DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING

VISION & MISSION OF THE INSTITUTE

Vision of the Institute

To be a globally recognized institution that fosters innovation, excellence, and leadership in education, research, and technology development, empowering students to create sustainable solutions for the advancement of society.

Mission of the Institute

- To foster a transformative learning environment that empowers students to excel in engineering, innovation, and leadership.
- To produce skilled, ethical, and socially responsible engineers who contribute to sustainable technological advancements and address global challenges.
- To shape future leaders through cutting-edge research, industry collaboration, and community engagement.

DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING

DEPARTMENT VISION & MISSION

Vision:

To empower the students to be technologically adept, innovative, self-motivated and responsible global citizen possessing human values and contribute significantly towards high quality technical education by harmonizing innovation with sustainability.

Mission:

- To offer high-quality education in the computing fields by providing an environment where the knowledge is gained and applied to participate in research, for both students and faculty.
- To develop the problem solving skills in the students to be ready to deal with cutting edge technologies of the industry.
- To make the students and faculty excel in their professional fields by inculcating the communication skills, leadership skills, team building skills with the organization of various co-curricular and extra-curricular programmers.
- To provide the students with theoretical and applied knowledge, and adopt an education approach that promotes lifelong learning and ethical growth.

MARRI LAXMAN REDDY
INSTITUTE OF TECHNOLOGY AND MANAGEMENT
(AN AUTONOMOUS INSTITUTION)
(Approved by AICTE, New Delhi & Affiliated to JNTUH, Hyderabad)
Accredited by NAAC with 'A' Grade & Recognized Under Section 2(f) & 12(B) of the UGC act,1956

DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING

CERTIFICATE

This is to certify that this manual is a **bonafide record of practical work** carried out in **Python programming Lab** for the B.Tech Computer Science and Engineering (CSE) III Semester Programme during the academic year **2026–2027**.

This manual has been prepared by the **Department of Computer Science and Engineering (CSE)** with our own efforts and to the best of our knowledge.

Signature of Lab Faculty Signature of HOD

DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING

PREFACE

This lab manual entitled **"Python programming Lab"** is intended for the use of II B.Tech I Semester Computer Science and Engineering (CSE) students of **Marri Laxman Reddy Institute of Technology and Management, Dundigal, Hyderabad**.

The main objective of the Software Engineering Lab is to provide hands-on experience with intelligent computing techniques such as neural networks, fuzzy logic, and genetic algorithms. Students learn to solve complex, real-world problems that do not have exact mathematical models.

It enhances skills in Artificial Intelligence and Machine Learning through practical implementation and experimentation. The lab prepares learners for advanced research and industry applications in adaptive and intelligent systems.

DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING

GENERAL INSTRUCTIONS

1. Students are instructed to come to the Python programming laboratory on time. Latecomers are not entertained in the lab.
2. Students should be punctual. Experiments will not be repeated for latecomers.
3. Students must come prepared with the experiments to be performed.
4. Students must display their identity cards before entering the lab.
5. Mobile phones are strictly not allowed inside the lab.
6. Any damage or loss of system components (keyboard, mouse, etc.) is the responsibility of the student, and a penalty will be imposed.
7. Students must update their records and observation books session-wise. The observation book should be signed by the faculty before leaving the lab.
8. Lab records must be submitted in the next lab session for correction.
9. Students should not move around unnecessarily during the lab session.
10. In case of emergency, students must obtain written permission from the concerned faculty.
11. Faculty members may suspend any student from the lab session on disciplinary grounds.
12. Students must not copy outputs from others and should write their own results.

DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING

SAFETY MEASURES

To ensure the safe and efficient use of the Computer Science and Engineering laboratory, all students must strictly follow the safety guidelines given below:

1. General Conduct

- Maintain silence and discipline during lab sessions.
- Do not bring food, drinks, or chewing gum into the lab.
- Use lab resources responsibly and follow all instructions given by the instructor or lab assistant.

2. Electrical Safety

- Do not touch electrical switches, sockets, or plugs with wet hands.
- Avoid overloading power sockets with unauthorized devices.
- Immediately report any loose connections, sparks, or unusual noises.

3. Computer and Equipment Handling

- Handle computers, keyboards, mouse, and peripherals with care.
- Do not open or tamper with hardware components.
- Use only the assigned system; do not switch without permission.

4. Software and Data Safety

- Use only authorized software installed by the lab administrator.
- Do not install or modify any software without approval.
- Save your work frequently and maintain backups.

5. Cyber Security and Network Usage

- Keep login credentials confidential.
- Do not access restricted websites or servers.
- Avoid activities such as hacking, gaming, or using pirated content.

6. Emergency Preparedness

- Be aware of emergency exits, fire extinguishers, and first aid kits.
- In case of emergencies, remain calm and inform the instructor immediately.
- Follow evacuation procedures properly.

7. Post-Lab Procedures

- Log out and shut down the system properly.
- Keep the workstation clean and organized.
- Return borrowed materials to their place.

8. Hygiene and Cleanliness

- Wash or sanitize hands before and after using systems.
- Do not place unnecessary items on the workstation.
- Report spills or cleanliness issues to lab staff.

DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING

PROGRAM OUTCOMES (PO'S)

PO NO	NBA Statement / Vital Features		
	Graduate Attributes	**Program Outcomes**	**No. of key competencies**
PO1	Engineering knowledge	Apply the knowledge of mathematics, science, engineering fundamentals, and an engineering specialization to the solution of complex engineering problems.	3
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.	10
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.	10
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.	11
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.	1
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	5

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.	3
PO8	Ethics	Apply ethical principles and commit to professional ethics and responsibilities and norms of the engineering practice.	3
PO9	Individual and team work	Function effectively as an individual, and as a member or leader in diverse teams, and in multidisciplinary settings.	12
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.	5
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.	12
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	8

DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING

PROGRAM SPECIFIC OUTCOMES(PSO'S)

PO NO	NBA Statement / Vital Features	
	Program Specific Outcomes	No. of key competencies
PSO1	Applications of Computing: Ability to use knowledge in various domains to provide solution to new ideas and innovations.	2
PSO2	Programming Skills: Identify required data structures, design suitable algorithms, develop and maintain software for real world problems.	3
PSO3	Make use of computational and experimental knowledge for creating innovative career paths, to be an entrepreneur and desire for higher studies.	2

DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING

PROGRAM EDUCATIONAL OBJECTIVES (PEOs)

PO NO	NBA Statement / Vital Features
	Program Specific Outcomes
PEO1	**Professional Competence**
	Graduates will possess strong theoretical and practical knowledge in Artificial Intelligence and Machine Learning, enabling them to solve complex real-world problems, pursue higher education, or excel in professional careers.
PEO2	**Innovation and Research Orientation**
	Graduates will engage in innovative practices, cutting-edge research, and contribute to the advancement of AI and ML technologies through collaboration with industry and academia.
PEO3	**Leadership and Lifelong Learning**
	Graduates will exhibit leadership qualities, effective communication, and teamwork skills, and will continuously upgrade their knowledge to adapt to evolving technological landscapes.
PE04	**Entrepreneurial and Community Engagement**
	Graduates will leverage entrepreneurial skills and a sense of civic responsibility to create AI-driven solutions that benefit local and global communities.

MARRI LAXMAN REDDY
INSTITUTE OF TECHNOLOGY AND MANAGEMENT
(AN AUTONOMOUS INSTITUTION)
(Approved by AICTE, New Delhi & Affiliated to JNTUH, Hyderabad)
Accredited by NAAC with 'A' Grade & Recognized Under Section 2(f) & 12(B) of the UGC act,1956

DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING

25X0579: PYTHON PROGRAMMING LAB

B.Tech II Year I Semester
L/T/P/C: 0/0/2/1

COURSE OBJECTIVES

To install and run the Python interpreter

- To learn control structures.
- To Understand Lists, Dictionaries in python
- To Handle Strings and Files in Python

COURSE OUTCOMES

Apply Python language fundamentals, interpreter features, control structures, functions, recursion, and modules for solving computational problems.

- Develop Python programs using lists, tuples, arrays, dictionaries, and strings for data manipulation and validation tasks.
- Implement file handling, exception handling, and text processing techniques for data analysis and information retrieval.
- Design object-oriented Python programs using classes, attributes, and methods including graphical object representation and GUI components.
- Utilize Python libraries and packages such as NumPy, SciPy, Plotting tools, and logic gate simulations for scientific computing and visualization.

LIST OF EXPERIMENTS

1. I. Use a web browser to go to the Python website http://python.org. This page contains information about Python and links to Python-related pages, and it gives you the ability to search the Python documentation. II. Start the Python interpreter and type help() to start the online help utility.
2. Start a Python interpreter and use it as a Calculator.
3. Write a program to calculate compound interest when principal, rate and number of periods are given.
4. Read the name, address, email and phone number of a person through the keyboard and print the details.

5. Print the below triangle using for loop. 5 4 4 3 3 3 2 2 2 2 1 1 1 1 1
6. Write a program to check whether the given input is digit or lowercase character or uppercase character or a special character (use 'if-else-if' ladder)
7. Python program to print all prime numbers in a given interval (use break)
8. Write a program to convert a list and tuple into arrays.
9. Write a program to find common values between two arrays.
10. Write a function called palindrome that takes a string argument and returns True if it is a palindrome and False otherwise. Remember that you can use the built-in function len to check the length of a string.
11. Write a function called is_sorted that takes a list as a parameter and returns True if the list is sorted in ascending order and False otherwise.
12. Write a function called has_duplicates that takes a list and returns True if there is any element that appears more than once. It should not modify the original list.
13. Write a function called remove_duplicates that takes a list and returns a new list with only the unique elements from the original. Hint: they don't have to be in the same order.
14. The wordlist I provided, words.txt, doesn't contain single letter words. So you might want to add "I", "a", and the empty string.
15. Write a python code to read dictionary values from the user. Construct a function to invert its content. i.e., keys should be values and values should be keys.
16. Add a comma between the characters. If the given word is 'Apple', it should become 'A,p,p,l,e'
17. Remove the given word in all the places in a string?
18. Write a function that takes a sentence as an input parameter and replaces the first letter of every word with the corresponding upper case letter and the rest of the letters in the word by corresponding letters in lower case without using a built-in function?
19. Writes a recursive function that generates all binary strings of n-bit length
20. Write a python program that defines a matrix and prints
21. Write a python program to perform multiplication of two square matrices
22. How do you make a module? Give an example of construction of a module using different geometrical shapes and operations on them as its functions.
23. Use the structure of exception handling all general-purpose exceptions.
24. Write a function called draw_rectangle that takes a Canvas and a Rectangle as arguments and draws a representation of the Rectangle on the Canvas.
25. Add an attribute named color to your Rectangle objects and modify draw_rectangle so that it uses the color attribute as the fill color.
26. Write a function called draw_point that takes a Canvas and a Point as arguments and draws a representation of the Point on the Canvas.
27. Define a new class called Circle with appropriate attributes and instantiate a few Circle objects. Write a function called draw_circle that draws circles on the canvas.
28. Write a python code to read a phone number and email-id from the user and validate it for correctness.
29. Write a Python code to merge two given file contents into a third file.
30. Write a Python code to open a given file and construct a function to check for given words present in it and display on found.
31. Write a Python code to Read text from a textfile, find the word with most number of occurrences

32. Write a function that reads a file file1and displays the number of words, number of vowels, blank spaces, lower case letters and uppercase letters.
33. Import numpy, Plotpy and Scipy and explore their functionalities.
34. Install NumPypackage with pip and explore it.
35. Write a program to implement Digital Logic Gates – AND, OR, NOT, EX-OR
36. Write a GUI program to create a window wizard having two text labels, two text fields and two buttons as Submit and Reset

PYTHON LAB PROGRAMS

1. I. Use a web browser to go to the Python website http://python.org. This page contains information about Python and links to Python-related pages, and it gives you the ability to search the Python documentation.
II. Start the Python interpreter and type help() to start the online help utility.

Solution :

I. Use a web browser to go to the Python website http://python.org. This page contains information about Python and links to Python-related pages, and it gives you the ability to search the Python documentation.

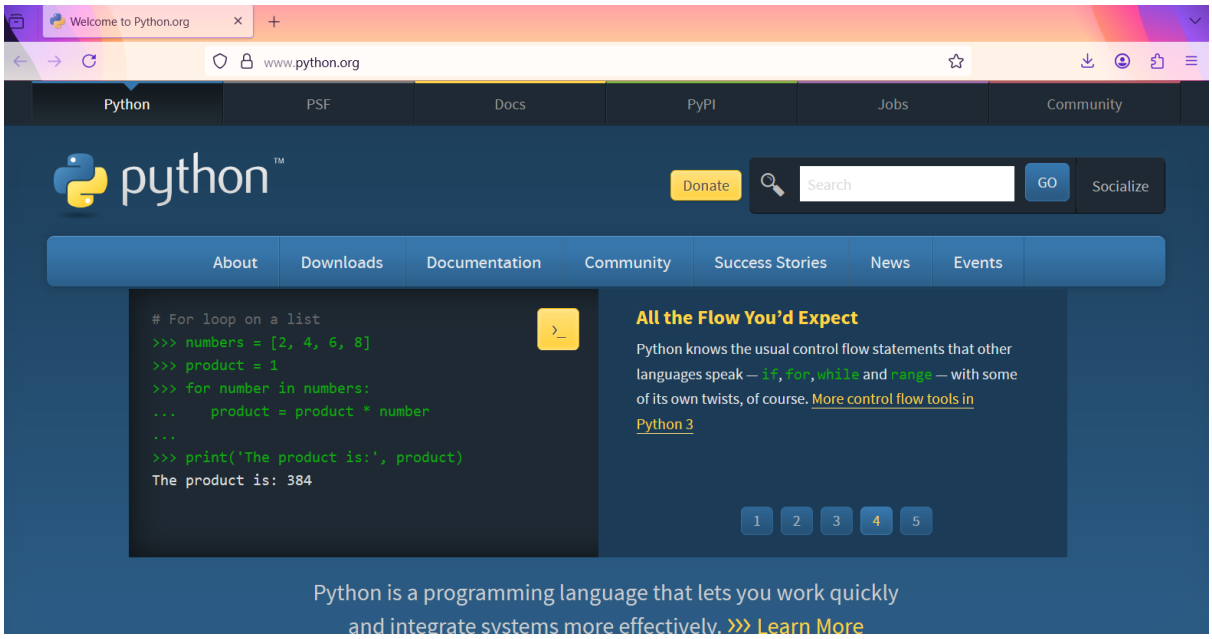
In this tutorial we are going to learn about installation of Python3 on **✍ Windows** and **✍Linux (Ubuntu)**

✍ Python installation procedure in Windows :

To install Python in windows, we have to download Python software pack from official website of Python Software Foundation.

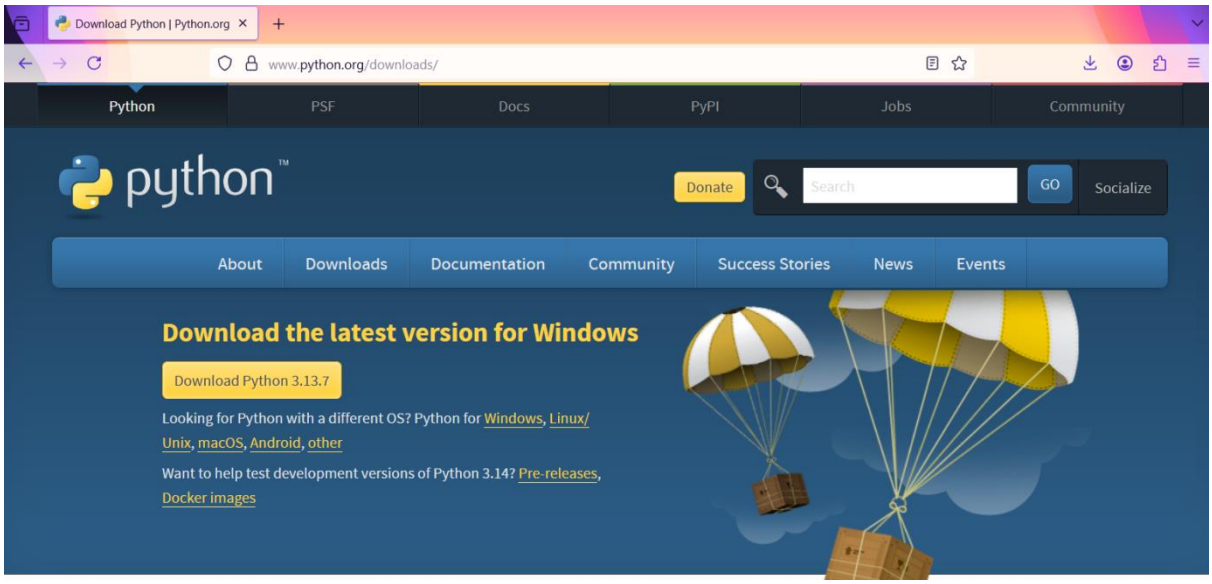
1. **Go to https://www.python.org/**

   The following page will displayed on your web browser.



2. **Go to Downloads**

   The following page will displayed on your web browser.



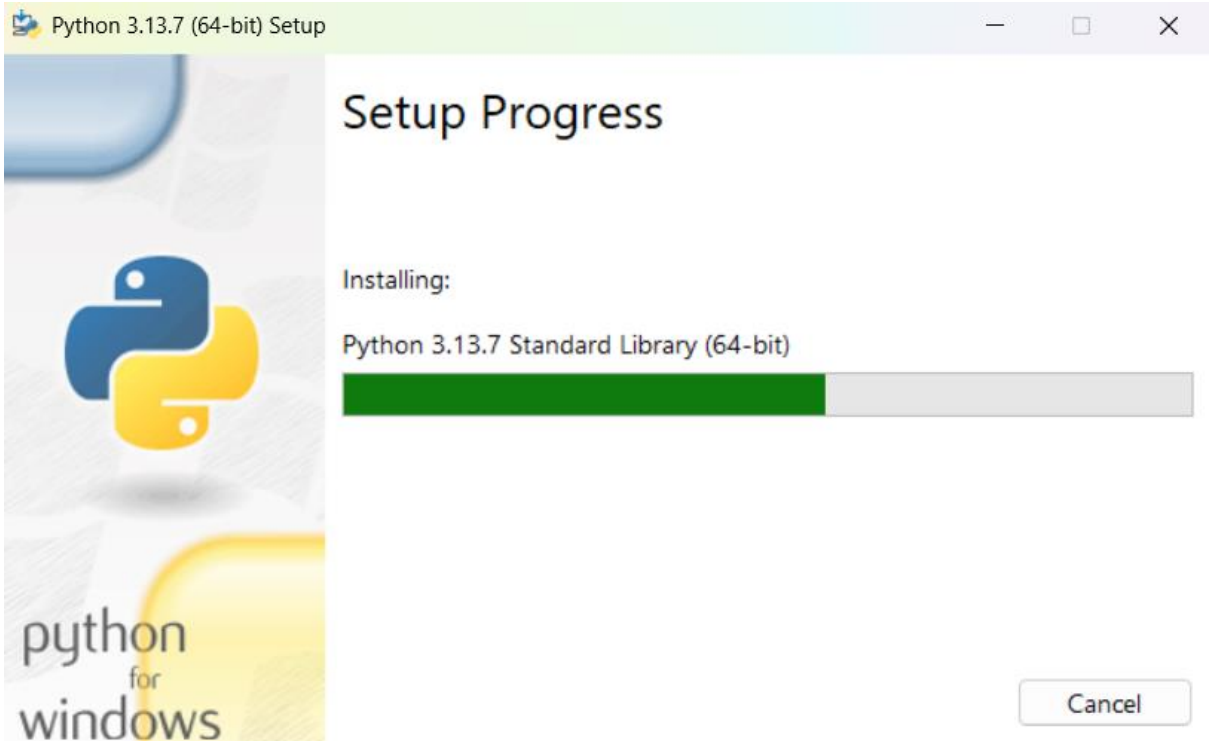
3. **Download the latest version of python(now latest version:python 3.13.7) for windows**

4. **After the successful completion of download,we need to run *python-3.13.7-amd64.exe* file**

Then the Python setup window will be displayed as shown in below.

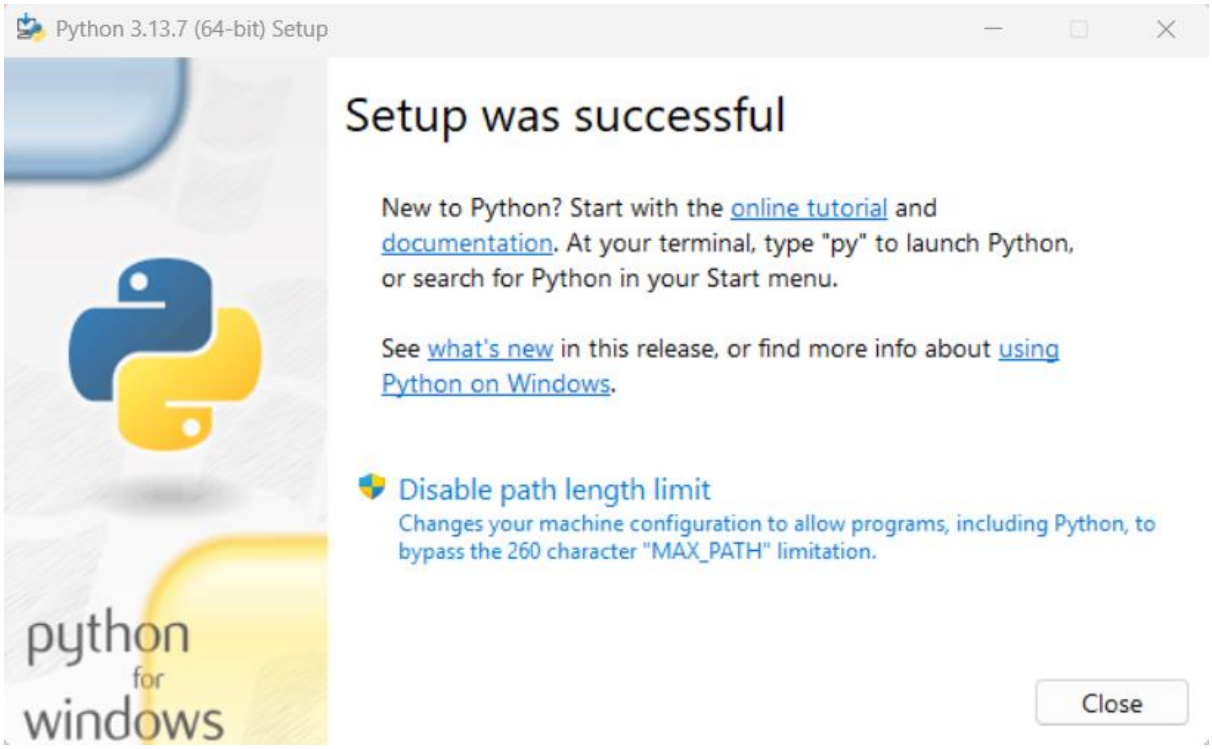
5. **Click on *Install Now***

   First we need to check ***Add Python 3.13.7 to PATH***, And then click on ***Install Now***, the Python setup progress window will be displayed as shown in below.



6. **Setup was Successful**

The Python setup will take 2 to 3 minutes of time.After successful installation the following window will be displayed

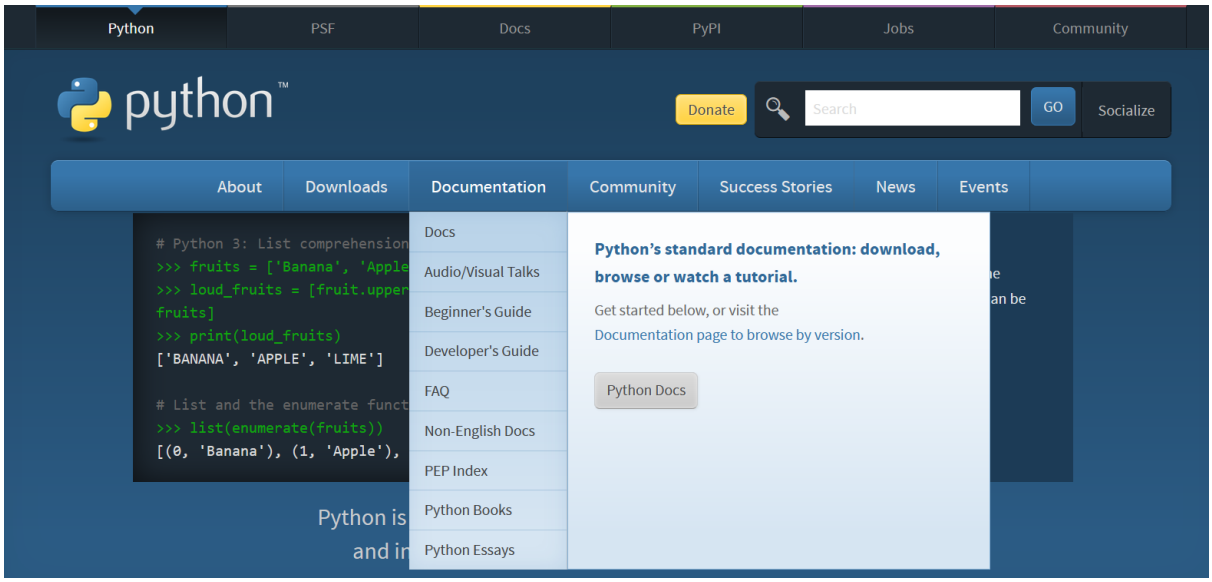


7. **Click on *Close* button**

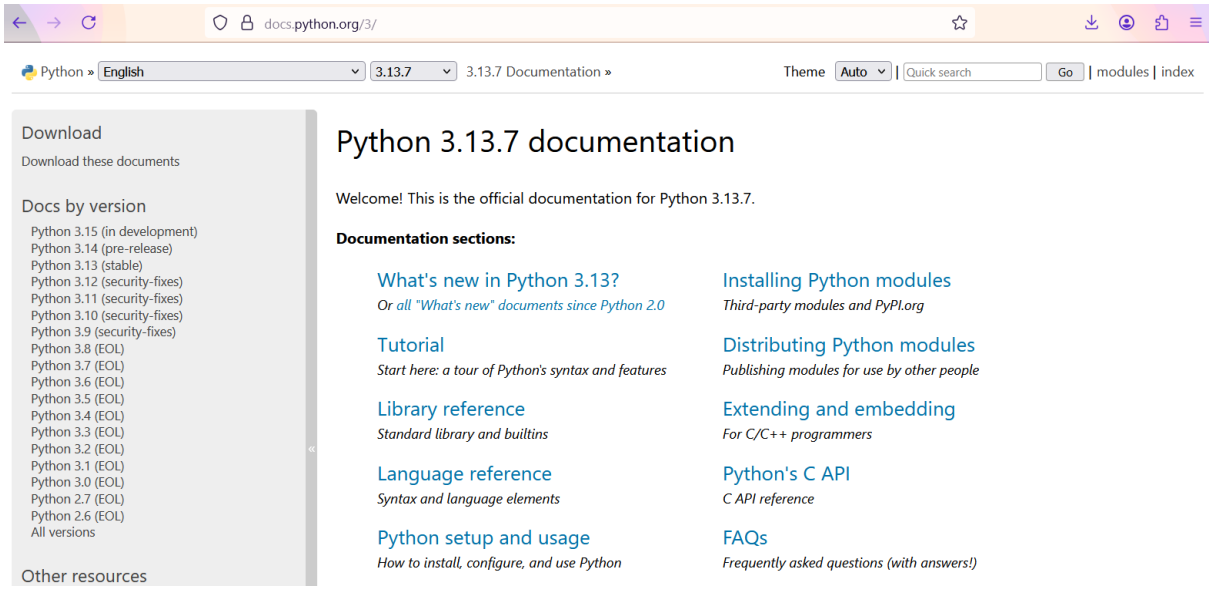
   That's it, Python Setup was Successful

Python Documentaion:

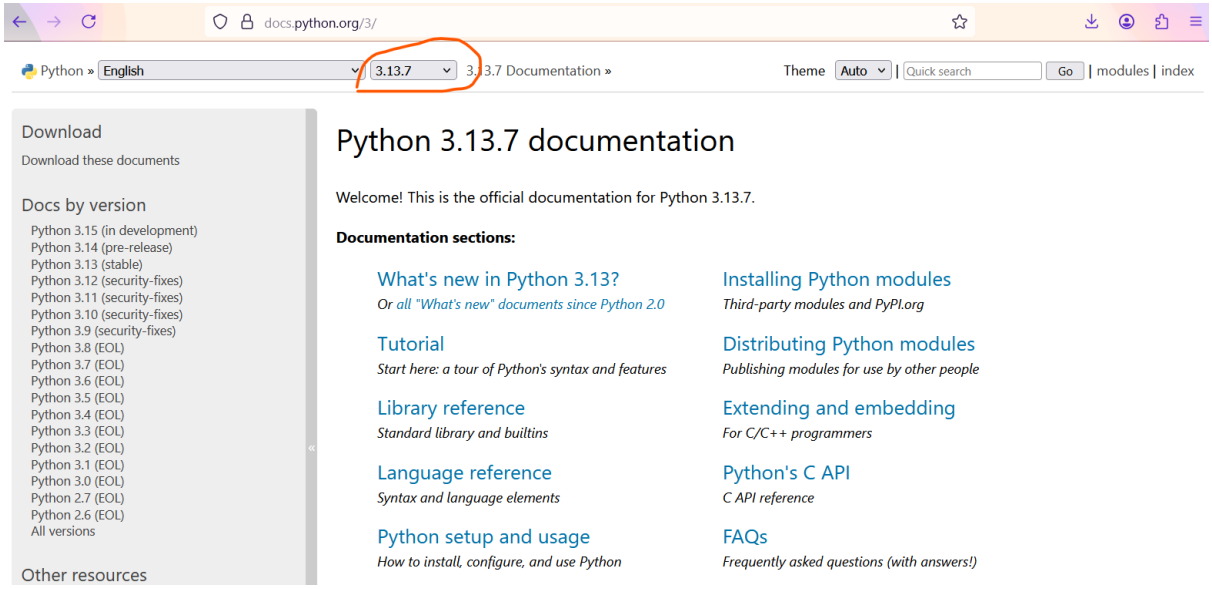
1. **Go to python official website www.python.org**
   click on documentation page



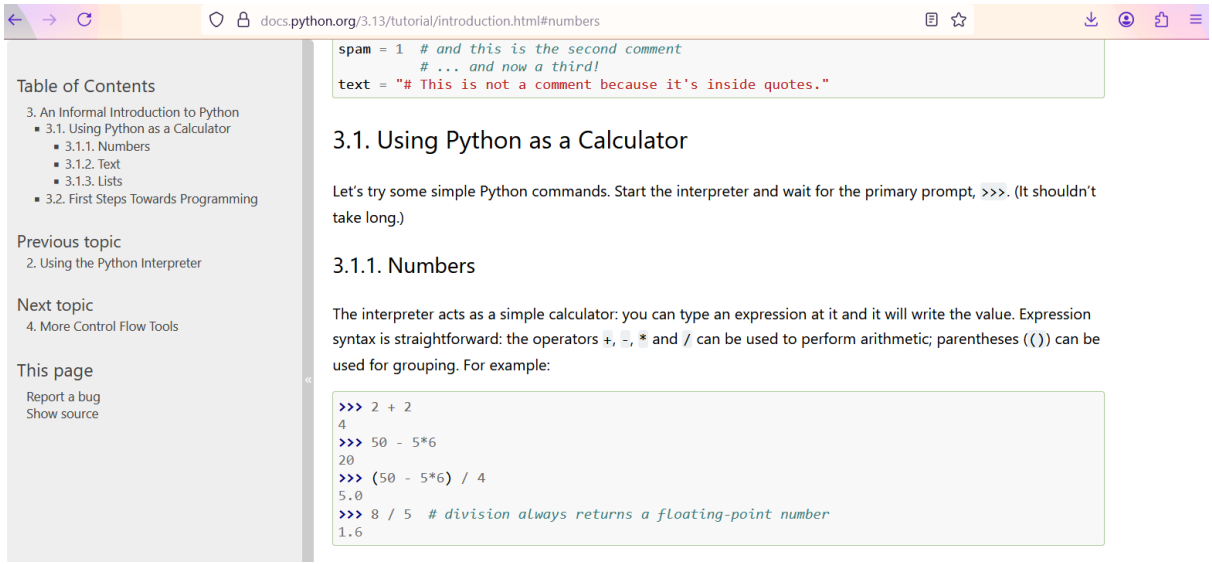
2. Click on **Python Docs**, then the following window will be displayed.



3. **Select version**



4. **Select topic**



II. Start the Python interpreter and type help() to start the online help utility.

To Start the Python interpreter in your Linux (Ubuntu) system, open terminal by pressing ***Ctrl*** + ***ALT*** + ***T*** and then type following command press enter.

$ python3

And then type following command press enter.

>>> help()

The following is the sample output screenshot

```
(base) nnrg@nnrg-OptiPlex-Tower-7010:~/csec$ python3
Python 3.12.7 | packaged by Anaconda, Inc. | (main, Oct  4 2024, 13:27:36) [GCC 11.2.0] on linux
Type "help", "copyright", "credits" or "license" for more information.
>>> help()
Welcome to Python 3.12's help utility! If this is your first time using
Python, you should definitely check out the tutorial at
https://docs.python.org/3.12/tutorial/.

Enter the name of any module, keyword, or topic to get help on writing
Python programs and using Python modules.  To get a list of available
modules, keywords, symbols, or topics, enter "modules", "keywords",
"symbols", or "topics".

Each module also comes with a one-line summary of what it does; to list
the modules whose name or summary contain a given string such as "spam",
enter "modules spam".

To quit this help utility and return to the interpreter,
enter "q" or "quit".

help> keywords

Here is a list of the Python keywords.  Enter any keyword to get more help.

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

2. Start a Python interpreter and use it as a Calculator.

```
>>> a = 10
>>> b = 30

>>> print ( type ( a ) )
    < class ' int ' >

>>> print ( type ( b ) )
    < class ' int ' >

>>> print ( a + b )
    40

>>> print ( a - b )
    -20

>>> print ( a * b )
    300

>>> print ( a / b )
    0.3333333333333333

>>> print ( a % b )
    10
```

```
>>> print ( a ** b )
    100000000000000000000000000000000
>>> print ( a // b )
    0
```

3. Write a program to calculate compound interest when principal, rate and number of periods are given.

The formula to calculate compound interest annually is given by:

$A = P (1 + R / 100)^t$

Compound Interest = A - P

Where,
A is amount,
P is the principal amount
R is the rate of interest,
T is the time span

```
# Python program to compute compound interest

p = int(input(" Enter the principal amount : "))

t = float(input(" Enter the time in years : " ))

r = float(input(" Enter the rate of interest : "))

# compute compound interest

amount = p * (pow((1 + r / 100), t))

ci = amount - p

# print

print(" Compound Interest : " , ci )
```

output

```
Enter the principal amount : 50000

 Enter the time in years : 3.8

 Enter the rate of interest : 2

 Compound Interest :  3907.6819031304913
```

4.Read the name, address, email and phone number of a person through the keyboard and print the details.

```
Name = input("Enter Person Name : ")

Address = input("Enter Person Address : ")

Email = input("Enter Person Email : ")
```

```
Phone_no = input("Enter Phone number : ")
print(" Please Confirm your provided Information\n " )
print(" Person Name : " ,Name )
print(" Person Address : " , Address )
print(" Person Email : " , Email )
print(" Person Phone_no : " , Phone_no )
```

output:

```
Enter Person Name : prasanna
Enter Person Address : beeramguda
Enter Person Email : abc@gmail.com
Enter Phone number : 78900000
 Please Confirm your provided Information
 Person Name :  prasanna
 Person Address :  beeramguda
 Person Email :  abc@gmail.com
 Person Phone_no :  78900000
```

5. Print the below triangle using for loop.

```
 5
 4 4
 3 3 3
 2 2 2 2
1 1 1 1 1
```

```
Num = int( input ( "Enter no of rows = " ) )
# reverse for loop from 5 to 1
for i in range ( 1 , Num + 1 ) :
        for j in range ( i ) :
                print ( Num , end = ' ' )
        Num -= 1
        print ( ' ' )
```

output:

Enter no of rows = 5

5

4 4

3 3 3

2 2 2 2

1 1 1 1 1

6.Write a program to check whether the given input is digit or lowercase character or uppercase character or a special character (use 'if-else-if' ladder)

```
ch = input ( " Enter a character : " )
if ( ord ( ch ) >= 65 and ord ( ch ) <= 90 ) :
        print ( ch , " is Upper Case Character " )
elif ( ord ( ch ) >= 97 and ord ( ch ) <= 122 ) :
        print ( ch , " is Lower Case Character " )
elif ( ord ( ch ) >= 48 and ord ( ch ) <= 57 ) :
        print ( ch , " is Digit " )
else:
        print ( ch , " is Symbol " )
```

output:

Enter a character : a

a is Lower Case Character

7.Python program to print all prime numbers in a given interval (use break)

```
lower_value = int ( input ( "Enter the Lowest Range Value: " ) )
upper_value = int ( input ( "Enter the Upper Range Value: " ) )
print ( "Prime numbers between " , lower_value , " and " , upper_value , " are : " )
for num in range ( lower_value , upper_value + 1 ) :
        # all prime numbers are greater than 1
```

```
    if num > 1 :
        for i in range ( 2 , num) :
            if ( num % i ) == 0 :
                break
        else :
            print ( num , end = " \t " )
```

output:

Enter the Lowest Range Value: 5

Enter the Upper Range Value: 36

Prime numbers between 5 and 36 are :

5 7 11 13 17 19 23 29 31

8.Write a program to convert a list and tuple into arrays.

```
import numpy as np
list = [ 1 , 2 , 3 , 4 , 5 , 6 , 7 , 8 ]
print ( " List to array: " )
array_1 = np . asarray ( list )
print ( type ( array_1 ) )
print ( array_1 )
print ( )
print ( )
tuple = ( [ 8, 4 , 6 ] , [ 1 , 2 , 3 ] )
print ( " Tuple to array: " )
array_2 = np . asarray ( tuple )
print ( type ( array_2 ) )
print ( array_2 )
```

output:

List to array:

< class ' numpy . ndarray ' >

[1 2 3 4 5 6 7 8]

Tuple to array:

< class ' numpy . ndarray ' >

[[8 4 6]

[1 2 3]]

9. Write a program to find common values between two arrays.

```
import numpy as np
array1 = np . array ( [ 5, 10 , 20 , 40 , 60 ] )
print ( " Array1 : " , array1 )
array2 = np . array ( [10 , 30 , 40 ] )
print ( " Array2 : " , array2 )
print ( " Common values between two arrays are : " )
print ( np . intersect1d ( array1 , array2 ) )
```

output:

[10 40]

10. Write a function called palindrome that takes a string argument and returns True if it is a palindrome and False otherwise. Remember that you can use the built-in function len to check the length of a string

```
def Palindrome(str):
        # Run loop from 0 to len/2
        for i in range( 0 , int (len(str) / 2)):
                if str[i] != str[len(str) - i - 1 ]:
                        return False
        return True
# main function
s = input(" Enter any string to check palindrome = ")
boolean = Palindrome(s)
if (boolean):
        print( s , " is palindrome ")
else :
```

```
        print( s , " is not palindrome ")
```

output:

```
Enter any string to check palindrome = madam
madam  is palindrome
```

11.Write a function called is_sorted that takes a list as a parameter and returns True if the list is sorted in ascending order and False otherwise.

```
def is_sorted(user_list):
        new_list = sorted(user_list)
        print(" Your Sorted list : " , sorted(user_list))
        if(new_list == user_list):
                return True
        else:
                return False
user_list = [int(x) for x in input("Please enter a list: ").split()]
print(" Your list : " , user_list)
print(is_sorted(user_list))
```

output:

```
Please enter a list: 7 8 9 4 5 6 10 12
 Your list :  [7, 8, 9, 4, 5, 6, 10, 12]
 Your Sorted list :  [4, 5, 6, 7, 8, 9, 10, 12]
False
```

12.Write a function called has duplicates that takes a list and returns True if there is any element that appears more than once. It should not modify the original list

```
def has_duplicate(my_list):
        for k in my_list:
                if my_list.count(k) > 1 :
                        return True
                        break
                else :
                        return False
user_list = input(" Please enter a list : ").split()
```

```
print(" Your list : " , user_list)
res = has_duplicate ( user_list )
if res :
        print( " The list contain duplicate values ")
else :
        print( " The list not contain duplicate values ")
```

output:

```
Please enter a list : 6 7 2 3 1 3 1 8 6
 Your list :  ['6', '7', '2', '3', '1', '3', '1', '8', '6']
 The list contain duplicate values
```

13.Write a function called remove_duplicates that takes a list and returns a new list with only the unique elements from the original. Hint: they don't have to be in the same order

```
def remove_duplicates(my_list):
        for k in my_list :
                if my_list . count(k) > 1 :
                        del my_list[my_list.index(k)]
user_list = input(" Please enter a list : ").split()
print(" Your list : " , user_list)
remove_duplicates(user_list)
print(" Your list without duplicates : \n " , user_list)
```

output:

```
Please enter a list : 6 7 2 3 1 3 1 8 6
 Your list :  ['6', '7', '2', '3', '1', '3', '1', '8', '6']
 Your list without duplicates :
  ['7', '2', '3', '1', '8', '6']
```

14.The wordlist I provided, words.txt, doesn't contain single letter words. So you might want to add "I", "a", and the empty string.

Text File: words.txt

apple

banana

cat

dog

elephant

(single_letter.py)

```
def load_words(filename):
    with open(filename, "r") as file:
        words = [line.strip() for line in file]

    # Add missing single-letter words and empty string
    if "" not in words:
        words.append("")
    if "I" not in words:
        words.append("I")
    if "a" not in words:
        words.append("a")

    return words
# Sample usage
words = load_words("words.txt")
print("Total number of words:", len(words))
print("Sample words:", words[:10])
```

output:

Total number of words: 8

Sample words: ['apple', 'banana', 'cat', 'dog', 'elephant', '', 'I', 'a']

15.Write a python code to read dictionary values from the user. Construct a function to invert its content. i.e., keys should be values and values should be keys

```
import ast
```

```
def swap_dict(old_dict):
        return { value : key for key , value in old_dict.items () }
old_dict = input(" Please enter a dictionary : \n ")
old_dict = ast.literal_eval(old_dict)
# Printing original dictionary
print(" Original dictionary is : \n " , old_dict)
print()
new_dict = swap_dict(old_dict)
# Printing new dictionary
print(" New dictionary is : \n " , new_dict)
```

output:

```
Please enter a dictionary :
 { 'A': 65 , 'B' : 66 , 'C' : 67 , 'D' : 68 }
 Original dictionary is :
  {'A': 65, 'B': 66, 'C': 67, 'D': 68}
 New dictionary is :
  {65: 'A', 66: 'B', 67: 'C', 68: 'D'}
```

16.Add a comma between the characters. If the given word is 'Apple', it should become 'A,p,p,l,e'

```
def add_comma(x):
        return ','.join(x)

in_str = "Apple"
print(add_comma(in_str))
```

output:

```
A,p,p,l,e
```

17.Remove the given word in all the places in a string?

```
def remove_all_words(in_str,sub):
```

```
    my_list = in_str.split( )
    for i in my_list :
        if i == sub :
            del my_list[my_list.index(i)]
    final_string = ' '
    for i in my_list :
        final_string = final_string + i + ' '
    return final_string
str_in = input(" Enter any string = ")
sub = input(" Enter sub string ( which word you want to delete ) = ")
result = remove_all_words(str_in , sub)
print(" Updated string after removal sub_string : \n " , result)
```

output:

```
Enter any string = hai,this is python,welcome to MLRITM
 Enter sub string ( which word you want to delete ) = MLRITM
 Updated string after removal sub_string :
  hai,this is python,welcome to
```

18.Write a function that takes a sentence as an input parameter and replaces the first letter of every word with the corresponding upper case letter and the rest of the letters in the word by corresponding letters in lower case without using a built-in function?

```
def replace_first(str_in):
    x = list(str_in)
    i = 0
    while i < len(x):
        # consider first character of word
        if i == 0 :
            if ord(x[i]) >= 97 and ord(x[i]) <= 122 :
                x[i] = chr(ord(x[i]) - 32)
```

```
            elif x[i] == ' ':
                i = i + 1
                if ord(x[i]) >= 97 and ord(x[i]) <= 122 :
                    y = ord(x[i])
                    x[i] = chr(y - 32)
            else :
                if ord(x[i]) >= 65 and ord(x[i]) <= 92 :
                    x[i] = chr(ord(x[i]) + 32)
            i = i + 1
        str_out = ' '
        for j in x :
            str_out = str_out + j
        return str_out
str_in = input(" Enter any string = ")
str_op = replace_first(str_in)
print(str_op)
```

output:

```
Enter any string = welcome TO MLritm
 Welcome To Mlritm
```

19.Writes a recursive function that generates all binary strings of n-bit length

```
def generate_binary_number (length, binary_string):
    if len(binary_string) == length:
        print(binary_string)
        return
    generate_binary_number (length, binary_string + '0')
    generate_binary_number (length, binary_string + '1')
n = int (input ("Enter number of bits to generate binary set = "))
```

```
print ("Binary numbers are: ")
generate_binary_number (n, ")
```

output:

```
Enter number of bits to generate binary set = 3
Binary numbers are:
000
001
010
011
100
101
110
111
```

20.Write a python program that defines a matrix and prints

```
R = int(input(" Enter the number of rows : "))
C = int(input(" Enter the number of columns : "))
# Initialize matrix
matrix = []
print(" Enter the entries row - wise : ")
# For user input
for i in range(R):
        a = []
        for j in range(C):
                a.append(int(input()))
        matrix.append(a)
print()
# For printing the matrix
print(" Matrix Values are: \n ")
```

```
for i in range(R) :
        for j in range(C):
                print(matrix[i][j] , end = " ")
        print()
```

output:

```
Enter the number of rows : 3
 Enter the number of columns : 3
 Enter the entries row - wise :
1
2
3
4
5
6
7
8
9
 Matrix Values are:
1 2 3
4 5 6
7 8 9
```

21. Write a python program to perform multiplication of two square matrices

In this program we use **"numpy"** package to create matrices. So we need to install **"numpy"** package using the following commands.

To Install in Windows:

```
pip install numpy
```

To Install in Linux:

```
sudo apt-get install numpy
```

PROGRAM: (matrix_mul.py)

```
import numpy as np
a = np.array([[ 1 , 2 ] , [ 4 , 5 ]])
b = np.array([[ 6 , 5 ] , [ 3 , 2 ]])
c = np.dot(a,b)
print(" Multiplication of two Square Matrices is \n " , c)
```

output:

```
Multiplication of two Square Matrices is
  [[12 9]
 [39 30]]
```

22.How do you make a module? Give an example of construction of a module using different geometrical shapes and operations on them as its functions

area.py

```
 from math import pi
def rectangle(l,b):
        return l * b
def square(s):
        return s * s
def circle(r):
        return pi * r * r
def triangle(l,b):
        return 1 / 2 * l * b
```

geometrical_shapes.py

```
 import area
l = int(input(" Enter value of height = "))
b = int(input(" Enter value of width = "))
s = int(input(" Enter value of side = "))
r = int(input(" Enter value of radius = "))

print(" Area of rectangle = " , area.rectangle(l,b))
print(" Area of square = " , area.square(s))
print(" Area of circle = " , area.circle(r))
print(" Area of triangle = " , area.triangle(l,b))
```

output:

```
Enter value of height = 6
 Enter value of width = 6
 Enter value of side = 6
 Enter value of radius = 6
 Area of rectangle =  36
 Area of square =  36
 Area of circle = 113.09733552923255
 Area of triangle =  18.0
```

23.Use the structure of exception handling all general-purpose exceptions

exception_handling.py

```
 def function(a,b):
        c = a / b
        print ( c )
try :
        a = int(input(" Enter a value : "))
        b = int(input(" Enter b value : "))
        function(a,b)
```

```
except ValueError :
        print(" Value Error Occurred \t Pease Enter Correct integer Values ")
except ZeroDivisionError :
        print(" Zero Division Error Occurred \t Can't division by zero ")
else :
        print(" Successfully executed without errors ")
```

output:

```
Enter a value : 5
 Enter b value : 0
 Zero Division Error Occurred          Can't division by zero
```

24. Write a function called draw_rectangle that takes a Canvas and a Rectangle as arguments and draws a representation of the Rectangle on the Canvas.

PROGRAM: (draw_rectangle.py)

```
 from tkinter import *
# Create an instance of tkinter frame
root = Tk ()
# Create Title
root.title(" Rectangle ")
# create canvas object
widget = Canvas(root, width = 700 , height = 400 , bg = "grey")
#function
def draw_rectangle(windget,rect):
        print(rect)
# Read Co-ordinates
x1 = int(input(" Enter x1 value = " ))
y1 = int(input(" Enter y1 value = " ))
x2 = int(input(" Enter x2 value = " ))
y2 = int(input(" Enter y2 value = " ))
```

```
# create object for rectangle
rect = widget.create_rectangle(x1,y1,x2,y2)
# call function by passing canvas & rectangle objects as argument
draw_rectangle(widget,rect)
widget.pack()
root.mainloop()
```

OUTPUT:

```
Enter x1 value = 100
Enter y1 value = 30
Enter x2 value = 250
Enter y2 value = 100
```

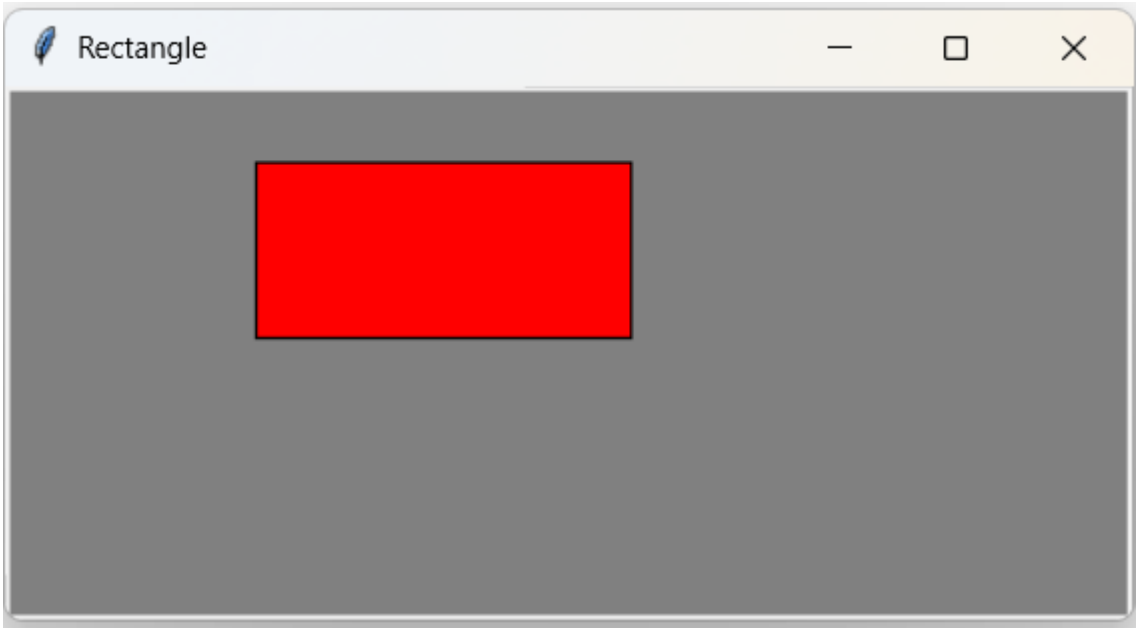
25. Add an attribute named color to your Rectangle objects and modify draw_rectangle so that it uses the color attribute as the fill color.

(fill_rectangle.py)

```
from tkinter import *
# Create an instance of tkinter frame
root = Tk ()
# Create Title
root.title(" Rectangle ")
# create canvas object
widget = Canvas(root, width = 700 , height = 400 , bg = "grey")
#function
def draw_rectangle(windget,rect):
        print(rect)
# Read Co-ordinates
x1 = int(input(" Enter x1 value = " ))
y1 = int(input(" Enter y1 value = " ))
x2 = int(input(" Enter x2 value = " ))
y2 = int(input(" Enter y2 value = " ))
# create object for rectangle
rect = widget.create_rectangle(x1,y1,x2,y2,fill="red")
# call function by passing canvas & rectangle objects as argument
draw_rectangle(widget,rect)
widget.pack()
root.mainloop()
```

OUTPUT:

```
Enter x1 value = 100
Enter y1 value = 30
Enter x2 value = 250
Enter y2 value = 100
```



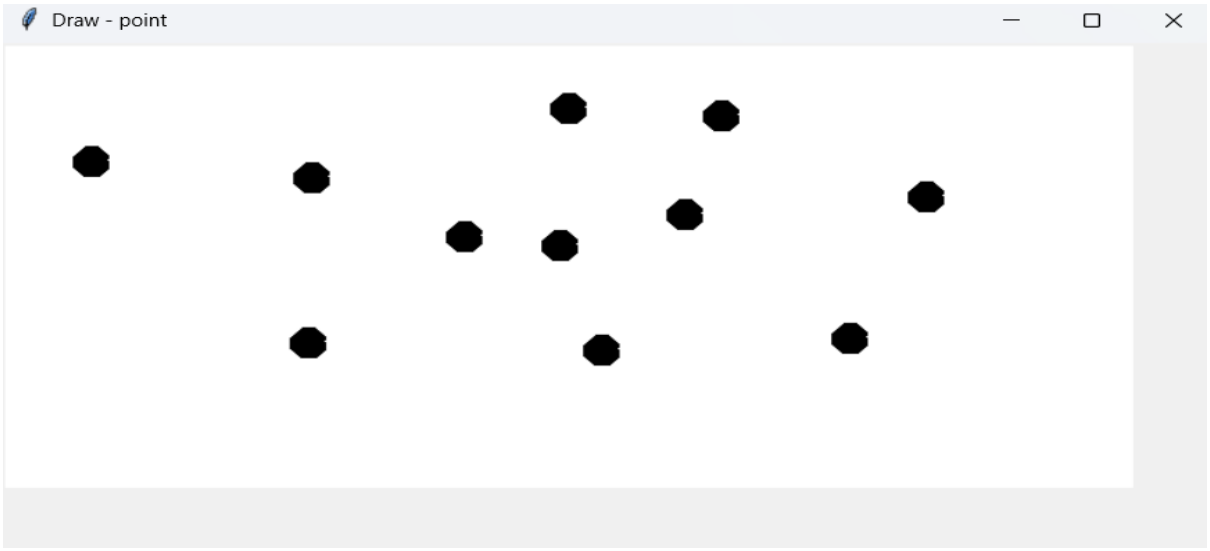
26. Write a function called draw_point that takes a Canvas and a Point as arguments and draws a representation of the Point on the Canvas.

(draw_point.py)

```
 from tkinter import *
# Create an instance of tkinter frame or window
win = Tk ()
# Create Title
win.title(" Draw - point ")
# Set the size of the window
win.geometry("700x350")
# Define a function to draw the line between two points
def draw_point(event):
        x1= event.x
        y1= event.y
        x2= event.x
        y2= event.y
        # Draw an oval in the given co-ordinates
        canvas.create_oval(x1,y1,x2,y2,fill="black",width = 20 )
```

```
# Create a canvas widget
canvas = Canvas(win,width = 650,height = 300,background = "white")
canvas.grid(row = 0 ,column = 0)
canvas.bind('<Button-1>',draw_point)
click_num = 0
win.mainloop()
```

OUTPUT:



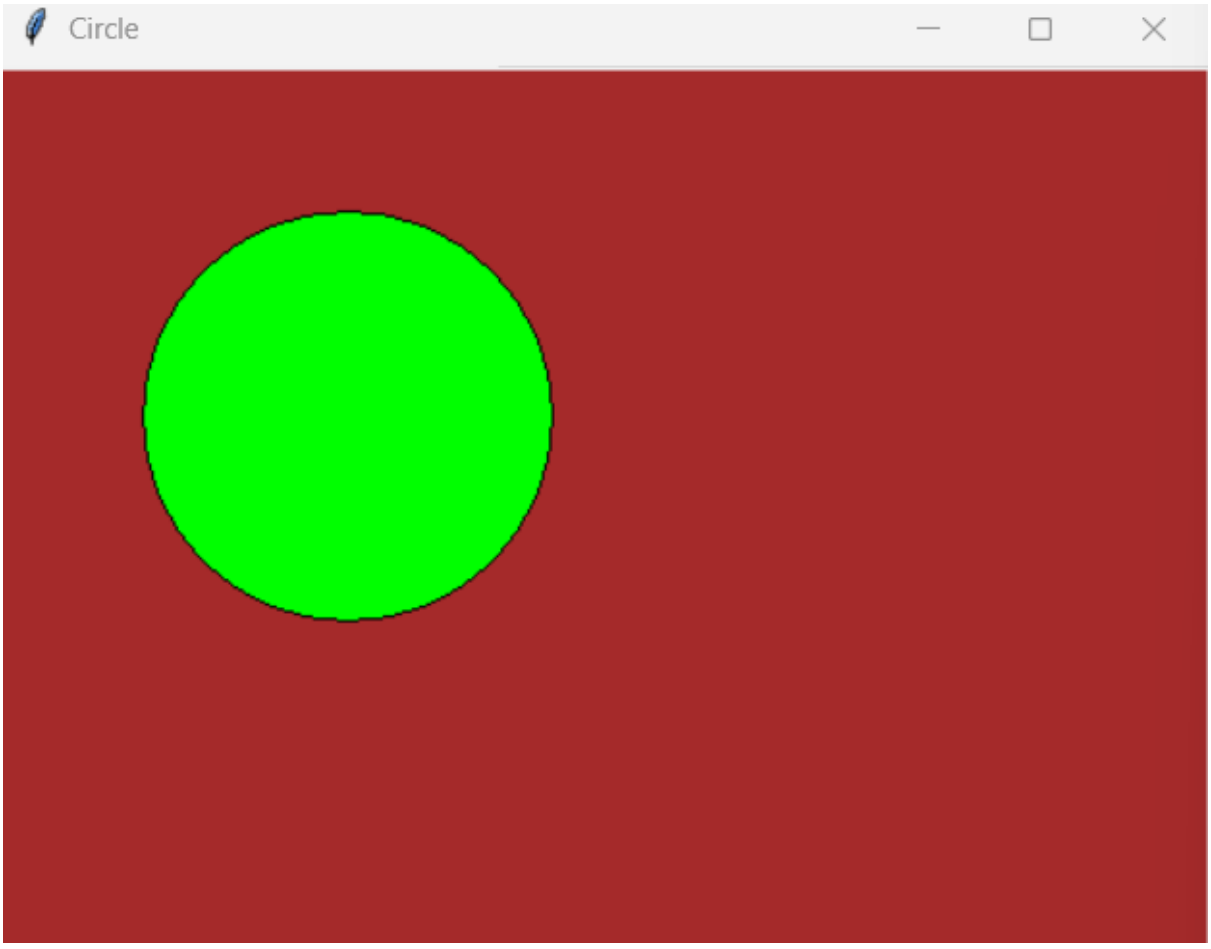
27.Define a new class called Circle with appropriate attributes and instantiate a few Circle objects. Write a function called draw_circle that draws circles on the canvas.

(draw_circle.py)

```
from tkinter import *
#Create an instance of tkinter frame
root = Tk()
# Create Title
root.title(" Circle ")
# create canvas object
widget = Canvas(root, width=500,height=400 , bg = "brown" )
#function
def draw_circle(widget,circle):
        print(circle)
```

```
# Read Co-ordinates
x1 = int(input(" Enter x1 value = "))
y1 = int(input(" Enter y1 value = "))
x2 = int(input(" Enter x2 value = "))
y2 = int(input(" Enter y2 value = "))
# create object for rectangle
circle = widget.create_oval(x1 , y1 , x2 , y2 , fill = "#000fff000")
# call function by passing canvas & rectangle objects as argument
draw_circle(widget,circle)
widget.pack()
root.mainloop()
```

OUTPUT:



28. Write a python code to read a phone number and email-id from the user and validate it for correctness.

validate_phmail.py

```
import re
# Make a regular expression for validating an Email
regex = r'^\b[A-Za-z0-9._%+-]+[@]\w[A-Za-z]+[.]\w[A-Z|a-z]{2,5}$'
# Make a regular expression for validating a Phone Number
Pattern = '^\\+?[6-9][0-9]{9}$'
# Define a function for validating an Email
def is_check_email(email):
        if(re.fullmatch(regex,email)):
                print(" Valid Email - ID ")
        else:
                print(" Invalid Email - ID ")
# Define a function for validating a PHONE NUMBER
def is_check_Phone_number(mobile_no):
        if(re.search(Pattern,mobile_no)):
                print(" Valid Phone Number ")
        else:
                print(" Invalid Phone Number ")
# Driver Code
if __name__ == '__main__':
        email = input(" Enter your MAIL-ID = ")
        phone_number = input(" Enter your PHONE NUMBER = ")
        is_check_email(email)
        is_check_Phone_number(phone_number)
```

output:

```
Enter your MAIL-ID = prasanna@123.mlritm.ac.in
 Enter your PHONE NUMBER = 5678899
 Invalid Email - ID
```

Invalid Phone Number

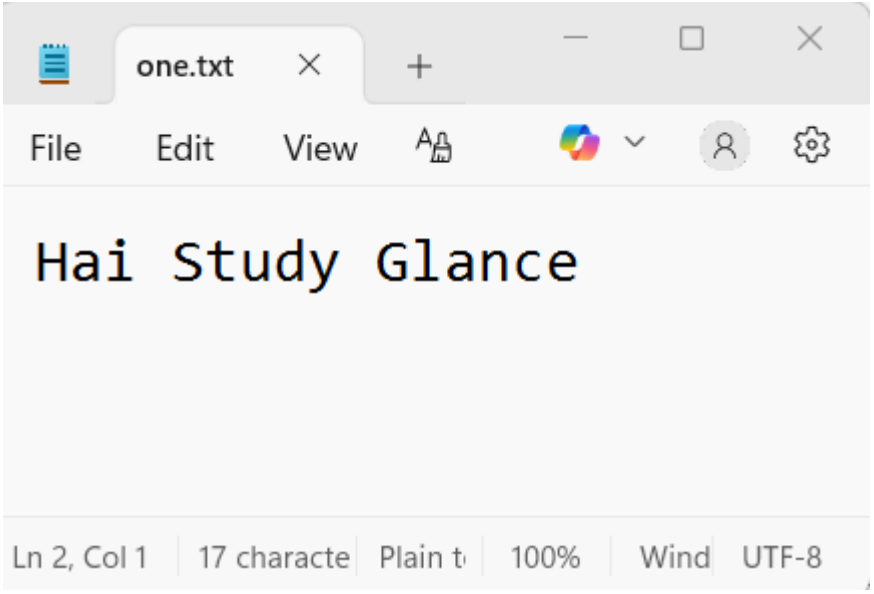
29. Write a Python code to merge two given file contents into a third file

merge_files.py)

```
# Open the source text file - 1
file1 = open('one.txt','r')
content1 = file1.read()
file1.close()
# Open the source text file - 2
file2 = open('two.txt','r')
content2 = file2.read()
file2.close()
# Open the destination file to merge content
file3 = open('three.txt','w')
file3.write(content1 + content2 )
file3.close()
# Print merge text file
file3 = open('three.txt','r')
print(file3.read())
file3.close()
```

OUTPUT:

```
Hai Study Glance
How are You?
```



30. Write a Python code to open a given file and construct a function to check for given words present in it and display on found.

check_words.py

```
def search_str(file_path):
        search_word = input(" Enter a word you want to search in file : ")
        with open(file_path , 'r') as file :
                # read all content of a file
                content = file.readlines()
                # check if string present in a file
                for word in content :
                        if word.find(search_word) != -1:
                                print('The search word exists in the file & available at line = '
,content.index(word))
                                break
                        else :
                                print(" string does not exist in a file ")
        file.close()


file = open("search.txt",'w')
```

```
Lines = ["Hai \n ","This is Python Programming \n ","Welcome to study glance \n "]
file.writelines(Lines)
file.close()
search_str('search.txt')
```

OUTPUT:

```
Enter a word you want to search in file : to
string does not exist in a file
string does not exist in a file
The search word exists in the file & available at line =  2
```

31. Write a Python code to Read text from a text file, find the word with most number of occurrences

Input Data file ("data.txt"):

This is Study Glance. Study Galnce provides various Study related meterials.

PROGRAM: (most_word.py)

```
count = 0
word = " "
max_Count = 0
words = []

#Opens a file in read mode
file = open("data.txt" , "r")

#Gets each line till end of file is reached
for line in file :
```

```
        #Splits each line into words
        string = line.lower().replace(',',' ').replace('.','').split(" ");
        #Adding all words generated in previous step into words
        for s in string:
                words.append(s)

#Determine the most repeated word in a file
for i in range(0,len(words)):
        count = 1
        #Count each word in the file and store it in variable count
        for j in range(i+1 , len(words)):
                if(words[i] == words[j]):
                        count = count + 1
        # If maxCount is less than count then store value of count in maxCount
        # and corresponding word to variable word
        if (count > max_Count):
                max_Count = count
                word = words[i]
print(" Most repeated word is = " + word)
file.close()
```

OUTPUT:

Most repeated word is = study

32. Write a function that reads a file file1and displays the number of words, number of vowels, blank spaces, lower case letters and uppercase letters

Input Data file ("search.txt"):

Hello python

this is study glance

123 456

PROGRAM: (counting.py)

```
def counting(filename):
    txt_file = open(filename , "r")
    no_of_vowels = 0
    no_of_words = 1
    no_of_lines = 1
    no_of_spaces = 0
    no_of_uppercase = 0
    no_of_lowercase = 0
    no_of_digits = 0
    # Make a vowels list
    vowels_list = ['a','e','i','o','u','A','E','I','O','U']
    # Iterate over the characters present in file
    for ch in txt_file.read():
        if ch == "\n" or ch == ' ':
            no_of_words = no_of_words + 1
        if ch == ' ':
            no_of_spaces = no_of_spaces + 1
        if ch in vowels_list:
            no_of_vowels = no_of_vowels + 1
        if ch == "\n":
            no_of_lines = no_of_lines + 1
        if(ord(ch) >= 97 and ord(ch) <= 122):
            no_of_lowercase = no_of_lowercase + 1
        if(ord(ch) >= 65 and ord(ch) <= 90):
            no_of_uppercase = no_of_uppercase + 1
        if(ord(ch) >= 48 and ord(ch) <= 57):
            no_of_digits = no_of_digits + 1
```

```
        # Print the desired output on the console.
        print(" Number of vowels = ", no_of_vowels)
        print(" Number of words = " , no_of_words)
        print(" Number of Blank spaces = " , no_of_spaces)
        print(" New Lines = " , no_of_lines)
        print(" Number of lower case characters in = " , no_of_lowercase)
        print(" Number of upper case characters = " , no_of_uppercase)
        print(" Number of digits = " , no_of_digits)
# Driver Code
filename = input(" Enter text file name = ")
counting(filename)
```

OUTPUT:

```
Enter text file name = search.txt
Number of vowels =  8
Number of words =  8
Number of Blank spaces =  5
New Lines =  3
Number of lower case characters in =  27
Number of upper case characters =  1
Number of digits =  6
```

33. Import numpy, Plotpy and Scipy and explore their functionalities.

```
import numpy as np
import matplotlib.pyplot as plt
from scipy import stats, integrate
```

1. NumPy (Numerical Computation)

```
a = np.array([1, 2, 3, 4])
b = np.array([5, 6, 7, 8])

print("Array a:", a)
print("Array b:", b)

print("Addition:", a + b)
print("Multiplication:", a * b)
print("Mean of a:", np.mean(a))
```

OUTPUT:

```
Array a: [1 2 3 4]
Array b: [5 6 7 8]
Addition: [ 6  8 10 12]
Multiplication: [ 5 12 21 32]
Mean of a: 2.5
```

2. Matplotlib (Pyplot – Data Visualization)

```
x = np.linspace(0, 5, 6)
y = x**2

print("x values:", x)
print("y values:", y)

plt.plot(x, y)
plt.title("y = x^2")
plt.xlabel("x")
plt.ylabel("y")
plt.show()
```

OUTPUT:

```
x values: [0. 1. 2. 3. 4. 5.]
y values: [ 0.  1.  4.  9. 16. 25.]
```

3. SciPy (Scientific Computing)

data = [1, 2, 2, 3, 4]

mode = stats.mode(data)

print("Mode:", mode.mode)

OUTPUT:

Mode: [2]

(b) Integration

result, error = integrate.quad(lambda x: x**2, 0, 3)
print("Integration Result:", result)
OUTPUT:

Integration Result: 9.0

34. Install NumPypackage with pip and explore it.

1. Installing NumPy using pip

Open Command Prompt / Terminal and type:

pip install numpy
OUTPUT:

Collecting numpy
Downloading numpy-1.x.x-cp3x.whl
Installing collected packages: numpy
Successfully installed numpy-1.x.x

◈ 2. Importing NumPy

import numpy as np

◈ 3. Exploring NumPy Functionalities

(a) Creating Arrays

a = np.array([1, 2, 3, 4])
print("Array:", a)
OUTPUT:

Array: [1 2 3 4]

(b) Mathematical Operations

```
print("Square:", np.square(a))
print("Sum:", np.sum(a))
print("Mean:", np.mean(a))
```

OUTPUT:

```
Square: [ 1  4  9 16]
Sum: 10
Mean: 2.5
```

(c) Multi-dimensional Array

```
b = np.array([[1, 2], [3, 4]])
print("2D Array:\n", b)
```

OUTPUT:

```
2D Array:
[[1 2]
 [3 4]]
```

(d) Array Range

```
c = np.arange(0, 10, 2)
print("Range Array:", c)
```

OUTPUT:

```
Range Array: [0 2 4 6 8]
```

35. Write a program to implement Digital Logic Gates – AND, OR, NOT, EX-OR

```
# Logic Gates Implementation

# AND Gate
def AND(a, b):
    return a & b

# OR Gate
def OR(a, b):
    return a | b

# NOT Gate
def NOT(a):
    return ~a & 1   # Ensures output is 0 or 1

# XOR Gate
def XOR(a, b):
    return a ^ b

# Inputs
a = 1
b = 0

print("Inputs: a =", a, ", b =", b)

print("AND Gate:", AND(a, b))
print("OR Gate:", OR(a, b))
print("NOT Gate (a):", NOT(a))
print("XOR Gate:", XOR(a, b))
```

OUTPUT:

Inputs: a = 1 , b = 0
AND Gate: 0
OR Gate: 1
NOT Gate (a): 0
XOR Gate: 1

36. Write a GUI program to create a window wizard having two text labels, two text fields and two buttons as Submit and Reset.

```
import tkinter as tk

# Create window

root = tk.Tk()

root.title("Simple Form Wizard")

root.geometry("300x200")


# Labels

label1 = tk.Label(root, text="Name:")

label1.grid(row=0, column=0, padx=10, pady=10)


label2 = tk.Label(root, text="Age:")

label2.grid(row=1, column=0, padx=10, pady=10)


# Text fields

entry1 = tk.Entry(root)

entry1.grid(row=0, column=1)


entry2 = tk.Entry(root)

entry2.grid(row=1, column=1)


# Submit function

def submit():

    name = entry1.get()

    age = entry2.get()

    print("Submitted Data:")
```

```
    print("Name:", name)
    print("Age:", age)

# Reset function
def reset():
    entry1.delete(0, tk.END)
    entry2.delete(0, tk.END)

# Buttons
submit_btn = tk.Button(root, text="Submit", command=submit)
submit_btn.grid(row=2, column=0, pady=10)

reset_btn = tk.Button(root, text="Reset", command=reset)
reset_btn.grid(row=2, column=1, pady=10)

# Run window
root.mainloop()
```

OUTPUT:

```
| Name: [_____________] |
| Age: [_____________] |
| |
| Submit Reset |
```