|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  | | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | | mgm logo.jpeg  **MGM's College of Engineering and Technology**  **Kamothe, Navi Mumbai-410209**  **Department of Computer Engineering**  **Laboratory Manual**  **Skill Base Lab Course - Python Programming**  **Course Code: CSL405**  **Second Year**  **Semester- IV**  **Academic Year 2021-22**  **CONTENTS**   |  |  |  | | --- | --- | --- | | **S. No.** | **TOPIC** | **PAGE NO.** | | 1 | Vision and Mission of College and Department | 3 | | 2 | Teaching Scheme, Examination Scheme and University Syllabus | 4 | | 3 | Lab Objectives and Lab Outcomes (COs) | 5 | | 4 | Program Specific Outcomes (PSOs) & Lab Outcomes (POs) | 6 | | 5 | CO & PO Mapping | 8 | | 6 | List of Experiments | 9 | | 7 | Experiment Plan | 10 | | 8 | Experiment 01:Introduction to Python & basic programming concepts in Python.(Downloading and Installation of Python) |  | | 9 | Experiment 02: Exploring basics of python like data types (strings, list, array, dictionaries, set, tuples) and control statements | 11 | | 10 | Experiment 03: Creating functions, classes and objects using python. Demonstrate exception handling and inheritance. | 15 | | 11 | Experiment No. 4: Exploring Files and directories  4.a. Python program to append data to existing file and then display the entire file  4.b. Python program to count number of lines, words and characters in a file  4.c. Python program to display file available in current directory | 19 | | 12 | Experiment No 5: Creating GUI with python containing widgets such as labels, textbox, radio, checkboxes and custom dialog boxes | 24 | | 13 | Experiment No 6: Menu driven program for data structure using built in function for link list, stack and queue. | 27 | | 14 | Experiment No 7: Program to demonstrate CRUD (create, read, update and delete) operations on database (SQLite/ MySQL) using python. | 31 | | 15 | Experiment No 8: Creation of simple socket for basic information exchange between server and client. | 35 | | 16 | Experiment No 9: Creating web application using Django web framework to demonstrate functionality of user login and registration (also validating user detail using regular expression). | 38 | | 17 | Experiment No 10: Programs on Threading using python. | 41 | | 18 | Experiment No 11 Exploring basics of NumPy Methods. |  | | 19 | Experiment No 12: Program to send email and read content of URL. |  | | |



**MGM’s College of Engineering and Technology**

**Vision:**

To become one of the outstanding Engineering Institutes in India by providing a conductive and vibrant environment to achieve excellence in the field of Technology.

**Mission:**

To empower the aspiring professional students to be prudent enough to explore the world of technology and mould them to be proficient to reach the pinnacle of success in the competitive global economy.

**Department of Computer Engineering**

**Vision:**

To motivate and empower the students of Computer Engineering to become globally competent citizens with ethics to serve and lead the society.

To provide a stimulating educational environment for computer engineering graduates to face tomorrow’s challenges and to inculcate social responsibility in them.

**Mission:**

1. To provide excellent academic environment by adopting an innovative teaching techniques through well developed curriculum.
2. To foster a self learning atmosphere for students to provide ethical solutions for societal challenges.
3. To establish Center of Excellence in various domains of Computer Engineering and promote active research and development.
4. To enhance the competency of the faculty in the latest technology through continuous development programs.
5. To foster networking with alumni and industries.

Study and Evaluation Scheme

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Course Code | Course Name | Teaching Scheme | | | Credits Assigned | | | |
| CSL405 | Skill Base Lab Course: Python Programming | Theory | Practical | Tutorial | Theory | Practical | Tutorial | Total |
| 2 | 02 | -- | -- | 2 | -- | 2 |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Course Code | Course Name | Examination Scheme | | |
| CSL405 | Skill Base Lab Course: Python Programming | Term Work | Oral | Total |
| 25 | -- | 25 |

|  |  |
| --- | --- |
| **Term Work** | |
| 1 | Term work should consist of 12 experiments. |
| 2 | Journal must include at least 2 assignments |
| 3 | Mini Project based on the content of the syllabus (Group of 2-3 students) |
| 4 | The final certification and acceptance of term work ensures that satisfactory performance of laboratory work and minimum passing marks in term work. |
| 5 | Total 25 Marks (Journal: 10-marks, Attendance: 05-marks, and Mini Project: 10-marks) |

Lab Objectives & Lab Outcome

**Course Objectives:**

|  |  |
| --- | --- |
|  | Basics of Python programming |
|  | Decision Making, Data structure and Functions in Python |
|  | Object Oriented Programming using Python |
|  | Web framework for developing |

**Course Outcomes:**

|  |  |  |
| --- | --- | --- |
| CO1 | To understand basic concepts in python. | |
| CO2 | To explore contents of files, directories and text processing with python |  |
| CO3 | To develop program for data structure using built in functions in python. |  |
| CO4 | To explore Django web framework for developing python-based web application. |  |
| CO5 | To understand Multi-threading concepts using python. |  |

**Program Specific Outcomes (PSOs)**

The Computer Engineering graduates will be able to

|  |  |
| --- | --- |
| **PSO 1** | Acquire skills to design, analyses and develop algorithms and implement them using high-level programming languages |
| **PSO 2** | Contribute their engineering skills in computing and information engineering domains like network design and administration, database design and knowledge engineering. |
| **PSO 3** | Develop strong skills in systematic planning, developing, testing implementing and providing IT solutions for different domains which helps in the betterment of life. |

**Program Outcomes (POs)**

**Engineering Graduates will be able to:**

**PO1) Engineering knowledge:** Apply the knowledge of mathematics, science, engineering fundamentals, and an engineering specialization to the solution of complex engineering problems.

**PO2) Problem analysis:** Identify, formulate, review research literature, and analyze complex engineering problems reaching substantiated conclusions using first principles of mathematics, natural sciences, and engineering sciences.

**PO3) Design/development of solutions:** Design solutions for complex engineering problems and design system components or processes that meet the specified needs with appropriate consideration for the public health and safety, and the cultural, societal, and environmental considerations.

**PO4) Conduct investigations of complex problems:** Use research-based knowledge and research methods including design of experiments, analysis and interpretation of data, and synthesis of the information to provide valid conclusions.

**PO5) Modern tool usage:** Create, select, and apply appropriate techniques, resources, and modern engineering and IT tools including prediction and modeling to complex engineering activities with an understanding of the limitations.

**PO6) The engineer and society:** Apply reasoning informed by the contextual knowledge to assess societal ,health, safety, legal and cultural issues and the consequent responsibilities relevant to the professional engineering practice.

**PO7) Environment and sustainability:** Understand the impact of the professional engineering solutions in societal and environmental contexts, and demonstrate the knowledge of, and need for sustainable development.

**PO8) Ethics:** Apply ethical principles and commit to professional ethics and responsibilities and norms of the engineering practice.

**PO9) Individual and team work:** Function effectively as an individual, and as a member or leader in diverse teams, and in multidisciplinary settings.

**PO10) Communication:** Communicate effectively on complex engineering activities with the engineering community and with society at large, such as, being able to comprehend and write effective reports and design documentation, make effective presentations, and give and receive clear instructions.

**PO11) Project management and finance:** Demonstrate knowledge and understanding of the engineering and management principles and apply these to one’s own work, as a member and leader in a team, to manage projects and in multidisciplinary environments.

**PO12) Life-long learning:** Recognize the need for, and have the preparation and ability to engage in independent and life-long learning in the broadest context of technological change.

Mapping Lab Outcomes (CO) - Program Outcomes (PO)

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Subject Weight** | **Course Outcomes** | | | **Contribution to Program outcomes** | | | | | | | | | | | |
|  |  | | | **P1** | **P2** | **P3** | **P4** | **P5** | **P6** | **P7** | **P8** | **P9** | **P10** | **P11** | **P12** |
| **PRATICAL** | CO1 | To understand basic concepts in python. | | 1 | 1 | 2 |  |  |  |  | 1 | 2 | 1 | 1 | 1 |
| CO2 | To explore contents of files, directories and text processing with python |  |  | 3 | 1 |  | 1 |  |  |  | 1 | 1 | 2 | 1 |
| CO3 | To develop program for data structure using built in functions in python. |  |  | 1 | 1 |  | 1 |  |  |  | 1 | 1 | 2 | 3 |
| CO4 | To explore Django web framework for developing python-based web application. |  |  | 1 |  |  | 1 |  |  | 1 | 1 | 1 | 2 | 3 |
| CO5 | To understand Multi-threading concepts using python. |  |  | 1 | 1 |  | 1 |  |  | 1 | 1 | 1 | 2 | 1 |

List of Experiments

|  |  |
| --- | --- |
| **Sr. No.** | **Experiments Name** |
| 1 | Experiment 01:Introduction to Python & basic programming concepts in Python.(Downloading and Installation of Python) |
| 2 | Experiment 02: Exploring basics of python like data types (strings, list, array, dictionaries, set, tuples) and control statements |
| 3 | Experiment 03: Creating functions, classes and objects using python. Demonstrate exception handling and inheritance. |
| 4 | Experiment No. 4: Exploring Files and directories  4.a. Python program to append data to existing file and then display the entire file  4.b. Python program to count number of lines, words and characters in a file  4.c. Python program to display file available in current directory |
| 5 | Experiment No 5: Creating GUI with python containing widgets such as labels, textbox, radio, checkboxes and custom dialog boxes |
| 6 | Experiment No 6: Menu driven program for data structure using built in function for link list, stack and queue. |
| 7 | Experiment No 7: Program to demonstrate CRUD (create, read, update and delete) operations on database (SQLite/ MySQL) using python. |
| 8 | Experiment No 8: Creation of simple socket for basic information exchange between server and client. |
| 9 | Experiment No 9: Creating web application using Django web framework to demonstrate functionality of user login and registration (also validating user detail using regular expression). |
| 10 | Experiment No 10: Programs on Threading using python. |
| 11 | Experiment No 11 Exploring basics of NumPy Methods. |
| 12 | Experiment No 12: Program to send email and read content of URL. |

Experiment Plan

|  |  |  |  |
| --- | --- | --- | --- |
| **Module**  **No.** | **Week**  **No.** | **Experiments Name** | **Course**  **Outcome** |
| 1 | W1 | Experiment 01:Introduction to Python & basic programming concepts in Python.(Downloading and Installation of Python) |  |
| 2 | W2 | Experiment 02: Exploring basics of python like data types (strings, list, array, dictionaries, set, tuples) and control statements | CO1 |
| 3 | W3 | Experiment 03: Creating functions, classes and objects using python. Demonstrate exception handling and inheritance. | CO5 |
| 4 | W4 | Experiment No. 4: Exploring Files and directories  4.a. Python program to append data to existing file and then display the entire file  4.b. Python program to count number of lines, words and characters in a file  4.c. Python program to display file available in current directory | CO5 |
| 5 | W5 | Experiment No 5: Creating GUI with python containing widgets such as labels, textbox, radio, checkboxes and custom dialog boxes | CO5 |
| 6 | W6 | Experiment No 6: Menu driven program for data structure using built in function for link list, stack and queue. | CO5 |
| 7 | W7 | Experiment No 7: Program to demonstrate CRUD (create, read, update and delete) operations on database (SQLite/ MySQL) using python. | CO5 |
| 8 | W8 | Experiment No 8: Creation of simple socket for basic information exchange between server and client. | CO5 |
| 9 | W9 | Experiment No 9: Creating web application using Django web framework to demonstrate functionality of user login and registration (also validating user detail using regular expression). | CO3 |
| 10 | W10 | Experiment No 10: Programs on Threading using python. | CO3 |
| 11 | W11 | Experiment No 11 Exploring basics of NumPy Methods. | CO4 |
| 12 | W12 | Experiment No 12: Program to send email and read content of URL. | CO4 |

Experiment No. 1

**Aim:** Introduction to Python & basic programming concepts in Python.(Downloading and Installation of Python)

**Software Required:** Python3.9 / Online Editor

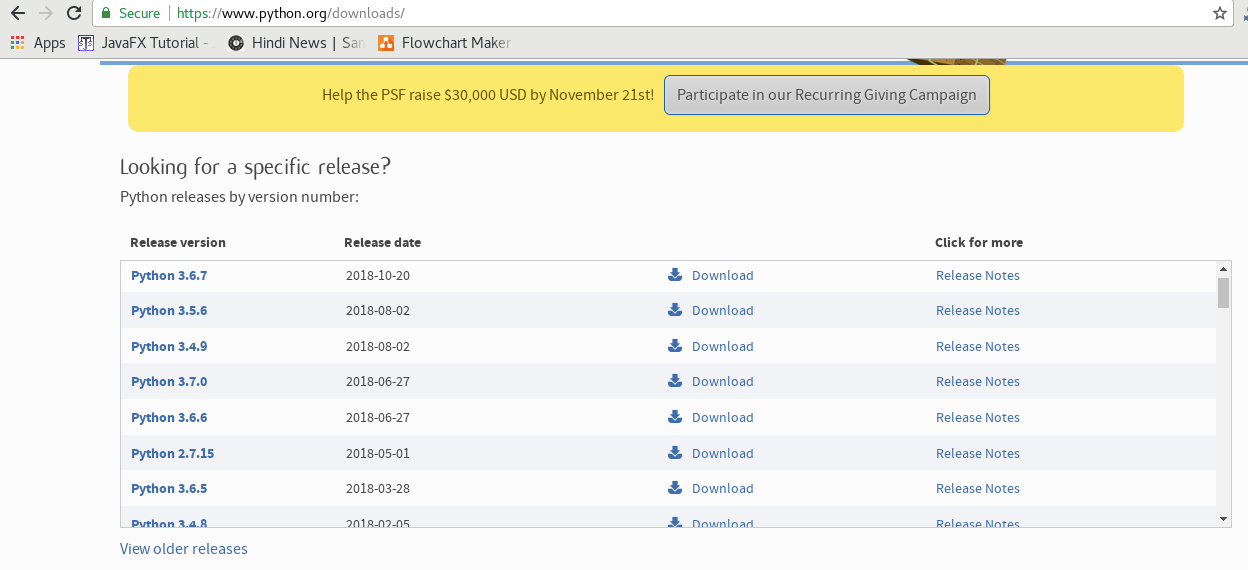
**Theory:**

* **Fundamentals of Python**
* Python is a general-purpose interpreted, interactive, object-oriented, and high-level programming language.
* It was created by Guido van Rossum during 1985- 1990. It’s Named After Monty Python
* Despite all the reptile icons in the Python world, the truth is that Python creator Guido van Rossum named it after the BBC comedy series Monty Python’s Flying Circus. He is a big fan of Monty Python.
* **Python** is a high-level, interpreted, interactive and object-oriented scripting language.
* Python is designed to be highly readable.
* It uses English keywords frequently where as other languages use punctuation, and it has fewer syntactical constructions than other languages.
* **Python** is a MUST for students and working professionals to become a great Software Engineer specially when they are working in Web Development Domain.
* **Characteristics of Python**
* It supports functional and structured programming methods as well as OOP.
* It can be used as a scripting language or can be compiled.
* It provides very high-level dynamic data types and supports dynamic type checking.
* It supports automatic garbage collection.
* It can be easily integrated with C, C++, COM, ActiveX, CORBA, and Java.
* **Python used in the field of**
* Systems Programming
* GUIs
* Internet Scripting
* Component Integration
* Database Programming
* Numeric and Scientific Programming
* Gaming, Images, Serial Ports, XML, Robots, and More
* **Technical strength of Python**
* It’s Object-Oriented
* It’s Free
* It’s Portable
* It’s Powerful
* It’s Mixable
* It’s Easy to Use
* It’s Easy to Learn

**Python Installation**

* **How to Install Python on Windows (Environment Set-up)**

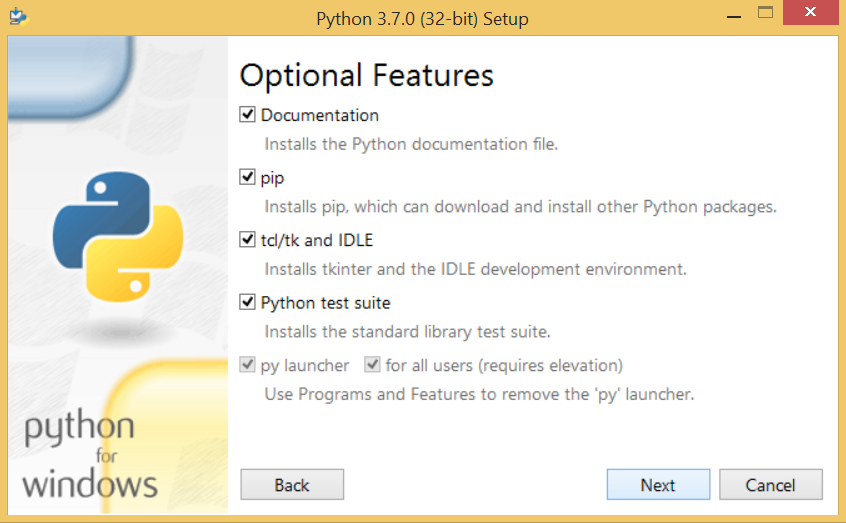
**1. Visit the link*<https://www.python.org/downloads/>* to download the latest release of Python.**

****

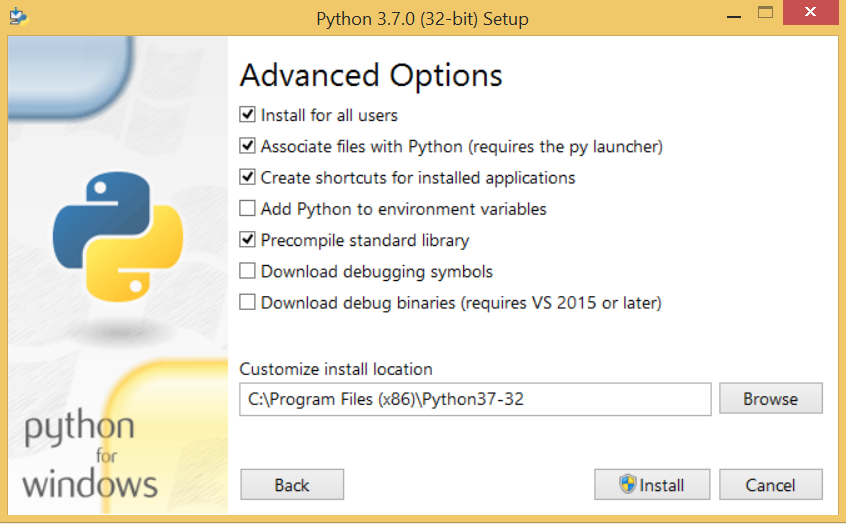
**2. Double-click the executable file, which is downloaded; the following window will open. Select Customize installation and proceed.**

****

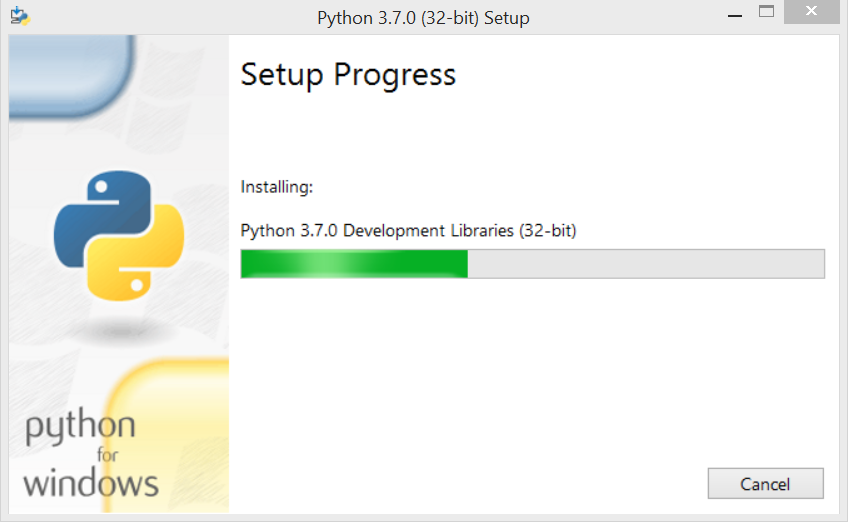
**3. The following window shows all the optional features. All the features need to be installed and are checked by default; we need to click next to continue.**

****

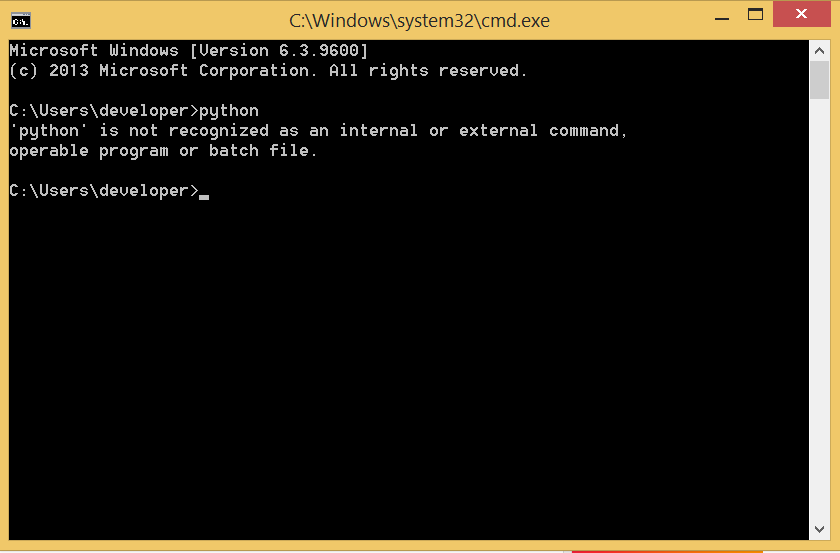
**4. The following window shows a list of advanced options. Check all the options which you want to install and click next. Here, we must notice that the first check-box (install for all users) must be checked.**

****

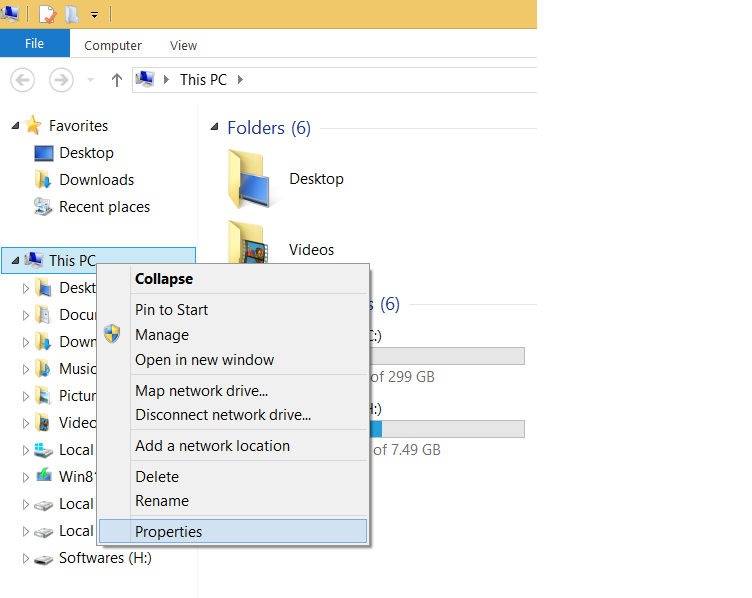
**5. Now, we are ready to install python-3.6.7. Let's install it.**

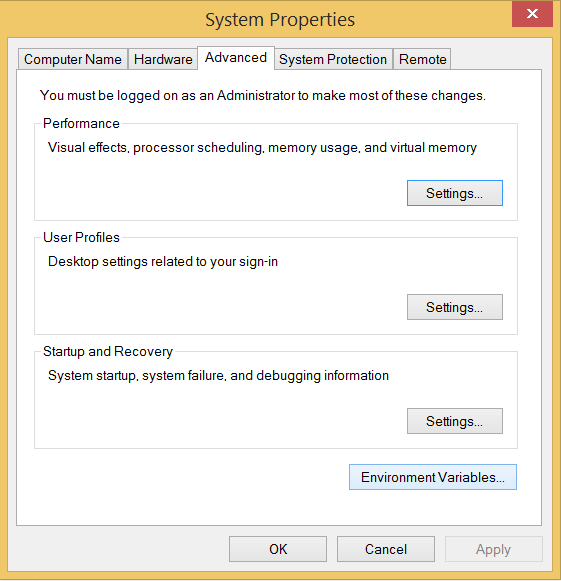
****

**6. Now, try to run python on the command prompt. ( press window button + R) Type the command python in case of python2 or python3 in case of python3. It will show an error as given in the below image. It is because we haven't set the path.**

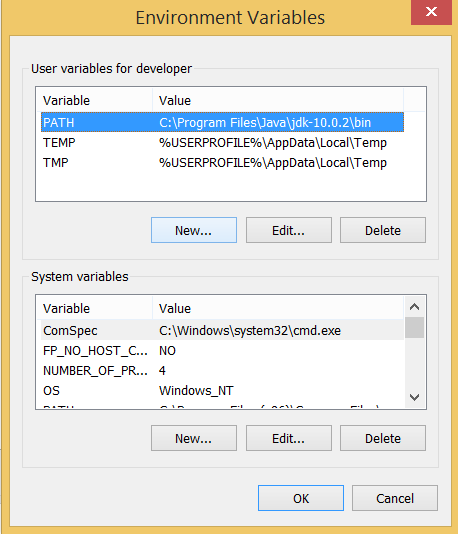
****

**7. To set the path of python, we need to the right click on "my computer / This PC" and go to Properties → Advanced → Environment Variables.**

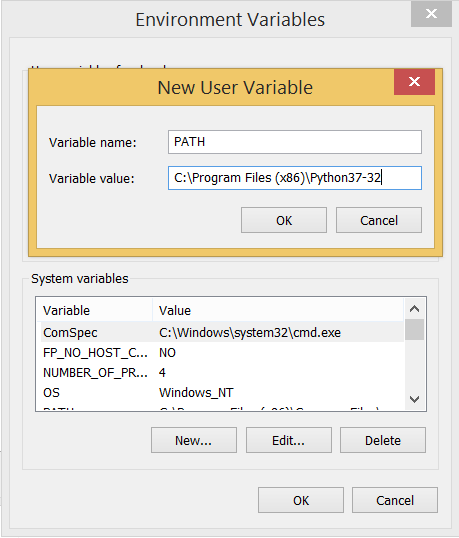
****

****

**8. Add the new path variable in the user variable section.**

****

**9. Type PATH as the variable name and set the path to the installation directory of the python shown in the below image.**

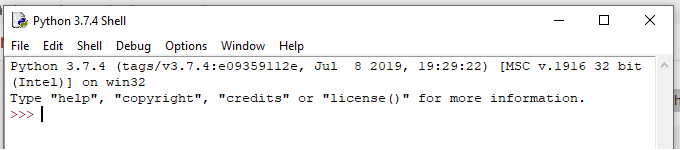
****

**Now, the path is set; we are ready to run python on our local system. Restart CMD, and type python again. It will open the python interpreter shell where we can execute the python statements.**

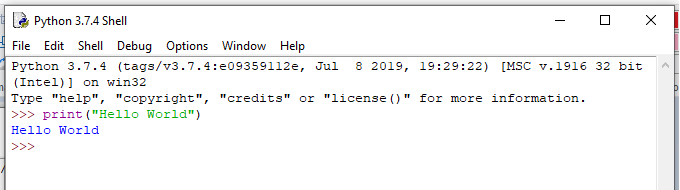
* **Executing Python program**
* Python provides us the two ways to run a program:

1. Using Interactive interpreter prompt

* Python provides us the feature to execute the Python statement one by one at the interactive prompt. It is preferable in the case where we are concerned about the output of each line of our Python program.
* To open the interactive mode, open the terminal (or command prompt) and type python (python3 in case if you have Python2 and Python3 both installed on your system).
* It will open the following prompt where we can execute the Python statement and check their impact on the console.

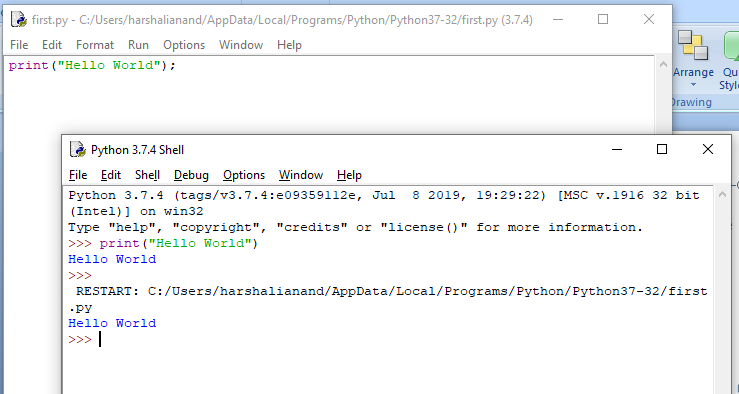


Here, we get the message **"Hello World !"** printed on the console.



2. Using a script file

* Interpreter prompt is good to run the individual statements of the code. However, we cannot write the code every-time on the terminal.
* We need to write our code into a file which can be executed later. For this purpose, open an editor like notepad or using new file in python editor, create a file named first.py (Python used .py extension) and write the following code in it.
* **print** ("hello world");
* here, we have used print() function to print the message on the console.
* To run this file named as first.py, we need to run the following command on the terminal.



* Execute the following task using the basic knowledge of Python Language

1. Python Program to Print Hello MGMCET
2. Python Program to Check If number is Even or Odd
3. [Python program to check whether the input character is an alphabet](https://beginnersbook.com/2018/01/python-program-check-alphabet/)
4. Python Program to Check if a Number is Positive Negative or Zero
5. Python Program to Check Whether a String is Palindrome or Not

**Conclusion:**

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**Result:**

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**Industrial Application:**

* Systems Programming
* GUIs
* Internet Scripting
* Component Integration
* Database Programming
* Numeric and Scientific Programming
* Gaming, Images, Serial Ports, XML, Robots, and More

**Questions:**

1. How to comment a statement in Python?

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

2. Enlist the applications of Python.

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

3. What is IDE?

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

4. State the shortcut to execute the Python code while using script file.

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

5. Write control statements in Python.

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

6. What is variable?

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

7. Enlist editors or framework where we can write Python programs and also can create Python project.

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

8. Shortcut to get help in Python.

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

9. What is the extension of Python file?

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

10. Define constant

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Experiment No. 2

**Aim:** Exploring basics of python like data types (strings, list, array, dictionaries, set, tuples) and control statements

**Software Required:** Python3.9 / Online Editor

**Theory:**

**(i) Numbers**

a = 5

print(a, "is of type", type(a))

**Output:**

a = 2.0

print(a, "is of type", type(a))

**Output:**

a = 1+2j

print(a, "is complex number?", isinstance(1+2j,complex))

**Output:**

**(ii) Python List**

a = [5,10,15,20,25,30,35,40]

# a[2] = 15

print("a[2] = ", a[2])

**Output:**

# a[0:3] = [5, 10, 15]

print("a[0:3] = ", a[0:3])

**Output:**

# a[5:] = [30, 35, 40]

print("a[5:] = ", a[5:])

**Output:**

**(iii) Python Tuple**

t = (5,'program', 1+3j)

# t[1] = 'program'

print("t[1] = ", t[1])

**Output:**

# t[0:3] = (5, 'program', (1+3j))

print("t[0:3] = ", t[0:3])

**Output:**

# Generates error

# Tuples are immutable

t[0] = 10

**(iv) Python Strings**

s = "This is a string"

print(s)

**Output:**

s = '''A multiline

string'''

print(s)

**Output:**

s = 'Hello world!'

# s[4] = 'o'

print("s[4] = ", s[4])

**Output:**

# s[6:11] = 'world'

print("s[6:11] = ", s[6:11])

**Output:**

# Generates error

# Strings are immutable in Python

s[5] ='d'

**(v) Python Set**

a = {5,2,3,1,4}

# printing set variable

print("a = ", a)

**Output:**

# data type of variable a

print(type(a))

**Output:**

**(vi) Python Dictionary**

d = {1:'value','key':2}

print(type(d))

**Output:**

print("d[1] = ", d[1]);

**Output:**

print("d['key'] = ", d['key']);

**Output:**

# Generates error

print("d[2] = ", d[2]);

**Output:**

**(vii) Conversion between data types**

We can convert between different data types by using different type conversion functions like int(), float(), str(), etc.

>>> float(5)

**Output:**

Conversion from float to int will truncate the value (make it closer to zero).

>>> int(10.6)

**Output:**

>>> int(-10.6)

**Output:**

Conversion to and from string must contain compatible values.

>>> float('2.5')

**Output:**

>>> str(25)

**Output:**

>>> int('1p')

**Output:**

>>> set([1,2,3])

**Output:**

>>> tuple({5,6,7})

**Output:**

>>> list('hello')

**Output:**

To convert to dictionary, each element must be a pair:

>>> dict([[1,2],[3,4]])

**Output:**

>>> dict([(3,26),(4,44)])

**Output:**

**Conditional Statements in Python**

**Syntax**

if condition1:

  statements

elif condition2:

statements

else:

statements

   Consider the example below:

X = 10

Y = 12

if X < Y: print('X is less than Y') elif X > Y:

    print('X is greater than Y')

else:

    print('X and Y are equal')

**Output:**

**While Loop**

Here, first the condition is checked and if it’s true, control will move inside the loop and execute the statements inside the loop until the condition becomes false. We use this loop when we are not sure how many times we need to execute a group of statements or you can say that when we are unsure about the number of iterations.

Consider the example:

Syntax and Usage:

count = 0

while (count < 10):

   print ( count )

   count = count + 1

print ("Good bye!")

**Output** =

**For Loop**

Like the While loop, the For loop also allows a code block to be repeated certain number of times. The difference is, in For loop we know the amount of iterations required unlike While loop, where iterations depends on the condition. You will get a better idea about the difference between the two by looking at the syntax:

Syntax:

for variable in Sequence:

    statements

Notice here, we have specified the range, that means we know the number of times the code block will be executed.

Consider the example:

fruits = ['Banana', 'Apple',  'Grapes']

for index in range(len(fruits)):

    print (fruits[index])

**Output:**

**Nested Loops**

It basically means a loop inside a loop. It can be a For loop inside a While loop and vice-versa. Even a For loop can be inside a For loop or a While loop inside a While loop.

Consider the example:

count = 1

for i in range(10):

    print (str(i) \* i)

    for j in range(0, i):

        count = count +1

**Output**

**Conclusion:**

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**Result:**

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Experiment No. 3

**Aim:** Creating functions, classes and objects using python. Demonstrate exception handling and inheritance.

**Software Required:** Python3.9 / Online Editor

**Theory:**

**Python Function**

Functions are the most important aspect of an application. A function can be defined as the organized block of reusable code, which can be called whenever required.

Python allows us to divide a large program into the basic building blocks known as a function. The function contains the set of programming statements enclosed by {}. A function can be called multiple times to provide reusability and modularity to the Python program.

The Function helps to programmer to break the program into the smaller part. It organizes the code very effectively and avoids the repetition of the code. As the program grows, function makes the program more organized.

Python provide us various inbuilt functions like **range()** or **print()**. Although, the user can create its functions, which can be called user-defined functions.

There are mainly two types of functions.

* **User-define functions** - The user-defined functions are those define by the **user** to perform the specific task.
* **Built-in functions** - The built-in functions are those functions that are **pre-defined** in Python.

There are the following advantages of Python functions.

* Using functions, we can avoid rewriting the same logic/code again and again in a program.
* We can call Python functions multiple times in a program and anywhere in a program.
* We can track a large Python program easily when it is divided into multiple functions.
* Reusability is the main achievement of Python functions.
* However, Function calling is always overhead in a Python program.

### Creating a Function

Python provides the **def** keyword to define the function. The syntax of the define function is given below.

**Syntax:**

**def** my\_function(parameters):

      function\_block

**return** expression

Let's understand the syntax of functions definition.

* The **def** keyword, along with the function name is used to define the function.
* The identifier rule must follow the function name.
* A function accepts the parameter (argument), and they can be optional.
* The function block is started with the colon (:), and block statements must be at the same indentation.
* The **return** statement is used to return the value. A function can have only one **return**

### Function Calling

In Python, after the function is created, we can call it from another function. A function must be defined before the function call; otherwise, the Python interpreter gives an error. To call the function, use the function name followed by the parentheses.

Consider the following example of a simple example that prints the message "Hello World".

#function definition

**def** hello\_world():

**print**("hello world")

# function calling

hello\_world()

**Output:**

hello world

## Arguments in function

The arguments are types of information which can be passed into the function. The arguments are specified in the parentheses. We can pass any number of arguments, but they must be separate them with a comma.

## Types of arguments

There may be several types of arguments which can be passed at the time of function call.

1. Required arguments
2. Keyword arguments
3. Default arguments
4. Variable-length arguments

### 1. Required Arguments

Till now, we have learned about function calling in Python. However, we can provide the arguments at the time of the function call. As far as the required arguments are concerned, these are the arguments which are required to be passed at the time of function calling with the exact match of their positions in the function call and function definition. If either of the arguments is not provided in the function call, or the position of the arguments is changed, the Python interpreter will show the error.

## 2. Default Arguments

Python allows us to initialize the arguments at the function definition. If the value of any of the arguments is not provided at the time of function call, then that argument can be initialized with the value given in the definition even if the argument is not specified at the function call.

### 3. Variable-length Arguments (\*args)

In large projects, sometimes we may not know the number of arguments to be passed in advance. In such cases, Python provides us the flexibility to offer the comma-separated values which are internally treated as tuples at the function call. By using the variable-length arguments, we can pass any number of arguments.

However, at the function definition, we define the variable-length argument using the **\*args** (star) as \*<variable - name >.

### 4. Keyword arguments (\*\*kwargs)

Python allows us to call the function with the keyword arguments. This kind of function call will enable us to pass the arguments in the random order.

The name of the arguments is treated as the keywords and matched in the function calling and definition. If the same match is found, the values of the arguments are copied in the function definition.

**Questions:**

1. What are the various Python language editor available for writing Python code?

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**2. Enlist advantages of Python language.

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ **\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

3. Enlist various applications of Python language.

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

4. Can we execute scilab code in Matlab? Explain.

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

5. What is IDE?

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**6. Can we execute numerical methods programmatically in Python? Explain.

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

7. Write in short about Python language.

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**8. Enlist various real life applications of Python language.

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

9. What is PyCharm? Write in short about PyCharm?

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

10. Write in details about Python list.

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

References:

1. <https://www.w3schools.com/python/>
2. <https://www.journaldev.com/15906/python-socket-programming-server-client>