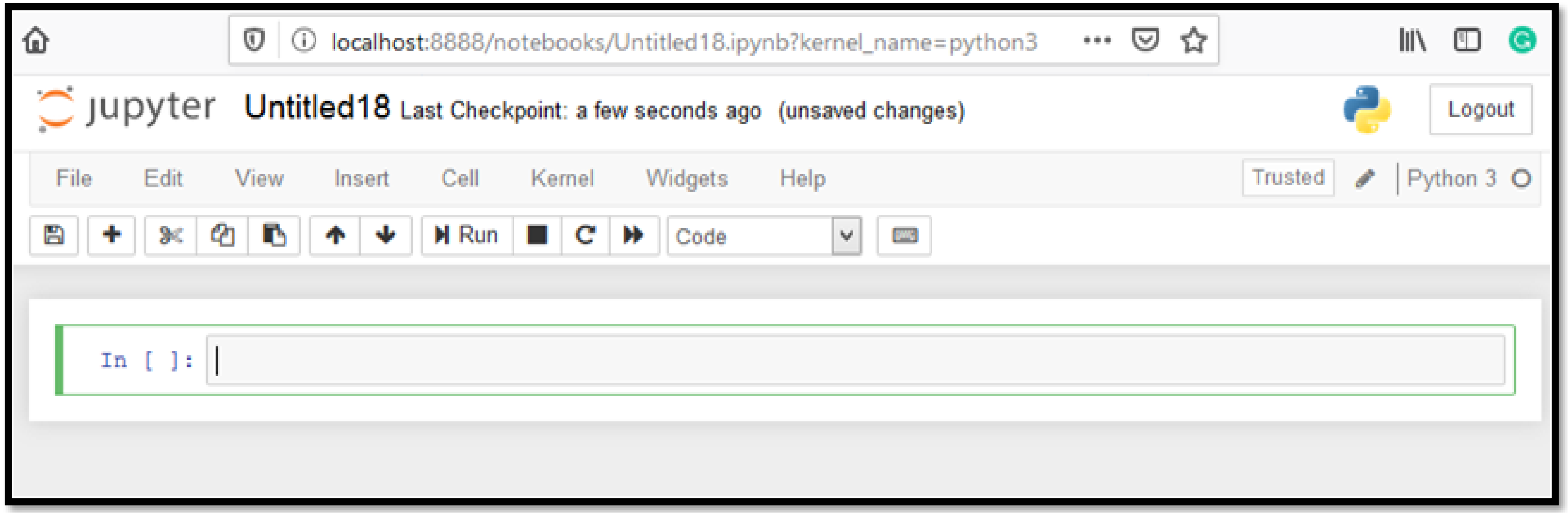
Notebook:
Python 3
Other:
Text File
Folder
Terminal

Anaconda3
Cisco Packet Tracer 7.2
Cisco Packet Tracer 7.3.0
Contacts
Desktop 16 hours ago
Documents 19 hours ago
Downloads a day ago

Contd..

If you want to create your first file, then you must click on new and then python3. When you return to the dashboard, you can see the new file **Untitled18.ipynb** with green boundary in cell.

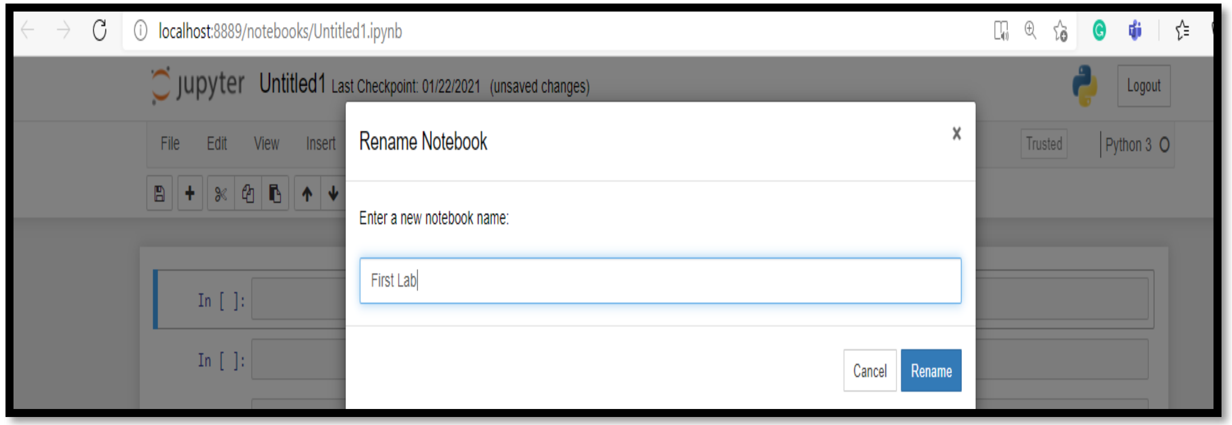
A cell is a container for the text that will be viewed in the notebook or code that the notebook's kernel will execute.


localhost:8888/notebooks/Untitled18.ipynb?kernel_name=python3
jupyter
Untitled18 Last Checkpoint: a few seconds ago (unsaved changes)
Logout
File
Edit
View
Insert
Cell
Kernel
Widgets
Help
Trusted
Python 3
Run
Code
In [ ]:

Contd..

Every *ipynb file* stands for a single notebook, which means that each time you create a new notebook, a new.ipynb file is generated .You should be aware of kernel that are unfamiliar to you.

A kernel is a kind of "computational engine" that runs the code in a notebook paper.


localhost:8889/notebooks/Untitled1.ipynb

jupyter Untitled1 Last Checkpoint: 01/22/2021 (unsaved changes)

Rename Notebook

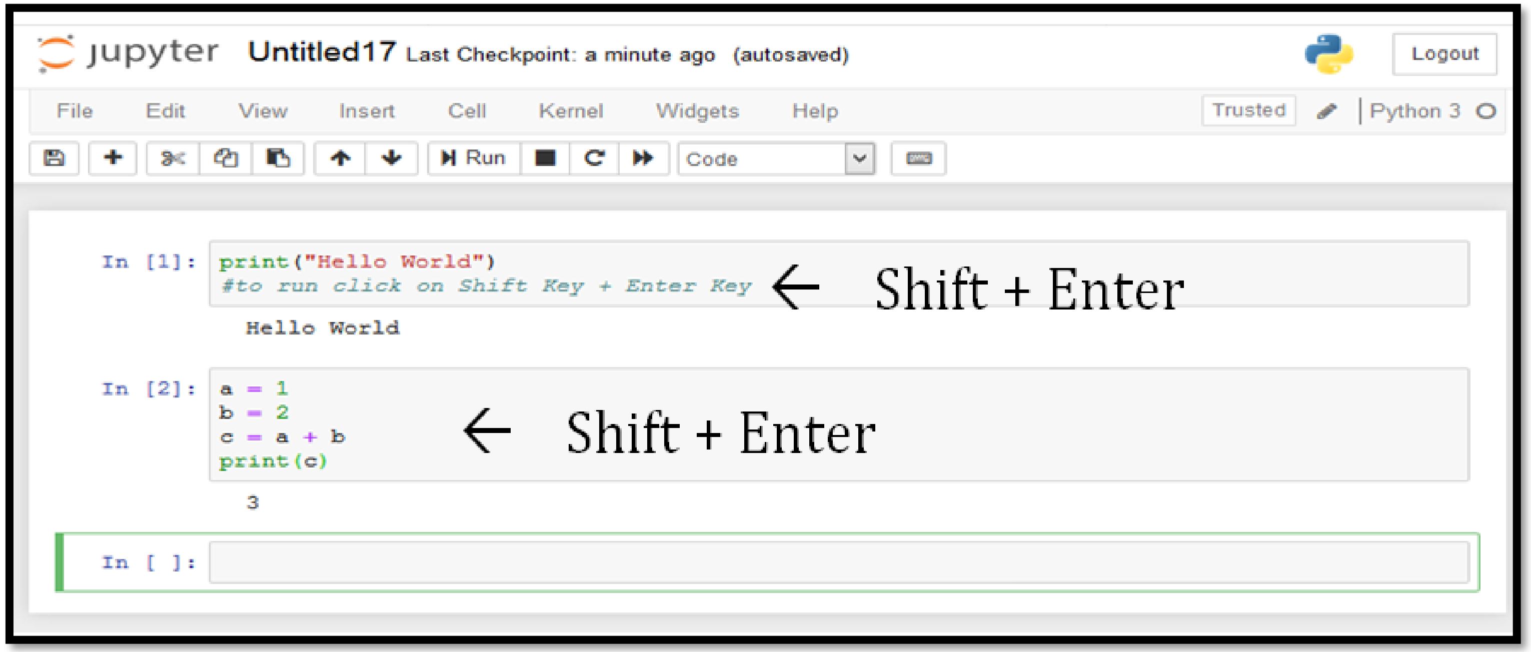
Enter a new notebook name:

First Lab

Contd..

Shift + Enter is the shortcut command to run your cell.

The green boundary over the cell shows the editable mode, and the Blue boundary over the cell shows the command mode.

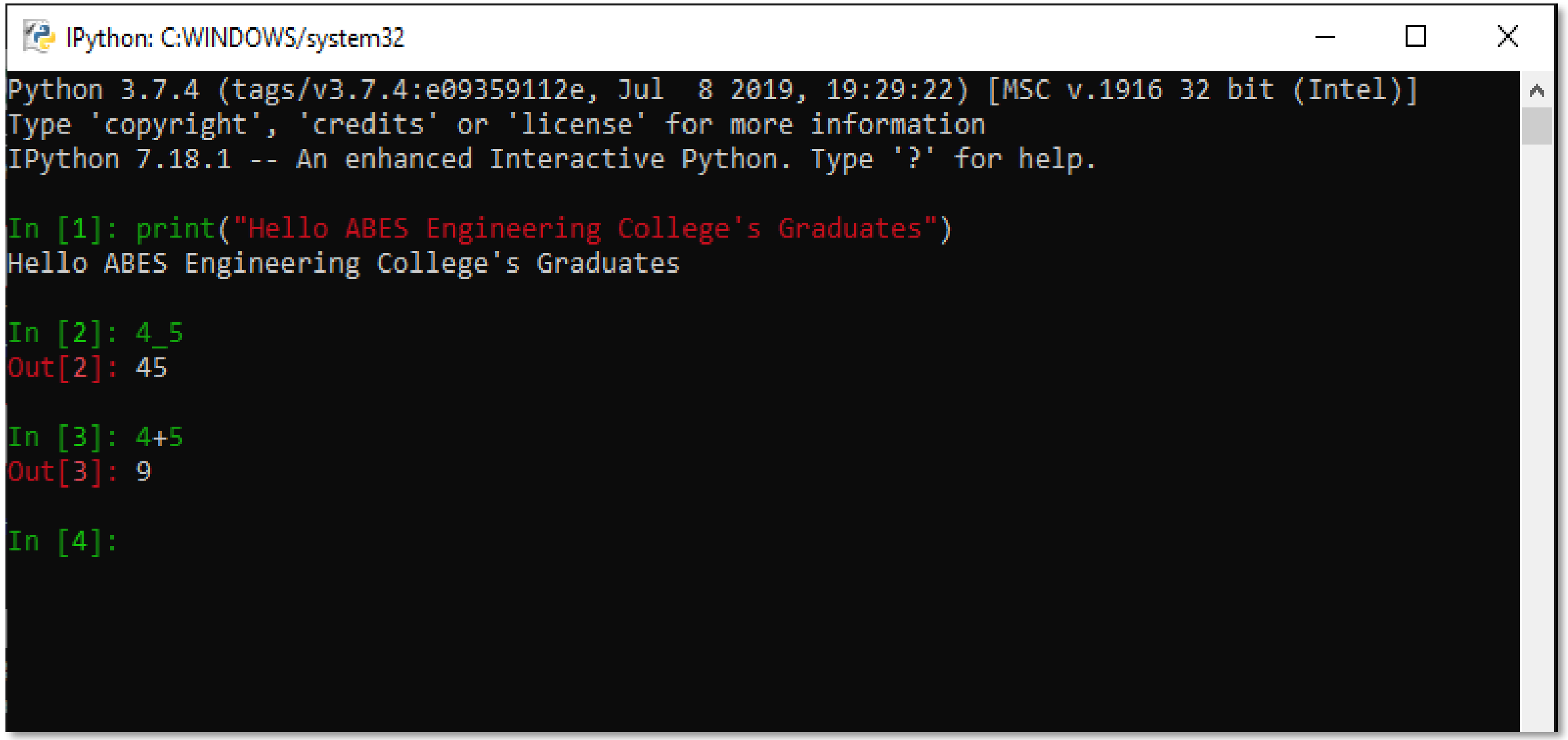

jupyter Untitled17 Last Checkpoint: a minute ago (autosaved)
Logout
File Edit View Insert Cell Kernel Widgets Help
Trusted
Python 3
Run
Code

```
In [1]: print("Hello World")
        #to run click on Shift Key + Enter Key    ← Shift + Enter

        Hello World

In [2]: a = 1
        b = 2
        c = a + b                                 ← Shift + Enter
        print(c)

        3

In [ ]:
```

1.2.5 Python vs. IPython

IPython's interactive shell is known as Ipython.

IPython is a Python graphical command-line terminal founded by Fernando Perez in 2001. IPython supplies an improved *read-eval-print loop* (REPL) environment that is particularly well suited to scientific computing.



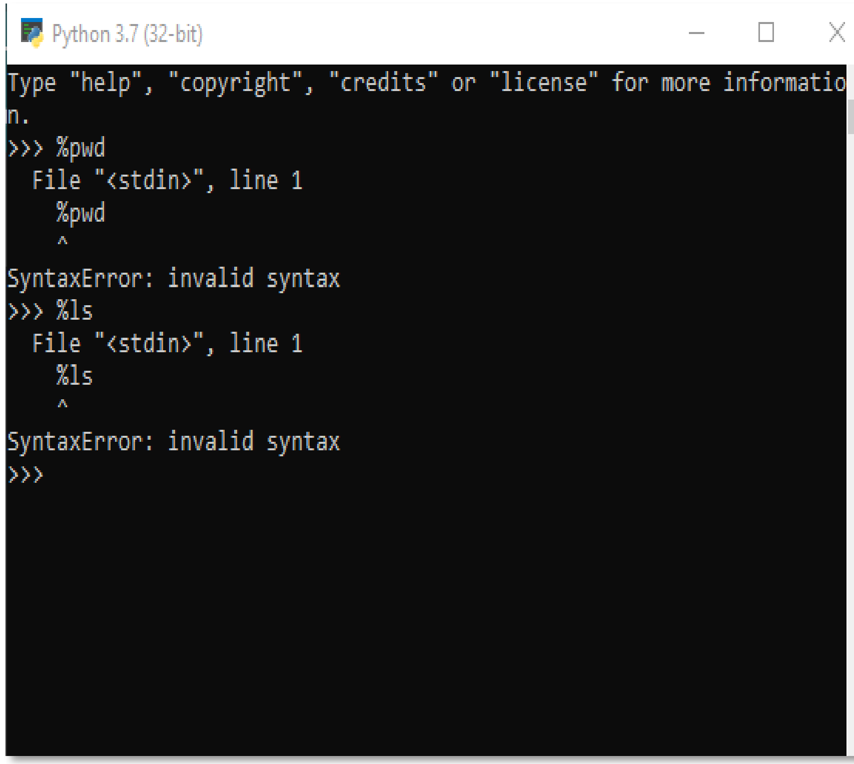
```
Python 3.7.4 (tags/v3.7.4:e09359112e, Jul  8 2019, 19:29:22) [MSC v.1916 32 bit (Intel)]
Type 'copyright', 'credits' or 'license' for more information
IPython 7.18.1 -- An enhanced Interactive Python. Type '?' for help.

In [1]: print("Hello ABES Engineering College's Graduates")
Hello ABES Engineering College's Graduates

In [2]: 4_5
Out[2]: 45

In [3]: 4+5
Out[3]: 9

In [4]:
```

Contd..

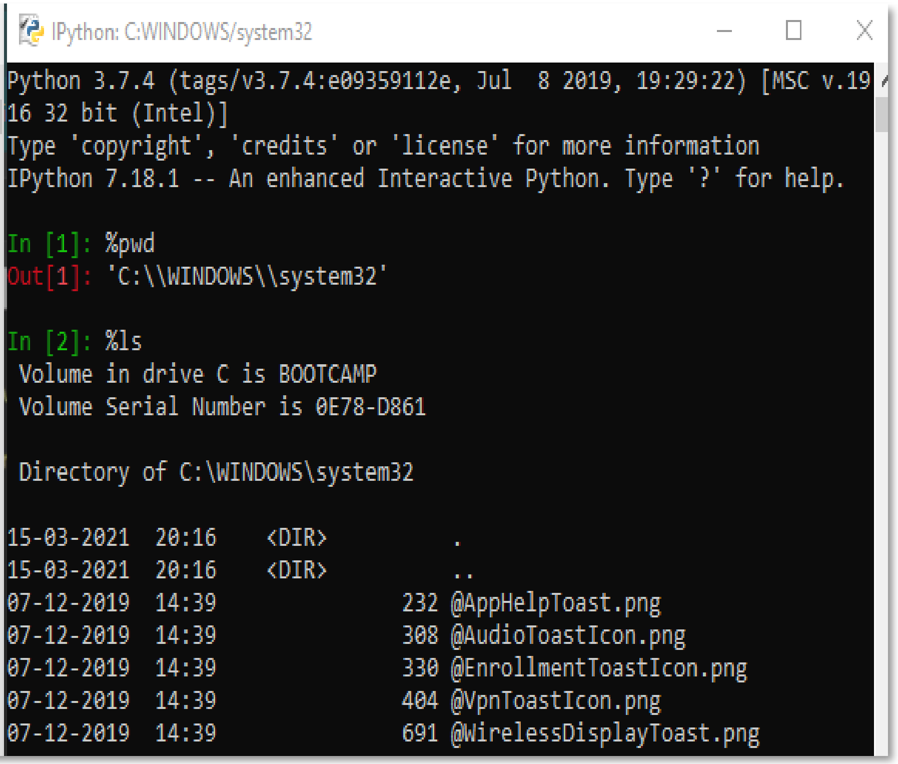
IPython is interactive, and it gives some relaxation to the eyes of coders by introducing some colors. *Some useful commands are not present with the existing Python idle.*

The following are the valuable commands:

- %pwd
- %ls
- %History



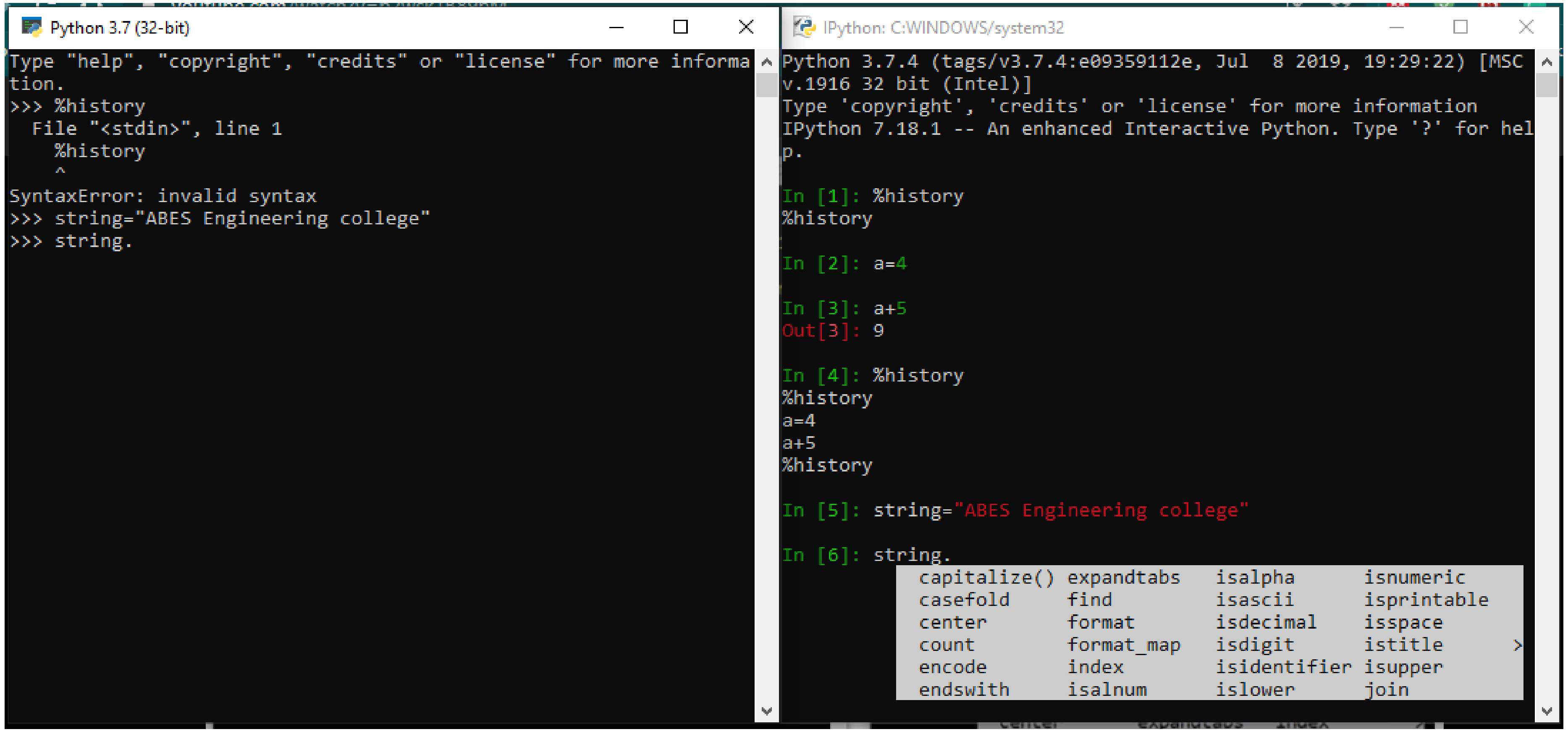
```
Type "help", "copyright", "credits" or "license" for more informatio
n.
>>> %pwd
  File "<stdin>", line 1
    %pwd
    ^
SyntaxError: invalid syntax
>>> %ls
  File "<stdin>", line 1
    %ls
    ^
SyntaxError: invalid syntax
>>>
```



```
Python 3.7.4 (tags/v3.7.4:e09359112e, Jul  8 2019, 19:29:22) [MSC v.19
16 32 bit (Intel)]
Type 'copyright', 'credits' or 'license' for more information
IPython 7.18.1 -- An enhanced Interactive Python. Type '?' for help.

In [1]: %pwd
Out[1]: 'C:\\WINDOWS\\system32'

In [2]: %ls
 Volume in drive C is BOOTCAMP
 Volume Serial Number is 0E78-D861

 Directory of C:\WINDOWS\system32

15-03-2021  20:16    <DIR>          .
15-03-2021  20:16    <DIR>          ..
07-12-2019  14:39               232 @AppHelpToast.png
07-12-2019  14:39               308 @AudioToastIcon.png
07-12-2019  14:39               330 @EnrollmentToastIcon.png
07-12-2019  14:39               404 @VpnToastIcon.png
07-12-2019  14:39               691 @WirelessDisplayToast.png
```

Contd..

We can also check methods associated with data structures by pressing the tab over the keyboard.



Python 3.7 (32-bit)

```
Type "help", "copyright", "credits" or "license" for more informa
tion.
>>> %history
  File "<stdin>", line 1
    %history
    ^
SyntaxError: invalid syntax
>>> string="ABES Engineering college"
>>> string.
```

IPython: C:WINDOWS/system32

```
Python 3.7.4 (tags/v3.7.4:e09359112e, Jul  8 2019, 19:29:22) [MSC
v.1916 32 bit (Intel)]
Type 'copyright', 'credits' or 'license' for more information
IPython 7.18.1 -- An enhanced Interactive Python. Type '?' for hel
p.

In [1]: %history
%history

In [2]: a=4

In [3]: a+5
Out[3]: 9

In [4]: %history
%history
a=4
a+5
%history

In [5]: string="ABES Engineering college"

In [6]: string.
            capitalize() expandtabs   isalpha      isnumeric
            casefold     find         isascii      isprintable
            center       format       isdecimal    isspace
            count        format_map   isdigit      istitle      >
            encode       index        isidentifier isupper
            endswith     isalnum      islower      join
```

1.2.6 Online compilation support

Assume your machine lacks the necessary resources to install, but you need to learn Python or run code to try something.

What if you could run Python online in your browser?

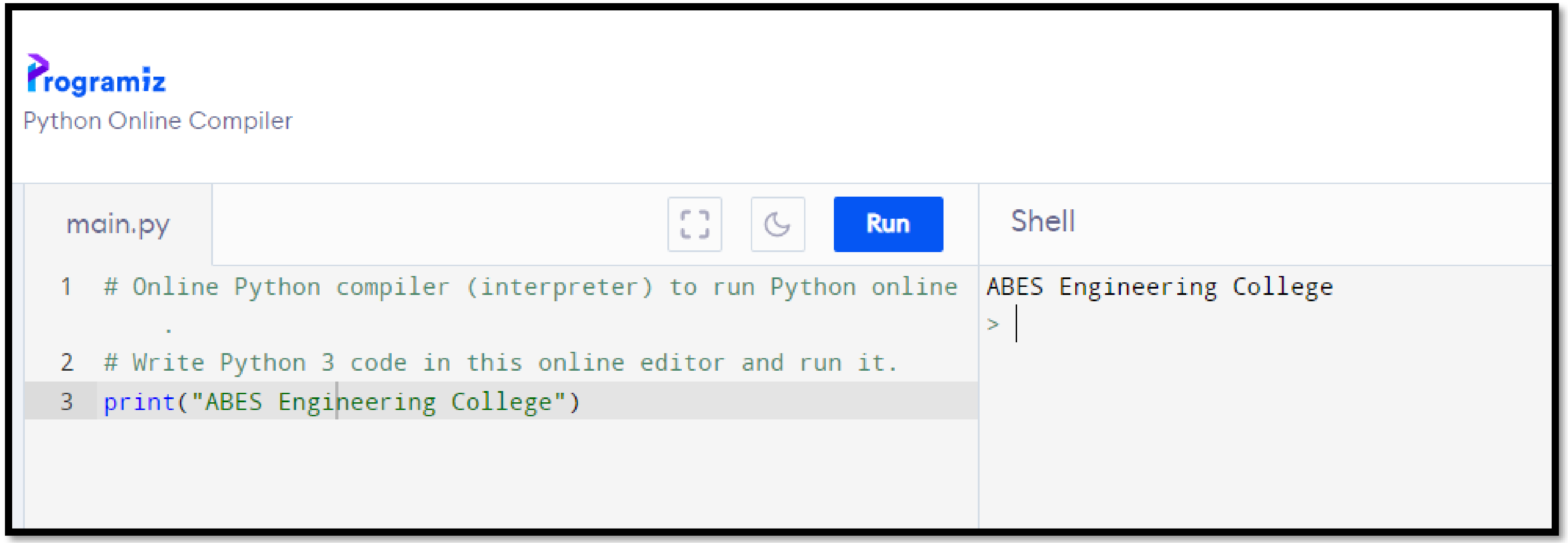
That is very great. Isn't that, right?

You will need a browser, which you already have. Using online IDEs saves yor time in the configuration process.

1.2.6 Online compilation support

There are various python online editors available in an open market. Here are the few editors:

Programiz(*https://www.programiz.com/python-programming/online-compiler/)*

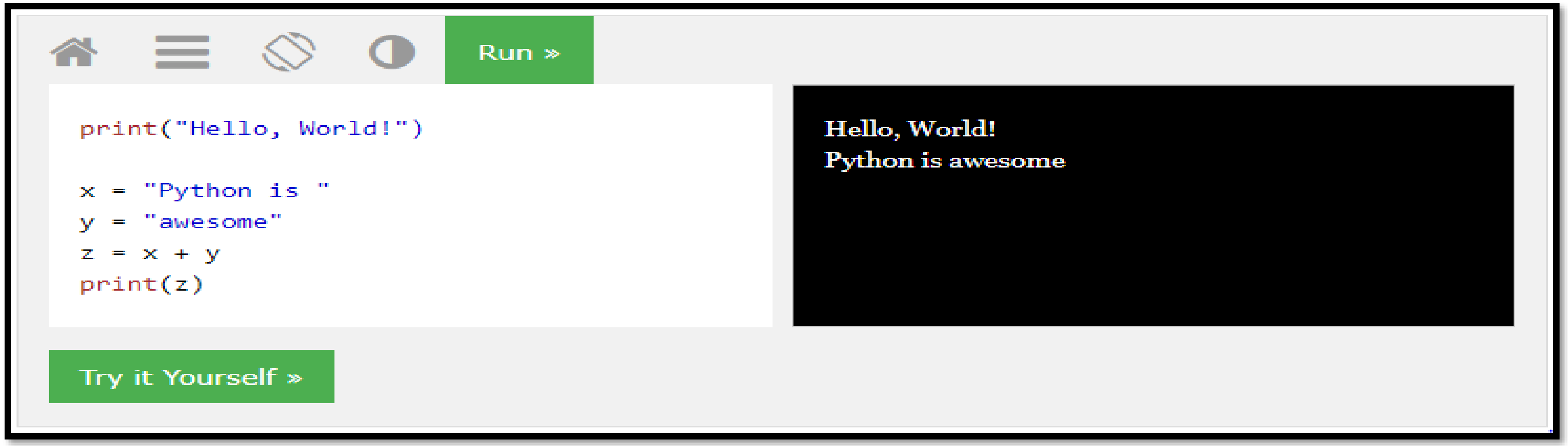


```
# Online Python compiler (interpreter) to run Python online.
# Write Python 3 code in this online editor and run it.
print("ABES Engineering College")
```

```
ABES Engineering College
>
```

Contd..

w3schools(*https://www.w3schools.com/python/python_compiler.asp*)

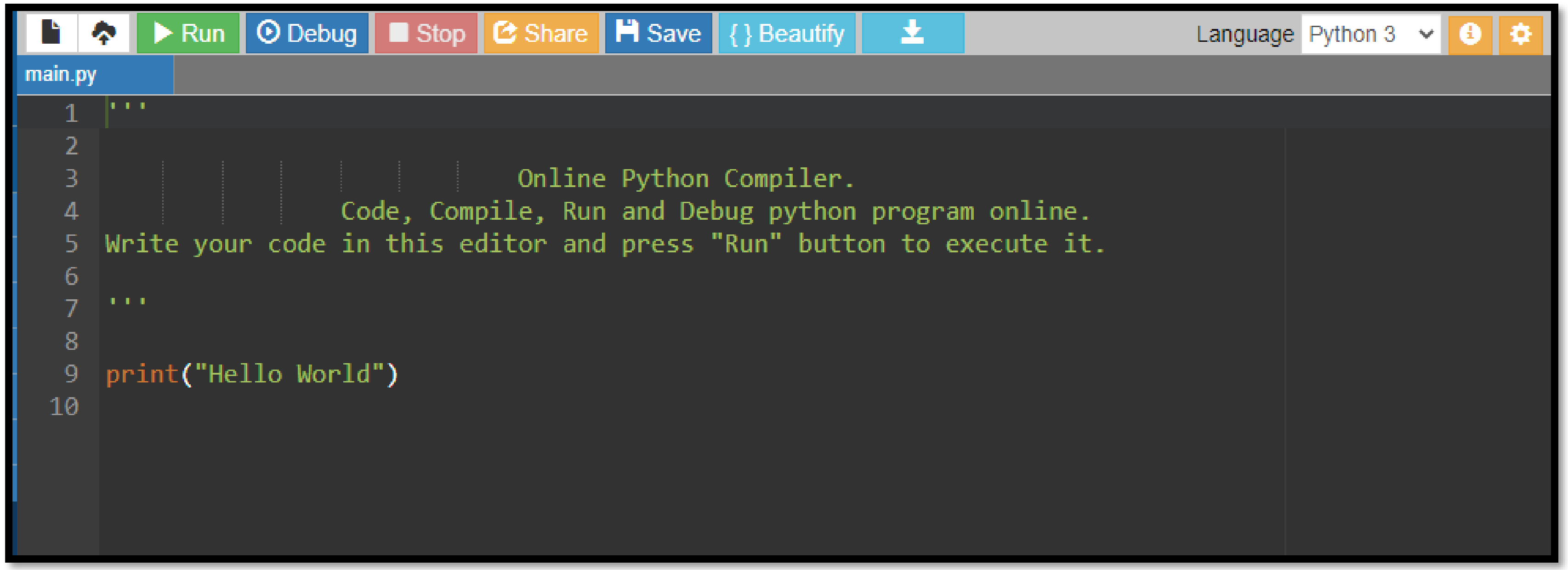


```
print("Hello, World!")

x = "Python is "
y = "awesome"
z = x + y
print(z)
```

```
Hello, World!
Python is awesome
```

Contd..

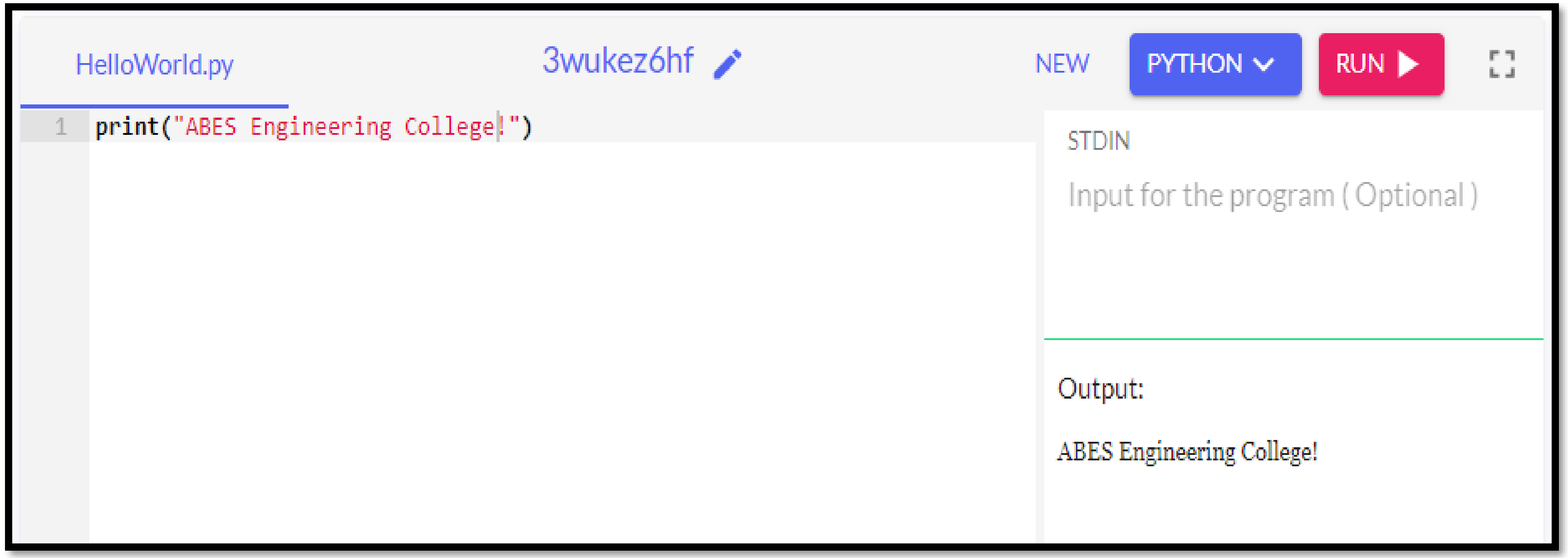
Onlinegdb *(https://www.onlinegdb.com/online_python_compiler)*



```
'''

                                Online Python Compiler.
                  Code, Compile, Run and Debug python program online.
Write your code in this editor and press "Run" button to execute it.

'''

print("Hello World")

```

Contd..

One compiler *(https://onecompiler.com/python/3wukez6hf)*



```
print("ABES Engineering College!")
```

```
ABES Engineering College!
```

Review Questions

- Which version of Python is currently up to date?
  - Python1
  - Python2
  - Python4
  - Python3

- The version of Python (if any) that comes pre-installed on your operating system is called _.
  - Onboard Python
  - Monty Python
  - Easy Python
  - System Python

Review Questions

- In a Python context, the acronym *IDLE* stands for:
  - **I**ntegrated **D**evelopment and **L**earning **E**nvironment
  - **I**nterpretive **D**ance **L**essons
  - **I**nterstellar **D**ust **L**aser **E**xplorer
  - None of the above

- When you see >>> inside IDLE, it means that:
  - An error has occurred
  - Your computer is having an existential crisis
  - Python is waiting for you to give it some instructions
  - Python is upset

Python keywords

Python keywords are **special reserved words** that have specific **meanings and purposes**.

These reserve words cannot be used as a –

- function name
- variable name
- identifiers

Note - As of python 3.9.2, there are **35 reserved** words in Python.

Contd..

The list of such keywords is mentioned below –

True	False	class	def	except
if	elif	else	try	is
raise	finally	for	in	lambda
not	from	import	global	continue
nonlocal	pass	while	break	del
and	with	as	yield	
or	assert	None	return	

Python Statement and Comments

Python Statement –

- In Python Programming, any executable instruction, that tell the computer to perform a specification action is refer to as **statements.**
- Program statement can be an
  - input-output statements,
  - arithmetic statements,
  - control statements,
  - simple assignment statements
  - and any other statements
  - it can also includes comments.

Python Statement and Comments

Python Comments –

- Comments are a set of statements that are ignored by the python interpreter.
- The use of comments makes it easy for humans to understand the source code.

Comments are of two types –

- Single-line Comments – A hash sign **(#)** is used to specify a single line comment

Example –

```
# This is a single comment
print("Hello World")  # It prints Hello World
```

- Multi-line Comments – You can do it other way using a triple quotes, either **''' '''** or **""" """** .

Example –

```
""" This is example of Multi line
Comment.
you can write any number of line in triplet
"""
```

Python Literals

Literals are the **type of data** that is used to store in a **variable or constant**.

Types of python literals:

- ❑ String literals
- ❑ Numeric Literals
  - ❑ Integer Literals
  - ❑ Float Literals
  - ❑ Complex Number Literals
- ❑ Boolean Literals
- ❑ Special literals
- ❑ Literal Collections

String literals

When set of character are enclosed in quotes (single quotes or double quotes) then it formed a string literals.

Example -

```
'abes'        'rate_of_interest'        '123',      '12.5'
"ABESEC"      "Simple_interest"         "A1"        "456 "
```

In Python we can also create multi-line literals by using '\'.

Example -

```
'ABES Engineering College \
NH-24 Delhi Hapur Bypass \
Near Crossing Republic'
```

```
"ABES Engineering College \
NH-24 Delhi Hapur Bypass \
Near Crossing Republic"
```

Numeric Literals

Numeric literals are of multiple types based on the number type. The types of numeric literals are **integer, float (decimal numbers), and complex numbers.**

Integer Literals - They can be either positive or negative.

Example –

```
12, 23000000, -45, 0 , -23987
```

Float Literals - These are basically real numbers that consist of both integer as well as fractional parts**.**

Example -

```
-12.45, 12.90, 100.0
```

Numeric Literals

Complex Number Literals - The numerals will be in the form of a + bj, where ‘a‘ is the real part and ‘b‘ is the complex part.

Python allows us to specify complex numbers like any other variable.

Example -

```
10j , 1 + 0j , 10 + 2J, 12 – 5j
```

Boolean Literals , Special literals

Boolean Literals - True or False are the values to be used as the Boolean values.

Example -

```
True, False
```

In Python, True represents the Non-zero value and False represents the value as Zero.

Special literals - Python has a special literal named None.

Note - None is used to signify the NULL value.

Literal Collections

List Literals – List is a set of values of different types. The values are separated by comma (,) and enclosed within square brackets([ ]).

Example –

```
[ 1, 23, 23.4, 100]

[ 'Blue', 'red', 123, 23.5 ]
```

Tuple literals – A tuple is a set of values of different types. The values are separated by comma(,) and enclosed within parentheses " ( ) ". It is immutable.

Example -

```
( 1, 23, 23.4, 100 )

( 'Blue', 'red', 123, 23.5 )
```

Literal Collections

Dictionary literals – It is in form of key-value pair. It is enclosed by curly-braces “{ }” and each key-value pair is separated by commas (,) .

Example –

```
{ 'name':'BOB', 'age':24, 'marks':59.4 }
```

Set literals – Set is a collection of values of different types. The values are separated by comma (,) and enclosed within curly-braces “{ }”. It is unordered and contains only unique value.

Example -

```
{ 1, 23, 45, 56, 67 }

{ 'Ram', 'Rajesh', 12, 34.5 }
```

Variables

Variable are the names given by the users to the memory locations to store the data values.
In Python, variable need not to be declared or defined in advances, as we do in many other programming language.

To create a variable, we just assign it a value and start using it.

Example –

Here ‘**a’, ‘name’ and ‘Marks’** are variable which refer an integer value 23, a string ‘Ram’ and float Value 23.5.

```
a = 23

name = 'Ram'

Marks = 23.5
```

Variables

Rules for variable name –

- ❑ Variables can be named with an alpha-numeric combination, started with an alphabet or underscore.
- ❑ Variable name can't start with digit.
- ❑ Multi- word space separated name can't be used as a variable name.
- ❑ The reserved words(keywords) cannot be used naming the variable.

Valid Variable Names

Name, num1, rate_of_interest, _abc, marks

In-Valid Variable Names

for, 123b, rate of interest, marks-math

Can you answer these questions?

1. Select all Valid variable names -

1) [✓ age]

2) [✓ _age]

3) -age

4) age_*

5) Item-Number-1

type() command

type () command helps in finding the type of the specific declared variable or a value.

Example –

```
a = 23
type(a)
```

```
int
```

```
b = 23.5
type(b)
```

```
float
```

```
name = "Ram"
type(name)
```

```
str
```

```
c = 10+2j
type(c)
```

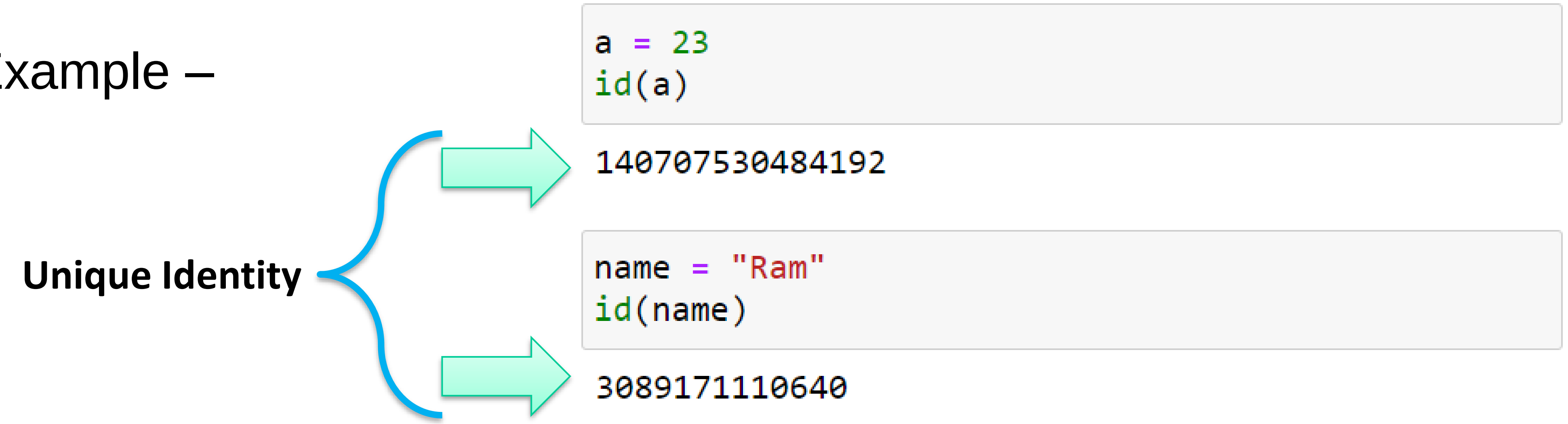
```
complex
```

id() command

id () command gives the unique id for a given objects / variable / values.

Note - The unique id is the memory address and will be different each time when you run for variable or values.

Example –



```
a = 23
id(a)
```

```
140707530484192
```

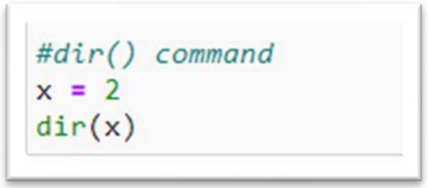
```
name = "Ram"
id(name)
```

```
3089171110640
```

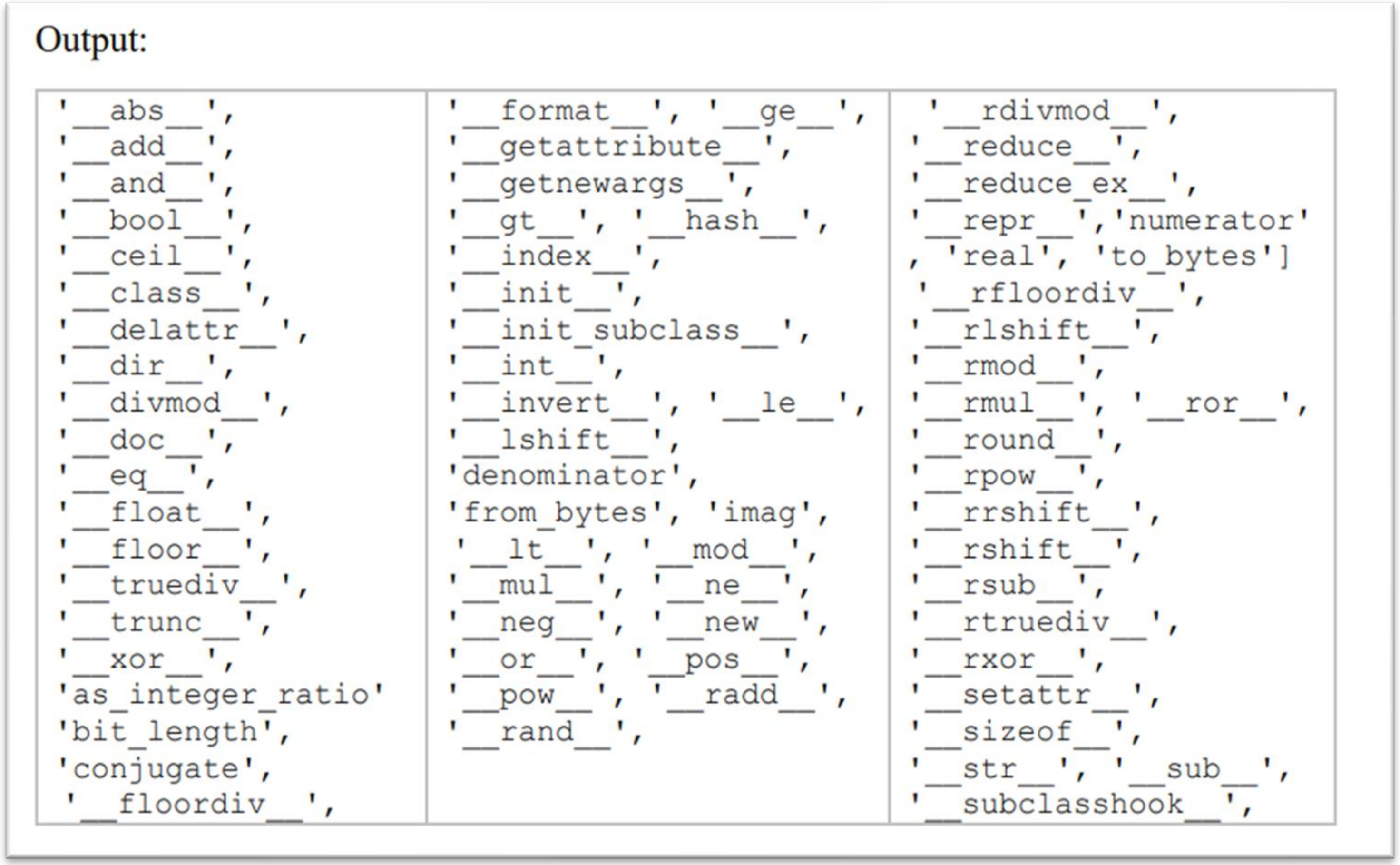
dir() command

dir () command is a vital function that returns all the properties and methods associated with given objects.

Example –



```
#dir() command
x = 2
dir(x)
```



Output:

```
'__abs__',
'__add__',
'__and__',
'__bool__',
'__ceil__',
'__class__',
'__delattr__',
'__dir__',
'__divmod__',
'__doc__',
'__eq__',
'__float__',
'__floor__',
'__truediv__',
'__trunc__',
'__xor__',
'as_integer_ratio'
'bit_length',
'conjugate',
 '__floordiv__',
```

```
'__format__', '__ge__',
'__getattribute__',
'__getnewargs__',
'__gt__', '__hash__',
'__index__',
'__init__',
'__init_subclass__',
'__int__',
'__invert__', '__le__',
'__lshift__',
'denominator',
'from_bytes', 'imag',
 '__lt__', '__mod__',
'__mul__', '__ne__',
'__neg__', '__new__',
'__or__', '__pos__',
'__pow__', '__radd__',
'__rand__',
```

```
 '__rdivmod__',
'__reduce__',
'__reduce_ex__',
'__repr__','numerator'
, 'real', 'to_bytes']
 '__rfloordiv__',
'__rlshift__',
'__rmod__',
'__rmul__', '__ror__',
'__round__',
'__rpow__',
'__rrshift__',
'__rshift__',
'__rsub__',
'__rtruediv__',
'__rxor__',
'__setattr__',
'__sizeof__',
'__str__', '__sub__',
'__subclasshook__',
```

Note – Detailed description will be explain in OOPs

Type conversion

The process of converting the value of one data type (e.g. integer, string, float, etc.) to another data type is called type conversion.

```
Example -
  12.5   → 12     ( float to integer )
 '123' → 123      ( string to integer )
  12   → 12.0     ( integer to float )
```

Python has two types of type conversion –

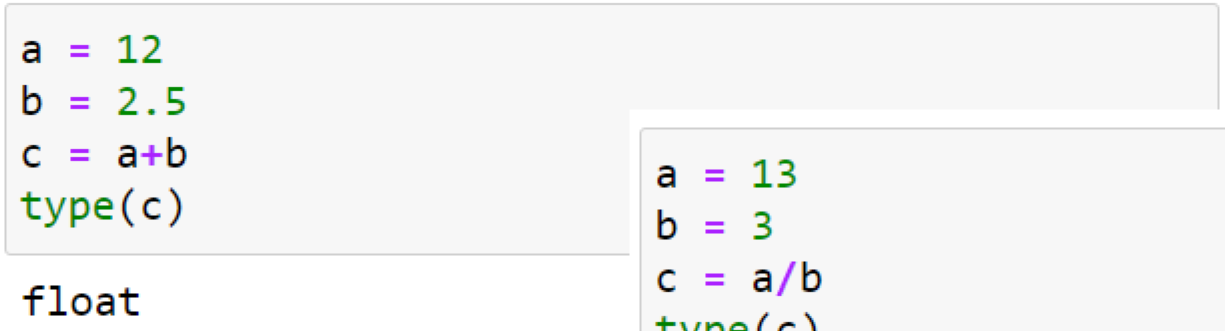
- ❑ Implicit Type Conversion
- ❑ Explicit Type Conversion

Type conversion : Implicit Type Conversion

Implicit Type Conversion – In this Python interpreter itself converts one type of data to another data type as per the performed operation.

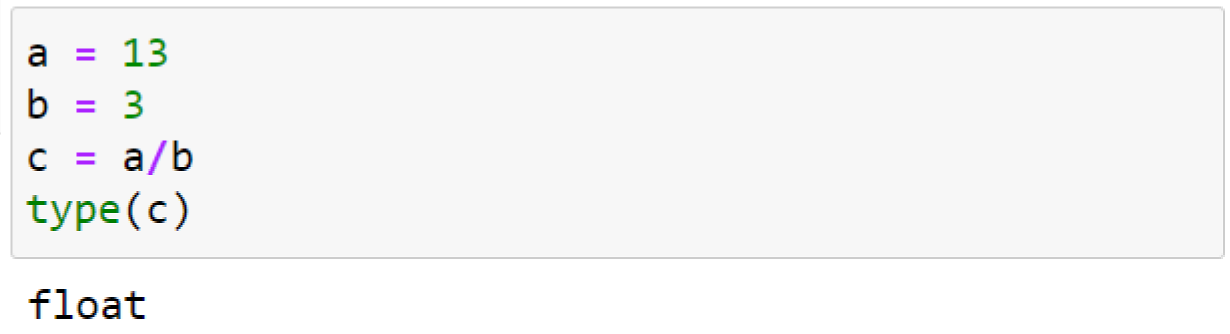
This type conversion happens automatically without any user intervention.

Example –



```
a = 12
b = 2.5
c = a+b
type(c)
```

```
float
```



```
a = 13
b = 3
c = a/b
type(c)
```

```
float
```

Note - Python promotes the conversion of the lower data type (integer) to the higher data type (float) to avoid data loss.

Type conversion : Explicit Type Conversion

Explicit Type Conversion – In this, user converts the data type of variable of value to needed data type.

Example –

```
a = 13
b = 3
c = int(a/b)
type(c)
```

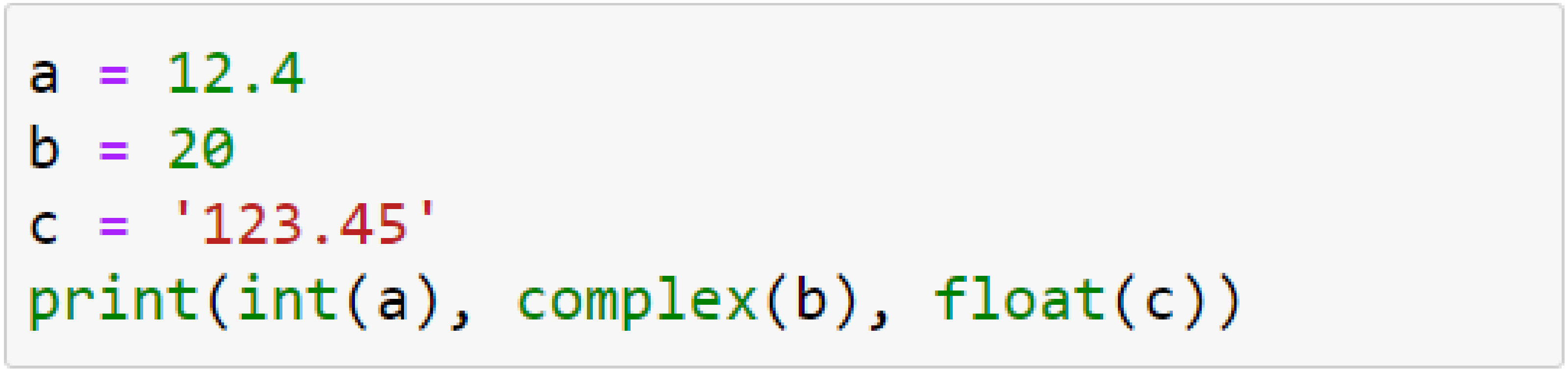
```
int
```

Note – There should be valid Numbers while converting any string to Numbers.

int() → for Integer
float() → for Float
str() → for string
Complex → for complex

Can you answer these questions?

1. What will be the output of the following -



```
a = 12.4
b = 20
c = '123.45'
print(int(a), complex(b), float(c))
```

a) [✓ 12 (20+0j) 123.45]

b) 12 20 123.45

c) Error

Basic I/O Operations

In python, various built-in functions are present.

The two important standard input-output functions in python are:

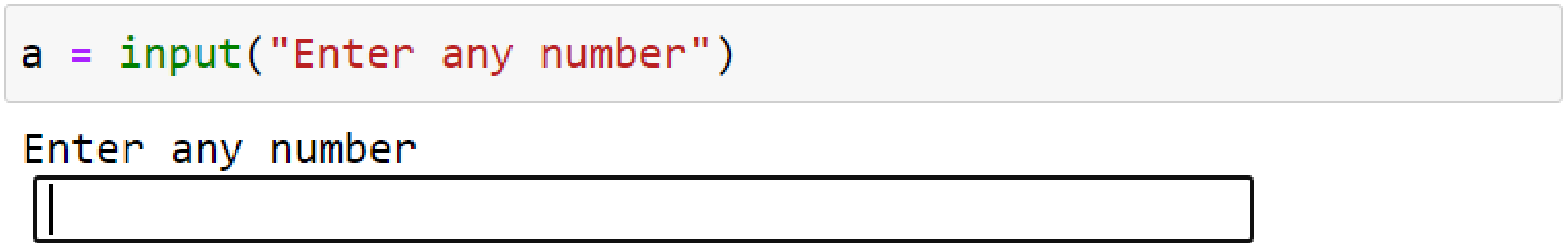
- ❑ input() : to take the input from the user
- ❑ print() : to show the output on the console

Basic I/O Operations : input()

Syntax –

```
input(prompt='')
Prompt - A String, representing a default message
before the input.
```

Example –



```
a = input("Enter any number")
```

```
Enter any number
```

Note – It return value as a string. So you need to typecast it.

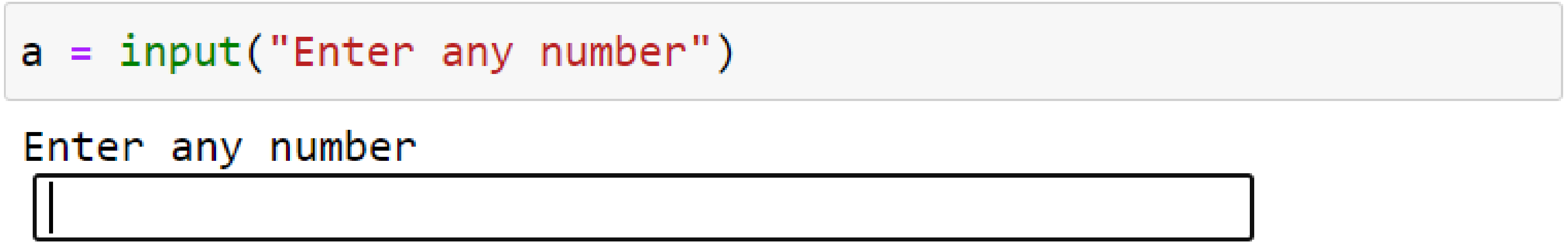
Basic I/O Operations : input()

Syntax –

```
input(prompt='')
```

Prompt – A String, representing a default message before the input.

Example –



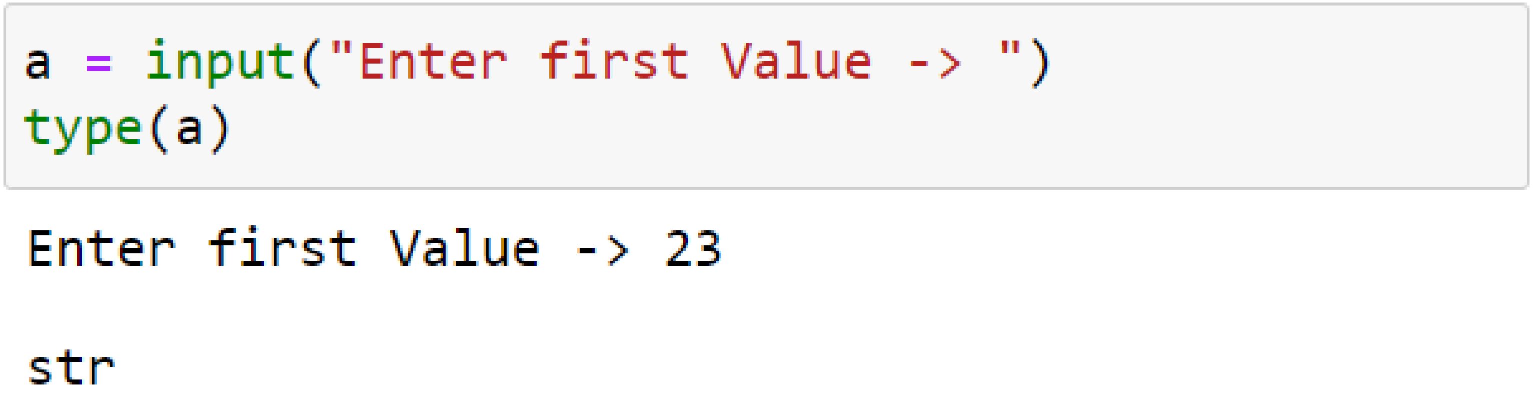
```
a = input("Enter any number")
```

```
Enter any number
```

Basic I/O Operations : input()

When we take input using input() function, it return value in string data type.

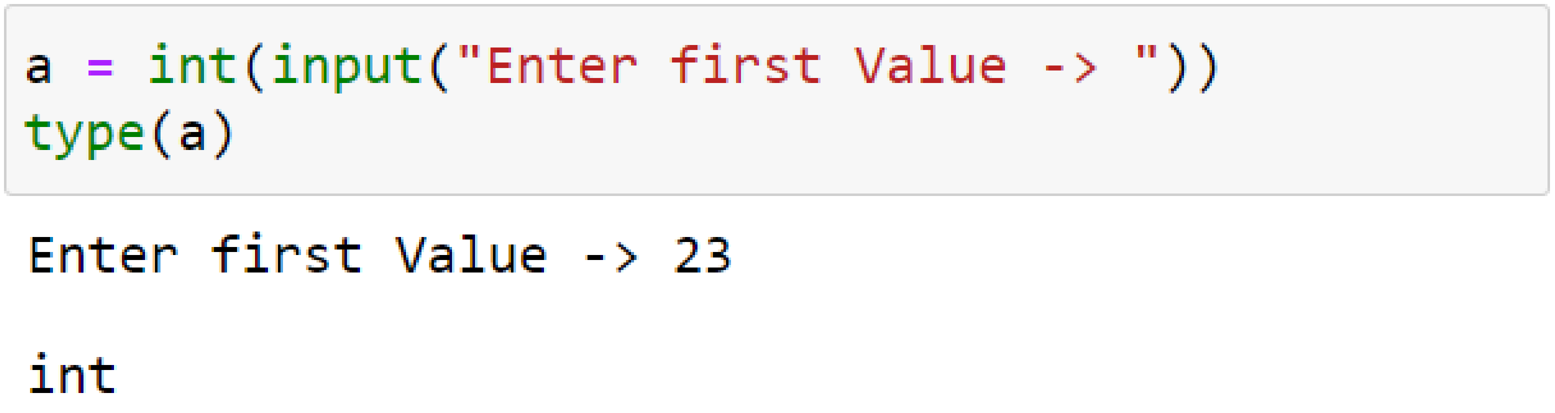
Example –



```
a = input("Enter first Value -> ")
type(a)
```

```
Enter first Value -> 23

str
```

As we can see in the above example type of “a” is string



```
a = int(input("Enter first Value -> "))
type(a)
```

```
Enter first Value -> 23

int
```

Note – We have to typecast input value into desired type.

Basic I/O Operations : print()

print () is a built-in standard function used to print the output to the console.

Syntax –

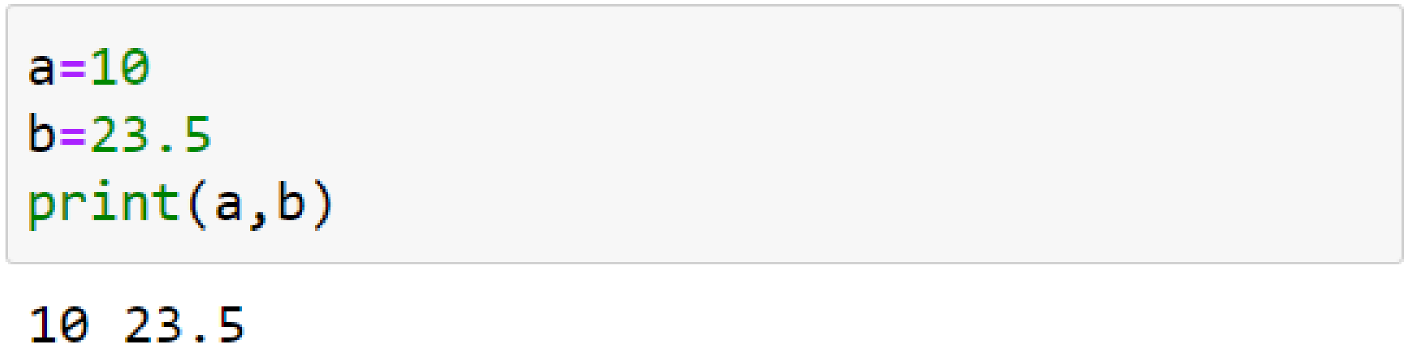
```
print(value, ..., sep='', end='\n’)
```

- ❑ value – Can be of any literals, variable, expression, statements
- ❑ sep - (optional), Specify how to separate the values, if there is more than one. Default is ' '.
- ❑ end - (optional), Specify what to print at the end. Default is '\n‘

Note – We can assign any set of character into sep.

Basic I/O Operations : print()

Example –

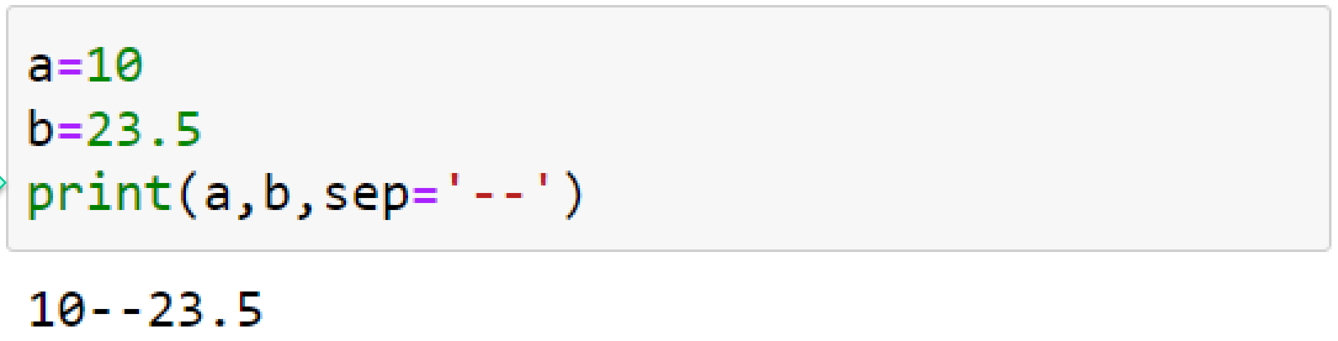


```
a=10
b=23.5
print(a,b)
```

```
10 23.5
```

In the above example value of a and b is separated by space i.e. default

Here separator is sep='--', so both value is separated by '--‘ as shown in output.



```
a=10
b=23.5
print(a,b,sep='--')
```

```
10--23.5
```

Basic I/O Operations : print()

Example – Write a Python program that takes two integer as input from user and print sum of both.

```python
a = int(input("Enter first Value -> "))
b = int(input("Enter second Value -> "))
c = a+b
print(c)
```

```
Enter first Value -> 10
Enter second Value -> 20
30
```

Can you answer these questions?

What will be the output of the following -

a) **12 23.5 ABESEC**

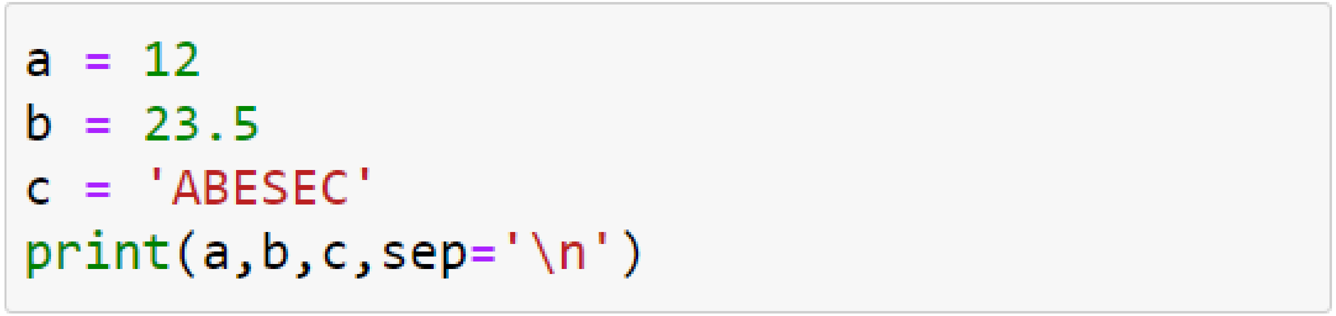
b) **12\n23.5\nABESEC**

[✓ **c)**]

12

23.5

ABESEC



```
a = 12
b = 23.5
c = 'ABESEC'
print(a,b,c,sep='\n')
```

Expressions

An **expression is a combination of operators and operands** that is interpreted to **produce some other value**.

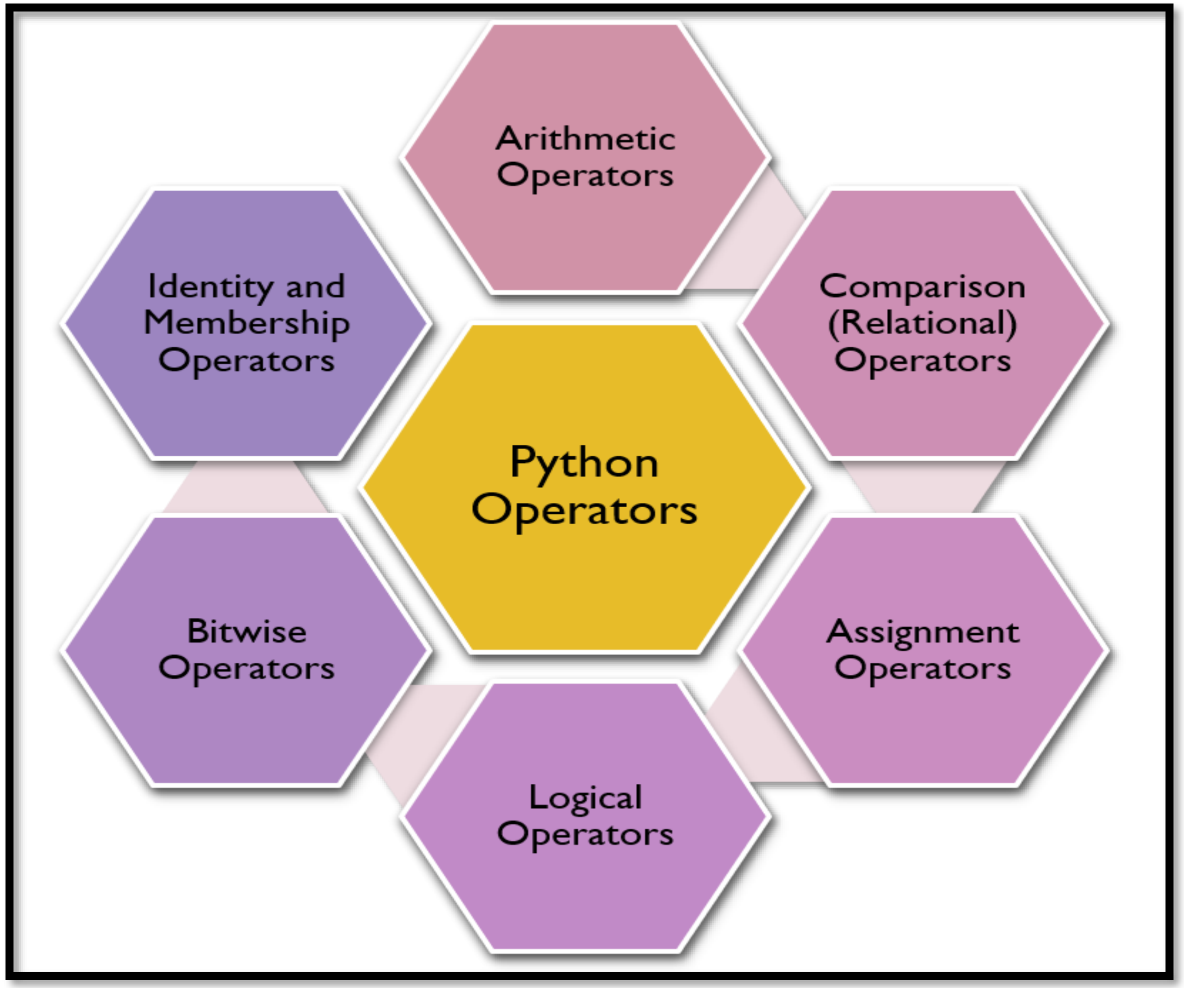
In any programming language, an expression is evaluated as per the precedence of its operators.

So that if there is more than one operator in an expression, their precedence decides which operation will be performed first.

Operators

Operators are symbol, used to perform mathematical and logical operation.

In python operators are categorized into six categories -



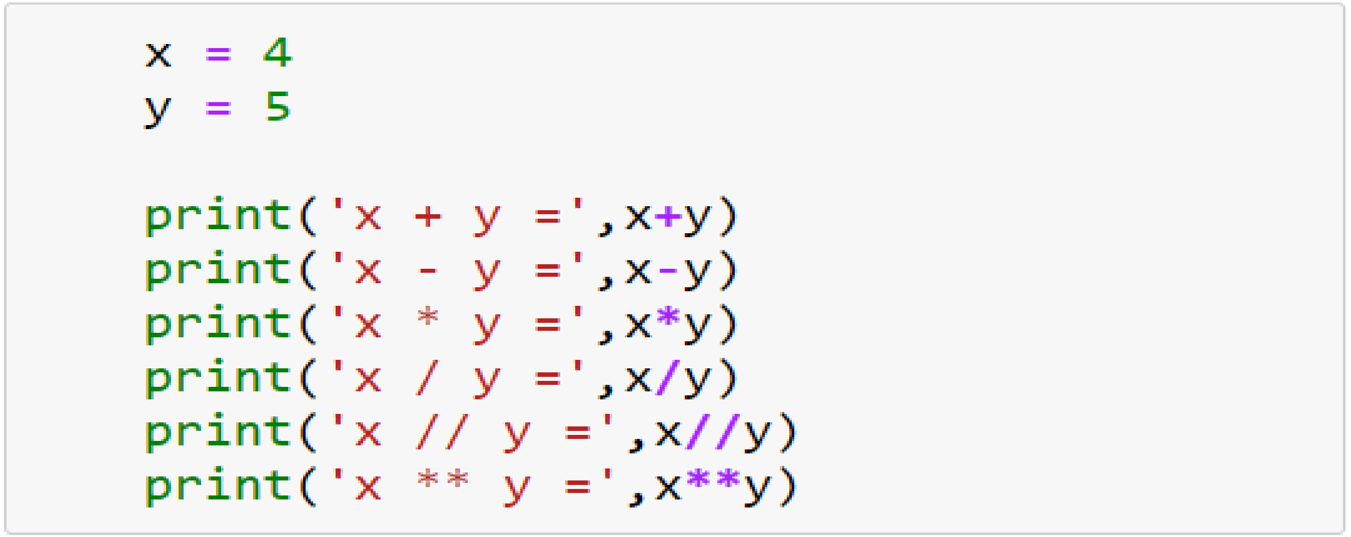
Arithmetic operators

There are seven arithmetic operators, and these are of:

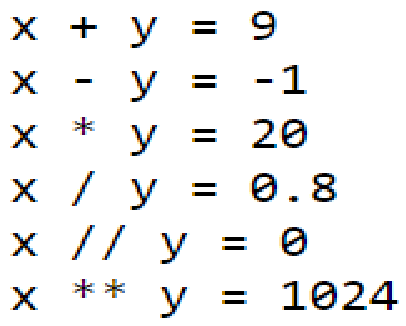
Operator	Meaning	Example
+	Add two operands or unary plus	x + y +2
-	Subtract right operand from the left or unary minus	x - y -2
*	Multiply two operands	x * y
/	Divide left operand by the right one (always results into float)	x / y
%	Modulus - remainder of the division of left operand by the right	x % y (remainder of x/y)
//	Floor division - division that results into whole number adjusted to the left in the number line	x // y
**	Exponent - left operand raised to the power of right	x**y (x to the power y)

Arithmetic operators

Example of all mentioned arithmetic operators



```python
x = 4
y = 5

print('x + y =',x+y)
print('x - y =',x-y)
print('x * y =',x*y)
print('x / y =',x/y)
print('x // y =',x//y)
print('x ** y =',x**y)
```



```
x + y = 9
x - y = -1
x * y = 20
x / y = 0.8
x // y = 0
x ** y = 1024
```

Comparison (Relational) Operators

The comparison operators are used for comparisons.

Comparison operators compare two values and evaluate down to a single **Boolean value (True / False).**

Operator	Meaning	Example
>	Greater than -> True if left operand is greater than the right	x > y
<	Less than -> True if left operand is less than the right	x < y
==	Equal to -> True if both operands are equal	x == y
!=	Not equal to -> True if operands are not equal	x != y
>=	Greater than or equal to -> True if left operand is greater than or equal to the right	x >= y
<=	Less than or equal to -> True if left operand is less than or equal to the right	x <= y

Comparison (Relational) Operators

Example of all mentioned comparison operators

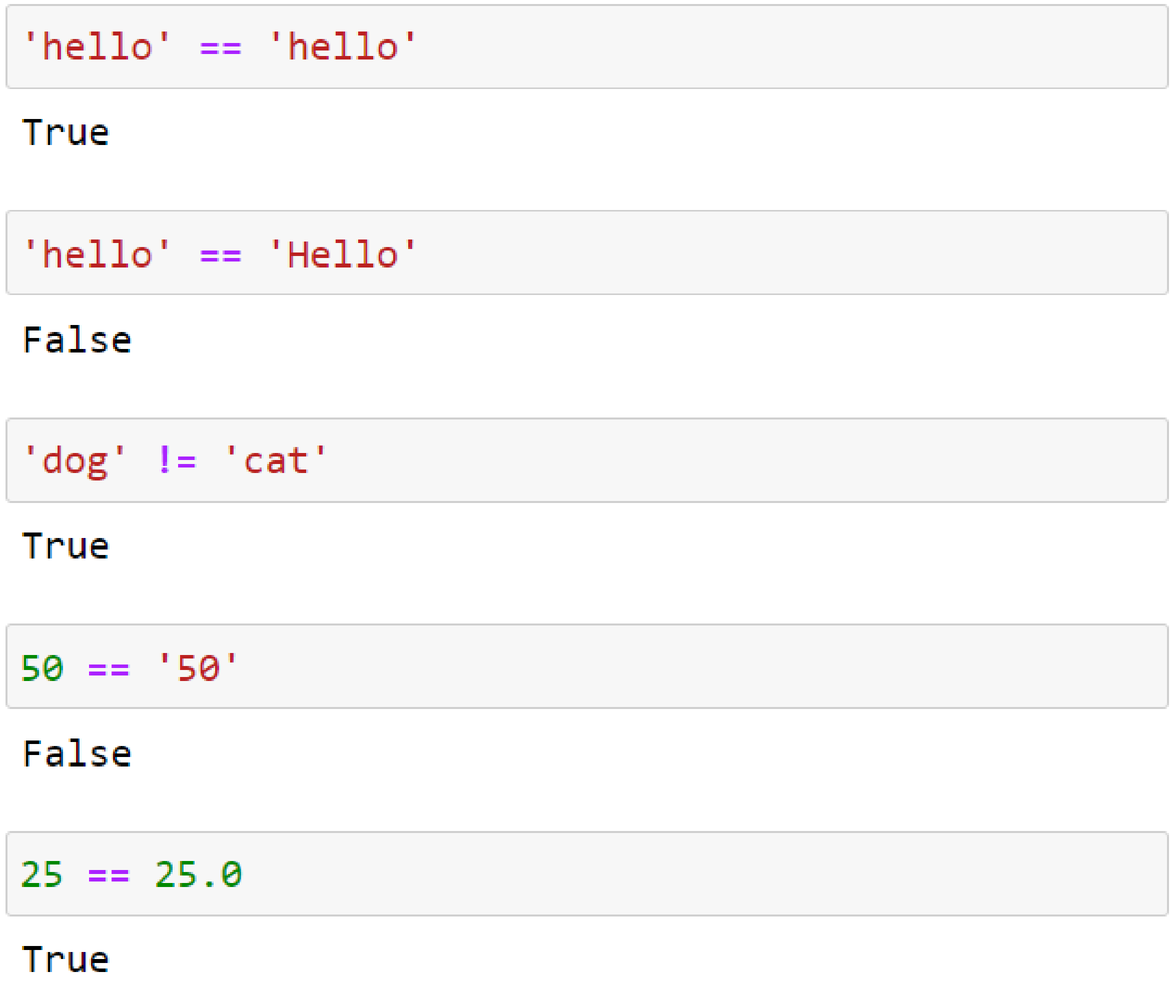
It always gives answer either **True or False** depending upon relation.

```
a = 10
b = 20

print(a > b)
print(a < b)
print(a == b)
print(a != b)
print(a >= b)
print(a <= b)
```

```
False
True
False
True
False
True
```

Comparison (Relational) Operators

Example –



```
'hello' == 'hello'
```

```
True
```

```
'hello' == 'Hello'
```

```
False
```

```
'dog' != 'cat'
```

```
True
```

```
50 == '50'
```

```
False
```

```
25 == 25.0
```

```
True
```

Note - The == and != operators can actually work with values of any data type.

Bitwise Operators

Bitwise operators act on the bits and performs bit by bit operation on the operands.

Example – Evaluate **2 & 7**

How it works -

Step 1 – Convert 2 in binary → 0010
Step 2 – Convert 7 in binary → 0111
Step 3 – Perform Bitwise & operation
Step 4 – Convert the result back to decimal

Operator	Meaning
&	Bitwise AND
|	Bitwise OR
~	Bitwise NOT
^	Bitwise XOR
>>	Bitwise right shift
<<	Bitwise left shift

Bitwise Operators – Cont..

Truth Table

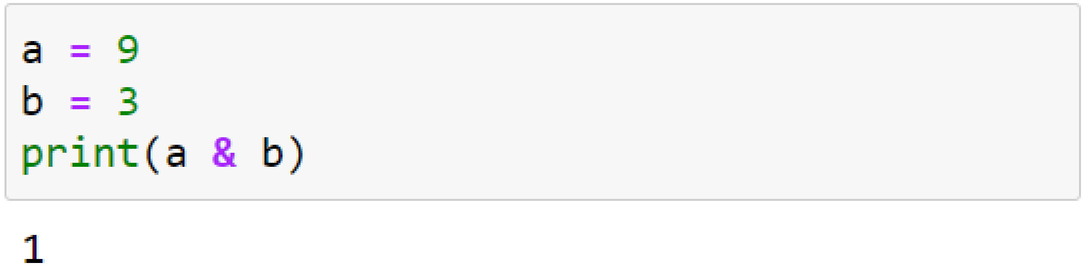
A	B	A & B	A | B	A ^ B
0	0	0	0	0
0	1	0	1	1
1	0	0	1	1
1	1	1	1	0

Bitwise NOT

A	~A
0	1
1	0

Bitwise Operators – Bitwise AND (&)

Example –



```
a = 9
b = 3
print(a & b)
```

```
1
```

Explanation –

Step 1 – Convert 9 in binary → 1001
Step 2 – Convert 3 in binary → 0011
Step 3 –

```
1001
0011
----
0001
```

Step 4 – Convert the result (0001) back to decimal → 1

Bitwise Operators – Bitwise OR(|)

Example –

```
a = 9
b = 3
print(a | b)
```

```
11
```

Explanation –

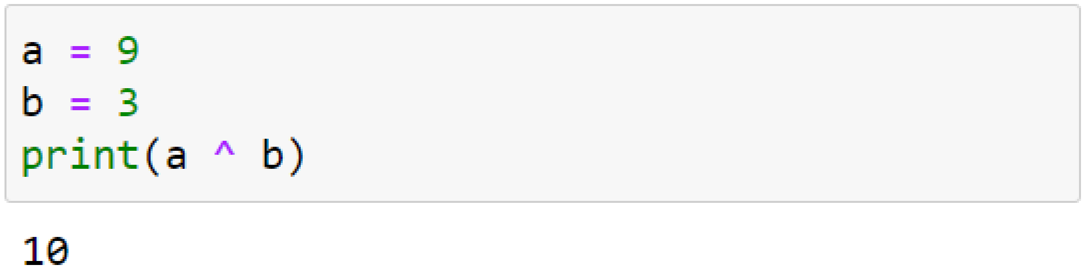
Step 1 – Convert 9 in binary → 1001
Step 2 – Convert 3 in binary → 0011
Step 3 –

$$\begin{array}{r} 1001 \\ 0011 \\ \hline 1011 \end{array}$$

Step 4 – Convert the result (1011) back to decimal → 11

Bitwise Operators – Bitwise XOR (^)

Example –



```
a = 9
b = 3
print(a ^ b)
```

```
10
```

Explanation –

Step 1 – Convert 9 in binary → 1001
Step 2 – Convert 3 in binary → 0011
Step 3 –

```
1001
0011
────
1010
```

Step 4 – Convert the result (1010) back to decimal → 10

Bitwise Operators – Bitwise Left shift (<<)

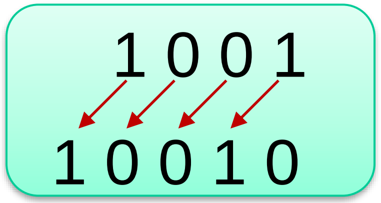
Example –

```
print(9<<1)
```

```
18
```

Explanation –

Step 1 – Convert 9 in binary → 1001
Step 2 – Shift 1001 towards left by one position and place Zero at end.



Step 3 – Convert the result (10100) back to decimal → 18

Bitwise Operators – Bitwise Right Shift (>>)

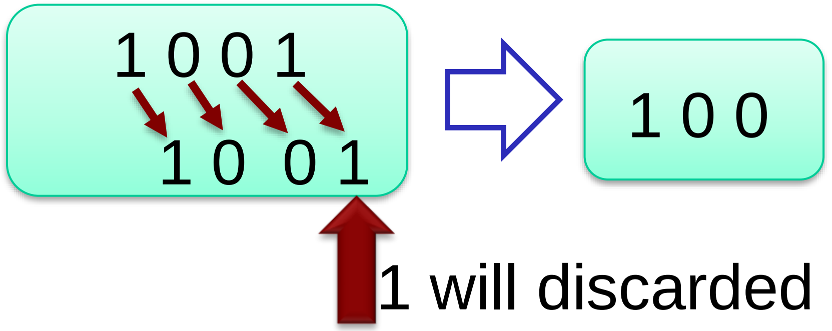
Example –

```
print(9>>1)
```

```
4
```

Explanation –

Step 1 – Convert 9 in binary → 1001
Step 2 – Shift 1001 towards right by one position.



Step 3 – Convert the result (100) back to decimal → 4

Assignment operators

Assignment operators are used to assign the values to the variables.

a = 5 is a simple assignment operator that assigns the **value 5** on the right to the variable **a on the left.**

There are various compound operators in Python like a += 5

Operator	Example	Equivalent to
=	x = 5	x = 5
+=	x += 5	x = x + 5
-=	x -= 5	x = x - 5
*=	x *= 5	x = x * 5
/=	x /= 5	x = x / 5
%=	x %= 5	x = x % 5
//=	x //= 5	x = x // 5
**=	x **= 5	x = x ** 5
&=	x &= 5	x = x & 5
|=	x |= 5	x = x | 5
^=	x ^= 5	x = x ^ 5
>>=	x >>= 5	x = x >> 5
<<=	x <<= 5	x = x << 5

Logical Operators

Logical operators perform Logical AND, Logical OR and Logical NOT operations.

Operator	Meaning	Example
and	True if both the operands are true	x and y
or	True if either of the operands is true	x or y
not	True if operand is false (complements the operand)	not x

Truth Table –

x	y	x and y	x or y
T	T	T	T
T	F	F	T
F	T	F	T
F	F	F	F

x	not x
T	F
F	T

Logical Operators

Example –

```python
a=10
b=20
print(a<b and a!=b)
print(a>b or b>a)
print(not a)
```

```
True
True
False
```

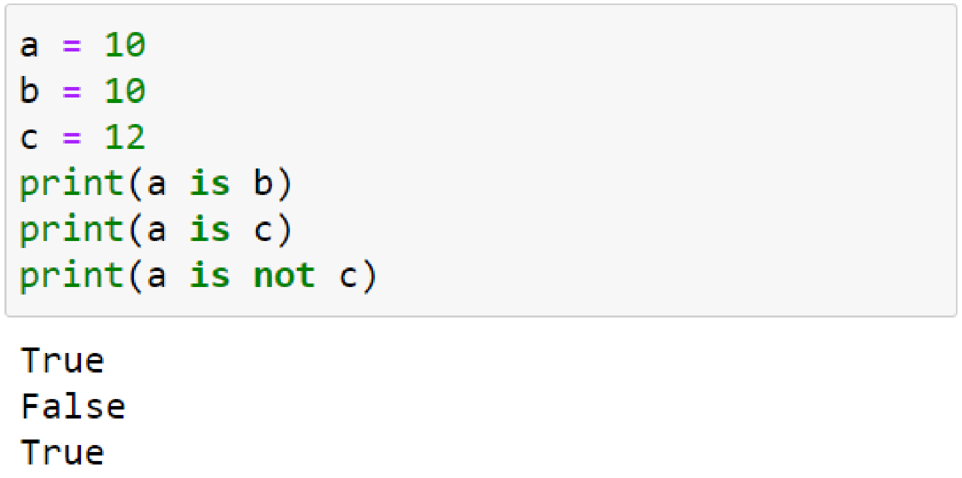
Identity Operators

Identity Operators – is and **is not** are the identity operators in Python.

They are used to check if two values (or variables) are located on the same part of the memory or not.

Operator	Meaning
is	Gives **True** if the operands are identical (refer to the same ID or Memory)
is not	Gives True if the operands are not identical (do not refer to the ID or Memory)

Example



```
a = 10
b = 10
c = 12
print(a is b)
print(a is c)
print(a is not c)
```

```
True
False
True
```

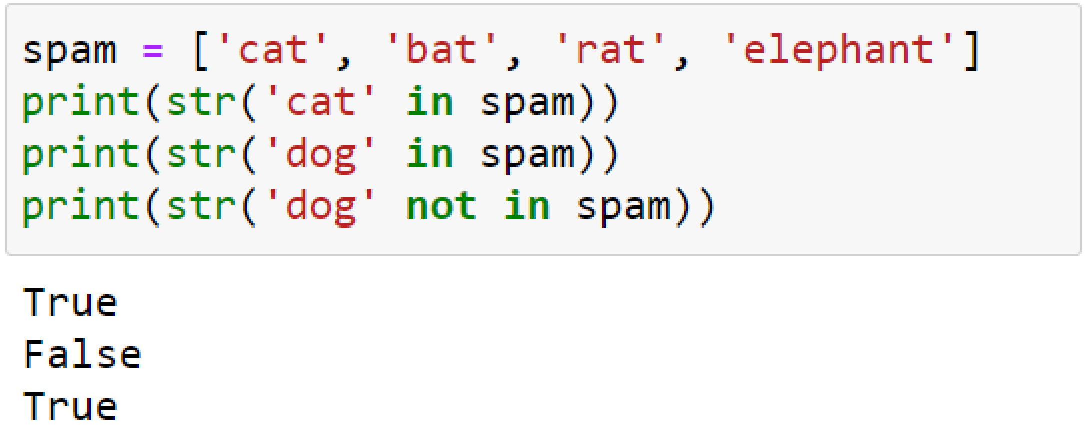
Membership Operators

in and not in are the membership operators in Python.

They are used to test whether a value or variable is found in a sequence (string, list, tuple, set and dictionary) or not.

Operator	Meaning
in	Gives **True** if value/variable is found in the sequence otherwise **False**
not in	Gives **True** if value/variable is not found in the sequence otherwise **True**

Example



```
spam = ['cat', 'bat', 'rat', 'elephant']
print(str('cat' in spam))
print(str('dog' in spam))
print(str('dog' not in spam))
```

```
True
False
True
```

Boolean Operators

- A boolean expression is an expression that yields just the two outcomes: **true or false**.
- When we work with multiple boolean expressions or perform some action on them, we make use of the boolean operators.
- Since the boolean expression reveals true or false, the operations on these expressions also result in either **"true"** or **"false".**
- Consequently, there are three types of boolean operators:
  - The AND operator (&& or "and")
  - The OR operator (|| or "or")
  - The NOT operator (not)

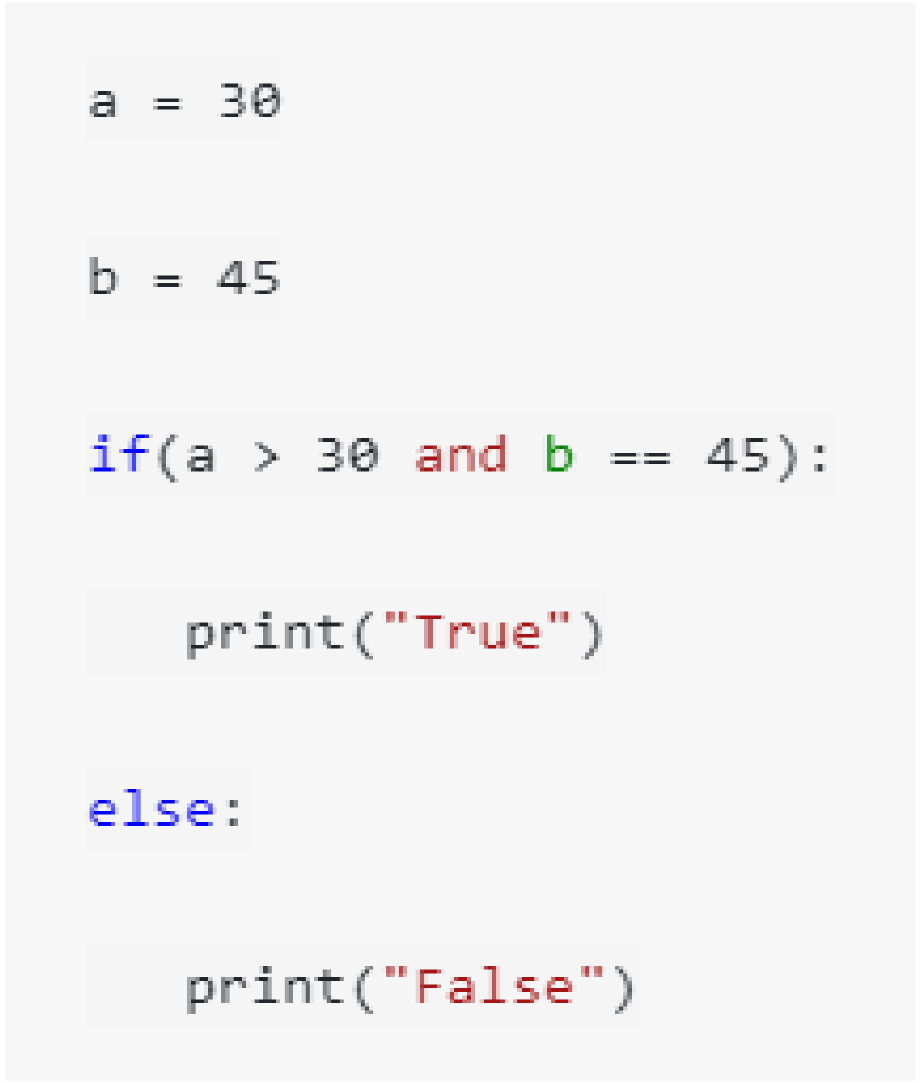
Boolean Operators

AND Boolean Operator in Python

The **AND boolean operator** is similar to the bitwise **AND operator** where the operator analyzes the expressions written on both sides and returns the output.

- True and True = True
- True and False = False
- False and True = False
- False and False = False

- Output: False



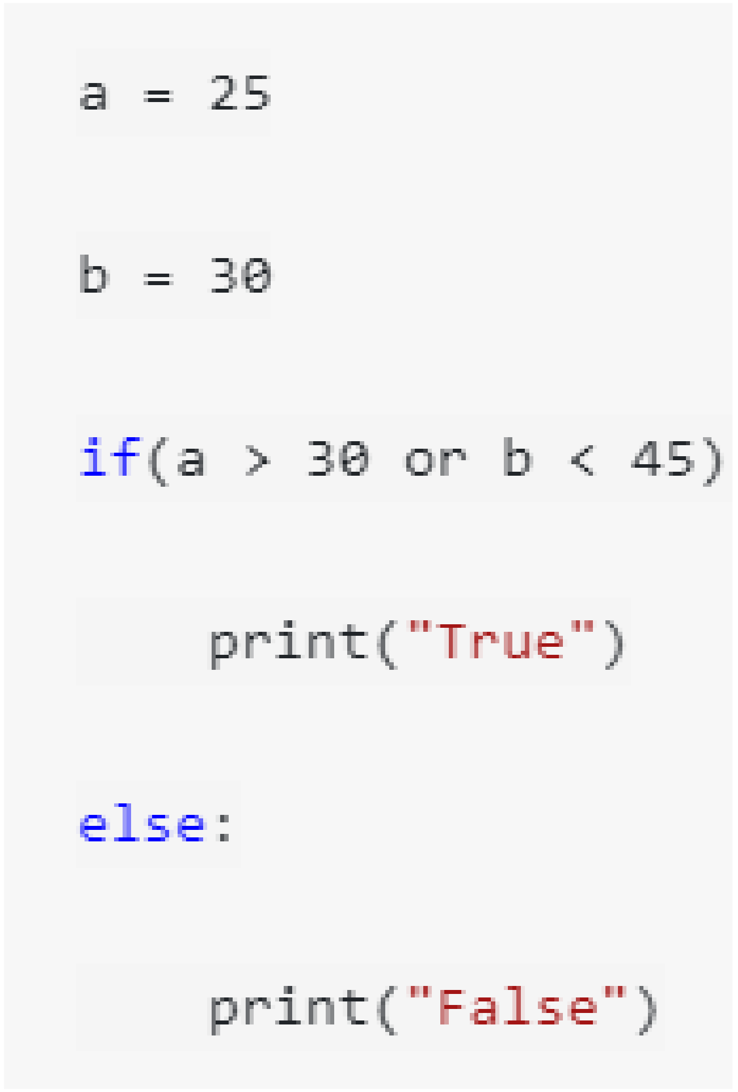
```
a = 30

b = 45

if(a > 30 and b == 45):

    print("True")

else:

    print("False")
```

Boolean Operators

The **OR operator** is similar to the **OR bitwise operator**. In the bitwise OR, we were focussing on either of the bit being 1. Here, we take into account if either of the expression is true or not. If at least one expression is true, consequently, the result is true.

- True or True = True
- True or False = True
- False or True = True
- False or False = False

- Output: True



```
a = 25

b = 30

if(a > 30 or b < 45)

    print("True")

else:

    print("False")
```

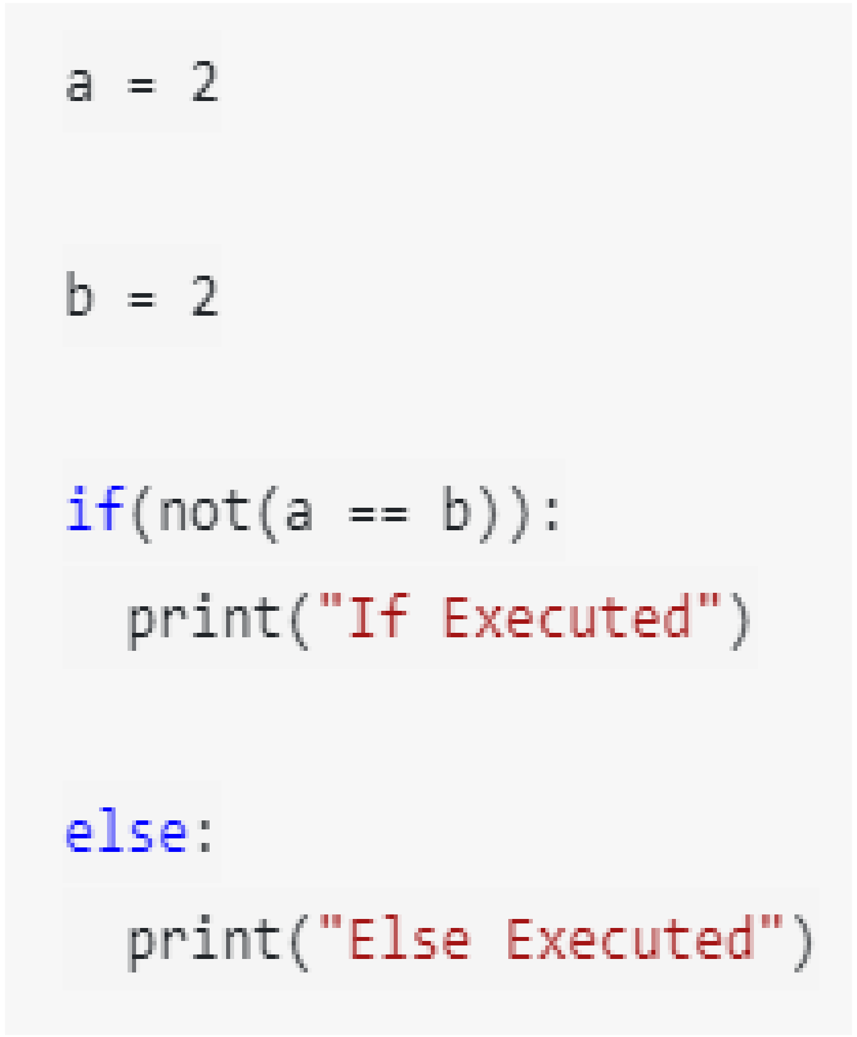
Boolean Operators

The **NOT operator** reverses the result of the boolean expression that follows the operator. It is important to note that the NOT operator will only reverse the final result of the expression that **immediately follows.** Moreover, the NOT operator is denoted by the keyword **"not".**

•not(True) = False

•not(False) = True

•Output: Else Executed



```
a = 2

b = 2

if(not(a == b)):
  print("If Executed")

else:
  print("Else Executed")
```

Precedence and associativity

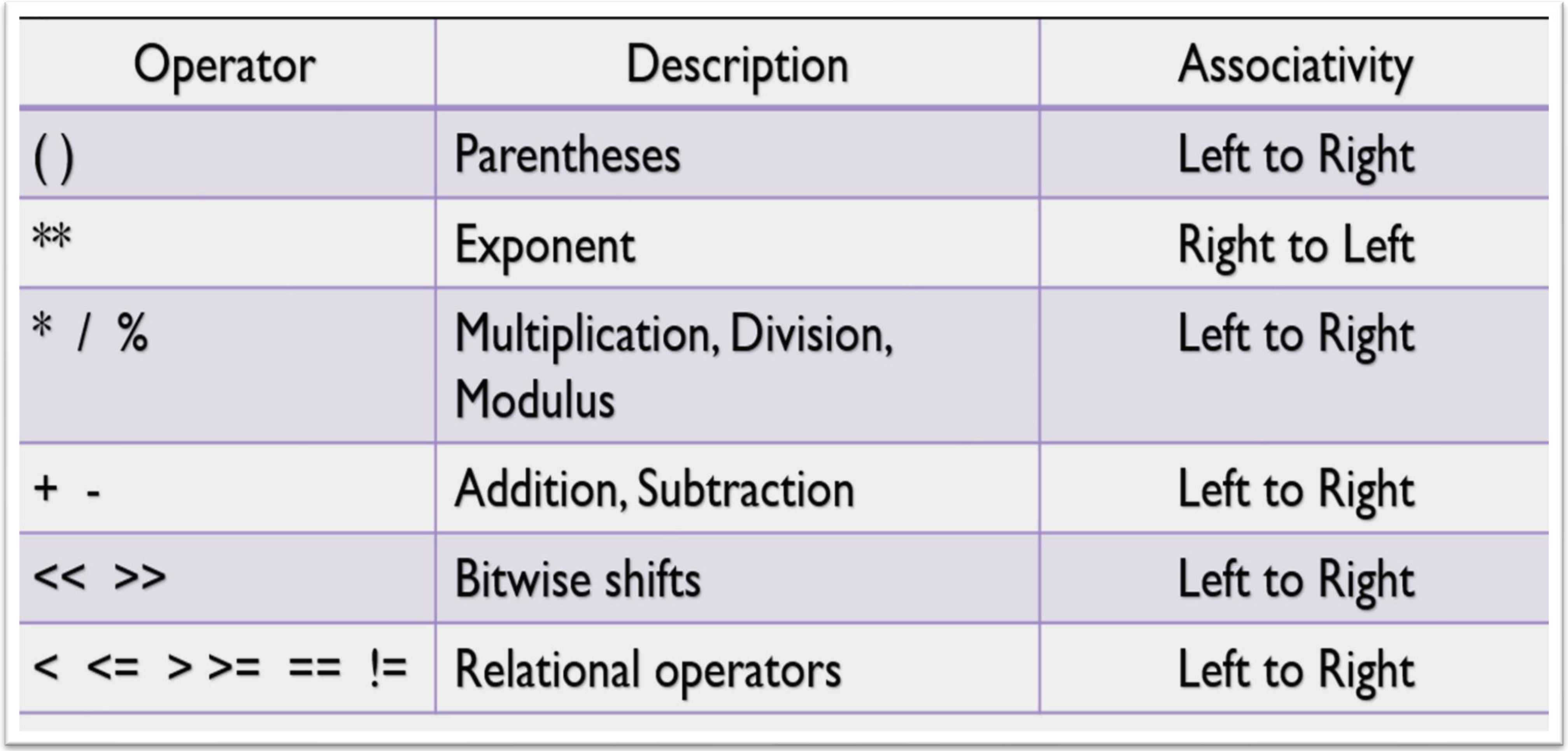
Operator precedence and associativity decide the priorities of the operator.

- ❑ **Operator Precedence:** This is used in an expression with more than one operator with different precedence to figure out which operation to perform first.

- ❑ **Operator Associativity:** If an expression has two or more operators with the same precedence, then Operator Associativity is used to find. It can either be Left to Right or from Right to Left.

Cont..

Precedence and Associativity Table –



Operator	Description	Associativity
( )	Parentheses	Left to Right
**	Exponent	Right to Left
* / %	Multiplication, Division, Modulus	Left to Right
+ -	Addition, Subtraction	Left to Right
<< >>	Bitwise shifts	Left to Right
< <= > >= == !=	Relational operators	Left to Right

Python 2 vs Python 3

- ❑ Python has started its journey in 1989-1990 when people started implementation on it.
- ❑ In year 2000, python 2.0 came with new features and have a healthy support to python.
- ❑ Memory management was the major part evolved in python 2.0.
- ❑ But in 2008, python has changed in a revolutionary manner to python 3.0.
- ❑ There was no support of backward compatibility in python 3.0.

Python 2 vs Python 3

Let us have a look to the differences between Python 2 and Python 3.

- In Python 2, **print is a statement** syntax, but in python 3, **print is a built-in function.**
- In python 2, the input was taken from the user by using **raw_input() function**. In python 3, **input () function** is used to take input instead of raw_input().
- When we divide two numbers in python 2, the output is the nearest whole number. Like 7/2 is 3. In python 3, the fractional numeric value will be shown as 7/2 is 3.5.

Python 2 vs Python 3

Let us have a look to the differences between Python 2 and Python 3.

- In for loop, the iterations are used using a **xrange function** in python 2, which is replaced by **range function** in python3.
- Some of the libraries which are available in python 2 are not moved in python 3.
- Similarly, now the developers are making new libraries for python 3 which are incompatible to python 2.

References

1. https://docs.python.org/3/tutorial/controlflow.html
2. Think Python: An Introduction to Software Design, Book by Allen B. Downey
3. Head First Python, 2nd Edition, by Paul Barry
4. Python Basics: A Practical Introduction to Python, by David Amos, Dan Bader, Joanna Jablonski, Fletcher Heisler
5. https://www.fullstackpython.com/turbogears.html
6. https://www.cubicweb.org
7. https://pypi.org/project/Pylons/
8. https://www.upgrad.com/blog/python-applications-in-real-world/
9. https://www.codementor.io/@edwardbailey/coding-vs-programming-what-s-the-difference-yr0aeug9o

Thank You