Expt. No:
Date :

AIM:
Write a python program to find what version of python you are using.

ALGORITHM:
Step 1: Import the 'sys' module provides access to some variables used or maintained by the Interpreter and to function that interact strongly with the Interpreter.
Step 2:sys.version is used to print the version of python that are in use.

PROGRAM:

```
importsys
sys.version
```

OUTPUT:

```
3.10.4
```

RESULT:

The program is executed and the output is verified.

Expt. No:
Date :

AIM:

Write the python program to determine python shell is execute in 32-bit or 64-bit operating system.

ALGORITHM:

Step 1: Import the ‘platform’ module for inbuilt library provided in python program.
Step 2: platform.architecture()[0] is used to print the os version of python used in a computer.

PROGRAM:

```
import platform
platform.architecture()[0]
```

OUTPUT:

```
‘64bit’
```

RESULT:

The Program Is Executed And The Output Is Verified.

Expt. No:
Date :

AIM:

Write a python program to get os name, platform, and release information through python shell.

ALGORITHM:

Step 1: Import the ‘platform’ module form inbuilt library.
Step 2: Import the ‘os’ module from inbuilt library.
Step 3: Now print the os.name, platform.system() and platform.release()

PROGRAM:

```
import platform
import os
print(os.name)
print(platform.system())
print(platform.release())
```

OUTPUT:

```
nt
windows
10
```

RESULT:

The progrm is executed and the output is verified.

Expt. No:
Date :

AIM:

Write the python program to display current date and time.

ALGORITHM:

Step 1: import the ‘date time’ module from inbuilt library.
Step 2: Print the ‘now.strftime’ and print the date and time.

PROGRAM:

```
import datetime
now=datetime.datetime.now()
print(now.strftime(“%Y-%M-%d  %H:%M:%S”))
```

OUTPUT:

```
2023-04-10   14:40:23
```

RESULT:

The program is executed and the output is verified.

Expt. No:
Date :

AIM:

Write the python program to calculate area of the circle after get radius from user.

ALGORITHM:

```
Step 1: Import math module and store as m.
Step 2: Get the input from user as radius.
Step 3: Calculate the radius of the circle.
Step 4: print the radius of the circle.
```

PROGRAM:

```
import math as m
radius=float(input("please enter the radius: "))
area=m.pi*(radius**2)
print("The area of the circle is: ",area)
```

OUTPUT:

```
please enter the radius: 3
The area of the circle is: 28.274333882308138
```

RESULT:

The program is executed and the output is verified.

Expt. No:
Date   :

AIM:

Write a python program to calculate the number of days between two dates.

ALGORITHM:

```
Step 1: import duilt in function called ‘datetime’ module from python
        library.
Step 2: input the two dates in date type format and subtract them.
```

PROGRAM:

```
from datetime import date as date_n

def number_of_days(date_1,date_2):
  return(date_2-date_1).days

date_1=date_n(2023,9,10)
date_2=date_n(2025,2,4)
print(“Number of days between the given dates are:
        “,number_of_days(date_1,date_2),”days”)
```

OUTPUT:

```
Number of days between the given dates are: 513 days
```

RESULT:

The program is executed and the output is verified.

Expt. No:
Date :

AIM:

Write the python program to get the volume of the sphere with radius six.

ALGORITHM:

```
Step 1: enter the radius value as 6
Step 2: enter the value for pi=3.14
Step 3: with the help of  formula '4.0/3.0*pi*r**3' calculate the volume
        of the sphere.
```

PROGRAM:

```
r=6
pi= 3.14
volume=4.0/3.0*pi*r**3
print("The volume of the sphere is: ",volume)
```

OUTPUT:

```
The volume of the sphere is: 904.32
```

RESULT:

The program is executed and the output is verified.

Expt. No:
Date :

AIM:

Write a python program to test whether a passed letter is vowel or not.

ALGORITHM:

```
Step 1: get the input from the user as letter.
Step 2: if the letter is in ‘a,e,I,o,u’ then it is a vowel, else it is not a vowel.
Step 3: print it is a vowel or not.
```

PROGRAM:

```
l=input(“input a letter of the alphabet: “)
if l in (‘a’,’e’,’I’,’o’,’u’):
   print(“%s is vowel.” %l)
else:
   print(“%s is not a vowel.”%l)
```

OUTPUT:

```
Input a letter of the alphabet: u
U is a vowel.
```

RESULT:

The program is executed and the output is verified.

Expt. No:
Date :

AIM:
Write a python program that computes the greatest common divisor(GCD) of the integer.

ALGORITHM:

```
Step 1: enter the two numbers.
Step 2: if b==0 then return a, else return the hcf of b,a%b
Step 3: print the hcf of the two numbers.
```

PROGRAM:

```
def hcf(a,b):
        if(b==0):
              return a
        else:
       return hcf(b,a%b)
a=60
b=48
print(“The gcd of 60 and 48 is: “,end=” “)
print(hcf(60,48))
```

OUTPUT:

```
The gcd of 60 and 48 is: 12
```

RESULT:
The program is executed and the output is verified.

Expt. No:
Date :

AIM:

Write a python program to find the least common multiple(LCM) of two positive integers.

ALGORITHM:

```
Step 1: LCM of two number will at least be equal or greater than
        max(num1,num2).
Step 2: largerst possibility of LCM will be num1*num2
```

PROGRAM:

```
num1=12
num2=14
for i in range(max(num1,num2),1+(num1*num2)):
    if i%num1==I % num2==0:
        lcm=I
        break
print("LCM of", num1, "and", num2, "is", lcm)
```

OUTPUT:

```
LCM of 12 and 14 is 84
```

RESULT:

The program is executed and the output is verified.

Expt. No:
Date :

AIM:

Python program to check whether a digit is present in a number.

ALGORITHM:

```
Step 1:  Start
Step 2:  Read a number form where the digit needs to be found.
Step 3:  Read a digit that needs to be found.
Step 4:  Convert the given number into a string.
          str(num)
Step 5:  If str(f_num) in n, then print the digit is found.
Step 6:  else print the digit is not found.
Step 7:  End
```

PROGRAM:

```
num = int(input("Enter a number: "))
f_num = int(input("Enter a digit: "))
n = str(num)
if str(f_num) in n:
print("The digit is found.")
else:
print("The digit is not found.")
```

OUTPUT:

```
Enter a number: 239
Enter a digit: 7
The digit is not found.
```

RESULT:

Thus the python program to find whether a digit is present in a number is executed successfully.

Expt. No:
Date :

AIM:

Python program to count number of vowels and consonants in a string.

ALGORITHM:

Step 1: Start
Step 2: Get a string form the user
Step 3: Initialize the values of vowels and consonants count to zero.
Step 4: using for loop check allthe characters of the string and increase vowels and consonants count
Step 5: Print vowels and consonants
Step 6: End

PROGRAM:

```
str=input("Enter a string: ");
vowels=0
consonants=0
for i in str:
if(i == 'a'ori == 'e'ori == 'i'ori == 'o'ori == 'u' or
i == 'A'ori == 'E'ori == 'I'ori == 'O'ori == 'U' ):
        vowels+=1;
else:
      consonants+=1;
print("The number of vowels:",vowels);
print("The number of consonant:",consonants);
```

OUTPUT:

```
Enter a string: Python Programming
The number of vowels:  4
The number of consonant:  14
```

RESULT:

Thus the python program to find the vowel and consonants count in a string is executed successfully.

Expt. No:
Date :

AIM:

Python program to switch case the string and print.

ALGORITHM:

```
Step 1: Start
Step 2: read the string value from the user
Step 3: using for loop check each character whether it is in lower case or
in upper case
Step 4: if the character is in lower case then change it to upper case
Step 5: else change it to lower case
Step 6: print the string
Step 7: End
```

PROGRAM:

```
string = input("Enter a string: ")
res = ""
for i in range(len(string)):
   if string[i].islower():
      res = res + string[i].upper()
   else:
      res = res + string[i].lower()
print(str(res))
```

OUTPUT:

```
Enter a string: pythoNProGraMMinG
PYTHOnpROgRAmmINg
```

RESULT:

Thus the python program to swap case the string is executed successfully.

Expt. No:
Date :

AIM:

Python program to find max,min,mean,median,mode,sort a list of Integer

ALGORITHM:

```
Step 1: Start
Step 2: Get the size of the list from the user
Step 3: Get the elements of the list from the user
Step 4: Find the sum of the list elements and divide it by the size of the list
Step 5: Print mean value
Step 6: Find middle element of the list and print the median
Step 7: Using for loop sort the elements of the list in ascending order and print it
Step 8: Print the largest element of the list using loop and comparing with each element
Step 9: Print the smallest element of the list using loop and comparing with each element
Step 10: Compare all the elements of the list with each other and find the most repeated element and print the mode value
Step 11: End
```

PROGRAM:

```
n=int(input("Enter the size of the list: "))
lst = list(map(int,input().split()))
sum_lst = sum(lst)
print("mean = ",sum_lst//n)
if n%2 == 0:
   m1= lst[n//2]
   m2 = lst[n//(2-1)]
   m = (m1+m2)/2
else:
   m = lst[n//2]
print("Median = ",m)
for i in range(len(lst)):
   for j in range(i+1,len(lst)):
      if lst[i] >lst[j]:
lst[i],lst[j] = lst[j],lst[i]
print("Sorted list = ",lst)
l= None
```

```
for i in lst:
   if l is None or l < i:
      l=i
print("Max value = ",l)
s=None
for i in lst:
   if m is None or m > i:
      m=i
print("Min value = ",m)
count=0
max_count=0
val = 0
for i in range(0,len(lst)):
   for j in range(i+1,len(lst)):
      if lst[i] == lst[j]:
         count += 1
   if count >max_count:
max_count=count
val = lst[i]
print("Mode = ",val)
```

<u>OUTPUT:</u>

```
Enter the size of the list: 9
67 54 98 67 24 84 56 92 67
mean =  67
Median =  24
Sorted list =  [24, 54, 56, 67, 67, 67, 84, 92, 98]
Max value =  98
Min value =  24
Mode =  67
```

<u>RESULT:</u>

Thus the python program to find max,min,mean,median,mode,sort a list of Integersis executed successfully.

Expt. No:
Date :

AIM:

Python program to convert Base 10 to Base 2, Base 8 and Base 16.

ALGORITHM:

```
Step 1: Start
Step 2: read a decimal number from the user
Step 3: convert the decimal number into an integer
Step 4: print the number in base 8 using oct() function
Step 5: print the number in base 16 using hex() function
Step 6: print the number in base 2 using bin() function
Step 7: End
```

PROGRAM:

```
b_num = float(input("Enter a decimal number: "))
num = int(b_num)
print("Base 8: ",oct(num))
print("Base 16: ",hex(num))
print("Base 2: ",bin(num))
```

OUTPUT:

```
Enter a decimal number: 89.65
Base 8:  0o131
Base 16:  0x59
Base 2:  0b1011001
```

RESULT:

Thus the python program to convert Base 10 to Base 2, Base 8 and Base 16is executed successfully.

Expt. No:
Date :

AIM : To write a python program to swap two numbers.

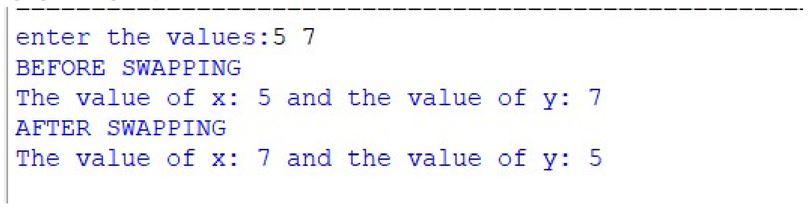
ALGORITHM:

```
STEP 1: Start
STEP 2: Get the input values
STEP 3: x , y=y , x
STEP 4: Print the values after swapping
STEP 5: Stop
```

PROGRAM:

```
x,y=map(int,input("enter the values:").split())
print("BEFORE SWAPPING:")
print("The value of x:",x,"and the value of y:",y)
print("AFTER SWAPPING:")
print("The value of x:",x,"and the value of y:",y)
```

OUTPUT:



```
enter the values:5 7
BEFORE SWAPPING
The value of x: 5 and the value of y: 7
AFTER SWAPPING
The value of x: 7 and the value of y: 5
```

RESULT:

Thus the above program to swap two numbers has been verified successfully

Expt. No:
Date :

AIM: To write a program to perform linear search using iteration.

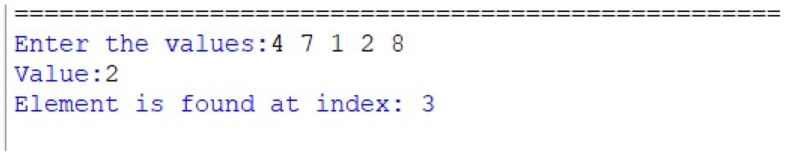
ALGORITHM:

```
STEP 1: Start
STEP 2: Get the values for list and x
STEP 3: Check whether the element x is present in the list by sequential movement.
STEP 4: If the element matches then print the index value.
STEP 5: If the element is not present then print as not found
STEP 6:Stop
```

PROGRAM:

```
l=list(map(int,input("Enter the values:").split()))
x=int(input("Value:"))
count=-1
for I in l:
    count+=1
    if(i==x):
        print("Element is found at index:",count)
        break
else:
    print("Element is not found")
```

OUTPUT:



```
================================================
Enter the values:4 7 1 2 8
Value:2
Element is found at index: 3
```

RESULT: Thus the above program to implement linear search has been verified successfully.

Expt. No:
Date :

AIM: To write a program to implement binary search using iteration.

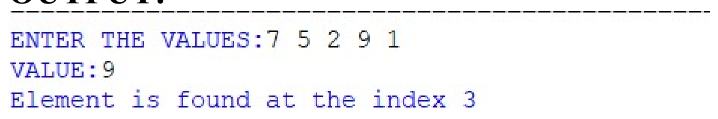
ALGORITHM:

STEP 1: Start
STEP 2: Using a user defined function and comparing the value of x with middle element,if it matches return the middle index.
STEP 3: If x is greater than the middle element, then high=mid-1
STEP 4:If x is lesser than the middle element, then low=mid+1
STEP 5:Stop

PROGRAM:

```
def bin_search(l,x):
        low=0
        high=len(l)-1
        mid=0
        while(low<=high):
                mid=(high+low)//2
                if l[mid]<x:
                        low=mid+1
                elif l[mid]>x:
                        high=mid-1
                else:
                        return mid
        return -1
l=list(map(int,input("ENTER THE VALUES:").split()))
x=int(input("VALUE:"))
a=bin_search(l,x)
if(a!=-1):
        print("Element is found at the index",a)
else:
        print("Element is not found")
```

OUTPUT:



```
ENTER THE VALUES:7 5 2 9 1
VALUE:9
Element is found at the index 3
```

RESULT: Thus the above program to implement binary search has been verified successfully.

Expt. No:
Date   :

AIM: To write a program for insertion sort.

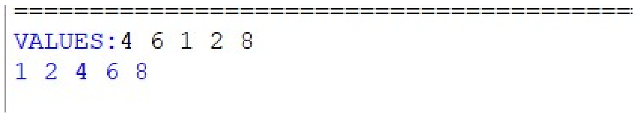
ALGORITHM:

STEP 1: Start
STEP 2: Assuming the first element as sorted.
STEP 3: Store the next element as key, and compare the key with all the elements in sorted list.
STEP 4: Shift the greater elements towards the right by checking the key element and the sorted list.
STEP 5:Insert the values in ascending order by comparing.
STEP 6:Continue until the array is sorted
STEP 7: Stop

PROGRAM:

```
def ins_sort(l):
        for I in range(1,len(l)):
                key=l[i]
                j=i-1
                while(j>=0 and key<l[j]):
                        l[j+1]=l[j]
                        j-=1
                l[j+1]=key
l=list(map(int,input("VALUES:").split()))
ins_sort(l)
print(' '.join(map(str,l)
```

OUTPUT:



```
=========================================
VALUES:4 6 1 2 8
1 2 4 6 8
```

RESULT: Thus the program for insertion sort has been verified successfully.

Expt. No:
Date :

AIM: To write a program to perform selection sort.

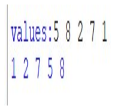
ALGORITHM:

STEP 1: Start
STEP 2:Take the minimum value from the list and place it in sorted list by swapping.
STEP 3: Repeat the process until all the elements are sorted.
STEP 4:Stop

PROGRAM:

```
l=list(map(int,input("Values:").split()))
for i in range(len(l)):
        min=i
        for j in range(i+1,len(l)):
                if(l[min]>l[j]):
                        min=j
                        l[i],l[min]=l[min],l[i]
print(' '.join(map(str,l)))
```

OUTPUT:



```
values:5 8 2 7 1
1 2 7 5 8
```

RESULT: Thus the above program for selection sort has been verified successfully.

Expt. No:
Date :

Aim : To write a program to perform merge sort.

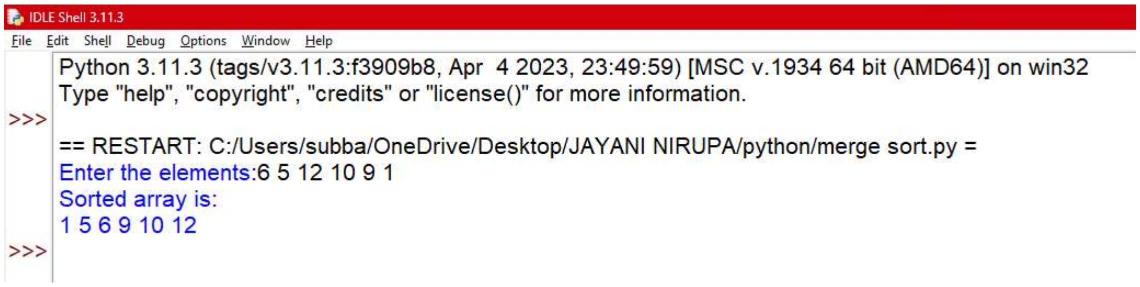
Algorithm

```
Step 1: Find the middle index of the array.
Middle = 1 + (last – first)/2
Step 2: Divide the array from the middle.
Step 3: Call merge sort for the first half of the array
MergeSort(array, first, middle)
Step 4: Call merge sort for the second half of the array.
MergeSort(array, middle+1, last)
Step 5: Merge the two sorted halves into a single sorted array.
```

PROGRAM:

```
# MergeSort in Python
def mergeSort(array):
    if len(array) > 1:
        r = len(array)//2
        L = array[:r]
        M = array[r:]
        mergeSort(L)
        mergeSort(M)
        i = j = k = 0
        while i < len(L) and j < len(M):
            if L[i] < M[j]:
                array[k] = L[i]
                i += 1
            else:
                array[k] = M[j]
                j += 1
            k += 1
        while i < len(L):
            array[k] = L[i]
            i += 1
            k += 1

        while j < len(M):
            array[k] = M[j]
            j += 1
            k += 1
def printList(array):
    for i in range(len(array)):
        print(array[i], end=" ")
    print()
if __name__ == '__main__':
    #array = [6, 5, 12, 10, 9, 1]
    array=list(map(int,input("Enter the elements:").split()))
    mergeSort(array)
    print("Sorted array is: ")
    printList(array)
```

OUTPUT:



```
Python 3.11.3 (tags/v3.11.3:f3909b8, Apr  4 2023, 23:49:59) [MSC v.1934 64 bit (AMD64)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>>
== RESTART: C:/Users/subba/OneDrive/Desktop/JAYANI NIRUPA/python/merge sort.py =
Enter the elements:6 5 12 10 9 1
Sorted array is:
1 5 6 9 10 12
>>>
```

RESULT:
The program has been compiled executed and output is verified successfully.

Expt. No:
Date :

Aim

To write a program to implement QUICK Sort

Algorithm

Step 1 - Consider the first element of the list as pivot (i.e., Element at first position in the list).

Step 2 - Define two variables i and j. Set i and j to first and last elements of the list respectively.

Step 3 - Increment i until list[i] > pivot then stop.

Step 4 - Decrement j until list[j] < pivot then stop.

Step 5 - If i < j then exchange list[i] and list[j].

Step 6 - Repeat steps 3,4 & 5 until i > j.

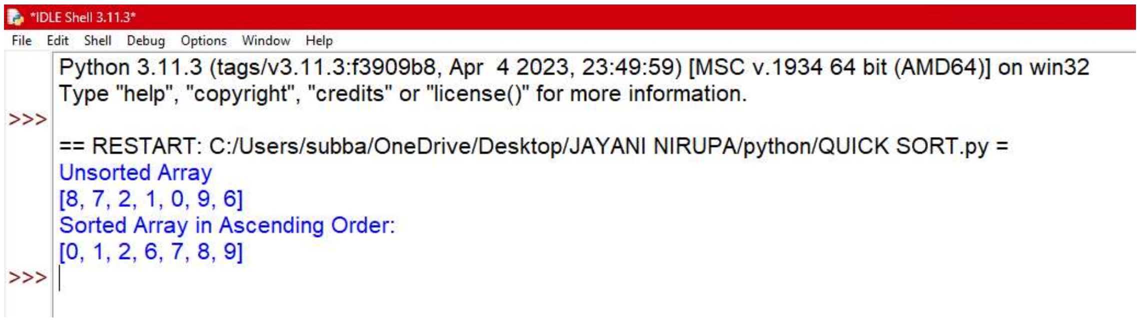
Step 7 - Exchange the pivot element with list[j] element.

PROGRAM:

```
# Quick sort in Python
def partition(array, low, high):
  pivot = array[high]
  i = low - 1
  for j in range(low, high):
    if array[j] <= pivot:
      i = i + 1
      (array[i], array[j]) = (array[j], array[i])
  (array[i + 1], array[high]) = (array[high], array[i + 1])
  return i + 1
```

```
# function to perform quicksort
def quickSort(array, low, high):
  if low < high:
    pi = partition(array, low, high)
    quickSort(array, low, pi - 1)
    quickSort(array, pi + 1, high)


data = [8, 7, 2, 1, 0, 9, 6]
print("Unsorted Array")
print(data)
size = len(data)
quickSort(data, 0, size - 1)
print('Sorted Array in Ascending Order:')
print(data)
```

OUTPUT:



```
Python 3.11.3 (tags/v3.11.3:f3909b8, Apr  4 2023, 23:49:59) [MSC v.1934 64 bit (AMD64)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>>
== RESTART: C:/Users/subba/OneDrive/Desktop/JAYANI NIRUPA/python/QUICK SORT.py =
Unsorted Array
[8, 7, 2, 1, 0, 9, 6]
Sorted Array in Ascending Order:
[0, 1, 2, 6, 7, 8, 9]
>>>
```

RESULT:
The program has been compiled executed and output is verified successfully.

Expt. No:
Date :

Aim

To write a Python program to print Prime Numbers between a range of numbers

Algorithm

Step 1: Loop through all the elements in the given range.

Step 2: Check for each number if it has any factor between 1 and itself.

Step 3: If yes, then the number is not prime, and it will move to the next number.

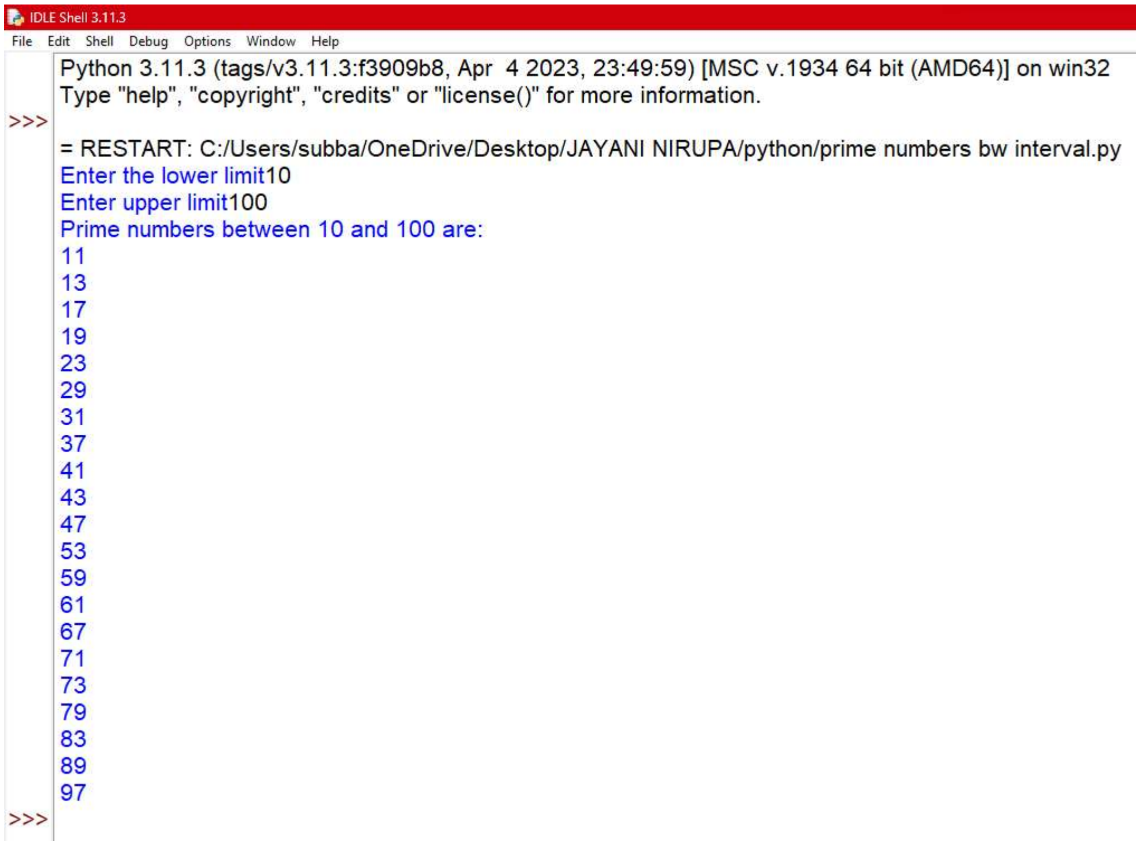
Step 4: If no, it is the prime number, and the program will print it and check for the next number.

Step 5: The loop will break when it is reached to the upper value.

PROGRAM:

```
# Python program to print Prime Numbers between a range of numbers

lower = int(input("Enter the lower limit"))
upper = int(input("Enter upper limit"))

print("Prime numbers between", lower, "and", upper, "are:")

for num in range(lower, upper + 1):
   if num > 1:
       for i in range(2, num):
           if (num % i) == 0:
               break
       else:
           print(num)
```

OUTPUT:



```
    Python 3.11.3 (tags/v3.11.3:f3909b8, Apr  4 2023, 23:49:59) [MSC v.1934 64 bit (AMD64)] on win32
    Type "help", "copyright", "credits" or "license()" for more information.
>>> 
    = RESTART: C:/Users/subba/OneDrive/Desktop/JAYANI NIRUPA/python/prime numbers bw interval.py
    Enter the lower limit10
    Enter upper limit100
    Prime numbers between 10 and 100 are:
    11
    13
    17
    19
    23
    29
    31
    37
    41
    43
    47
    53
    59
    61
    67
    71
    73
    79
    83
    89
    97
>>> 
```

RESULT:

The program has been compiled executed and output is verified successfully.

Expt. No:
Date :

Aim

To write a Python program to multiply two matrices

Algorithm

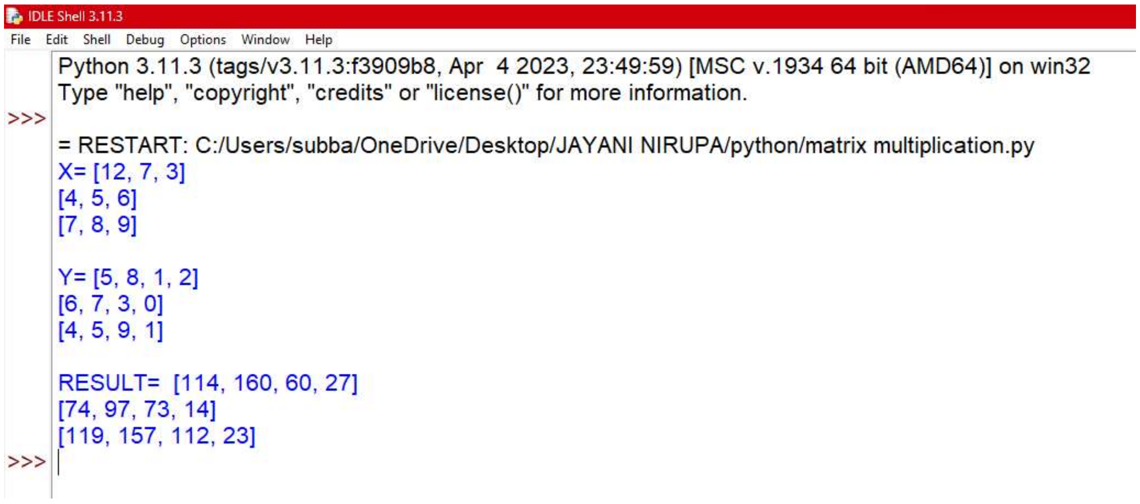
1. Store the matrix dimensions in different variables
2. Check if the matrices are multiplication compatible. If no, terminate the program, otherwise continue.
3. Iterate over the rows of matrix A using an index-variable i
4. Inside the first loop, iterate over the columns of matrix B using the index-variable j
5. Now initialize a variable curr_val to 0
6. Create another loop iterating over the column dimension of A (or equivalently the row dimension of B) using a variable k
7. For each iteration of the innermost loop, add the value of A[i][k]×B[k][j] to the variable curr_val
8. After each iteration of the innermost loop, assign the value of curr_val to C[i][j]

PROGRAM:

```
# Program to multiply two matrices

# 3x3 matrix

X = [[12,7,3],

   [4 ,5,6],

   [7 ,8,9]]

print("X=",end=" ")

for r1 in X:

  print(r1)

print(" ")
```

```
# 3x4 matrix
Y = [[5,8,1,2],
    [6,7,3,0],
    [4,5,9,1]]
print("Y=",end=" ")
for r2 in Y:
   print(r2)
print(" ")
# result is 3x4
result = [[0,0,0,0],
         [0,0,0,0],
         [0,0,0,0]]
# iterate through rows of X
for i in range(len(X)):
   # iterate through columns of Y
   for j in range(len(Y[0])):
       # iterate through rows of Y
       for k in range(len(Y)):
           result[i][j] += X[i][k] * Y[k][j]
print("RESULT= ",end=" ")
for r in result:
   print(r)
```

OUTPUT:



```
Python 3.11.3 (tags/v3.11.3:f3909b8, Apr  4 2023, 23:49:59) [MSC v.1934 64 bit (AMD64)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>>
= RESTART: C:/Users/subba/OneDrive/Desktop/JAYANI NIRUPA/python/matrix multiplication.py
X= [12, 7, 3]
[4, 5, 6]
[7, 8, 9]

Y= [5, 8, 1, 2]
[6, 7, 3, 0]
[4, 5, 9, 1]

RESULT=  [114, 160, 60, 27]
[74, 97, 73, 14]
[119, 157, 112, 23]
>>>
```

RESULT:
The program has been compiled executed and output is verified successfully.

Expt. No:
Date :

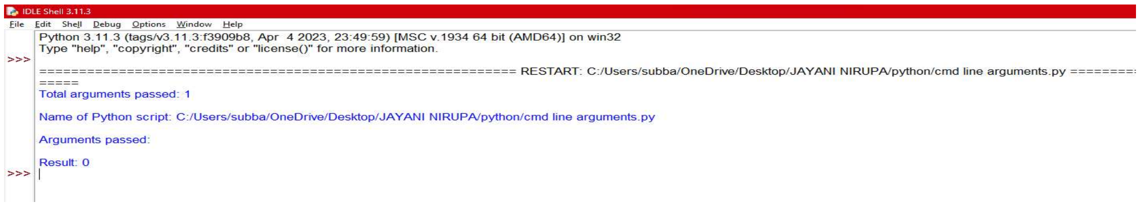
Aim

To write a python program to implement command line arguments

PROGRAM:

```
import sys
# total arguments
n = len(sys.argv)
print("Total arguments passed:", n)
# Arguments passed
print("\nName of Python script:", sys.argv[0])
print("\nArguments passed:", end = " ")
for i in range(1, n):
        print(sys.argv[i], end = " ")
# Addition of numbers
Sum = 0
# Using argparse module
for i in range(1, n):
        Sum += int(sys.argv[i])
print("\n\nResult:", Sum)
```

OUTPUT:



```
Python 3.11.3 (tags/v3.11.3:f3909b8, Apr  4 2023, 23:49:59) [MSC v.1934 64 bit (AMD64)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>>
============================================================== RESTART: C:/Users/subba/OneDrive/Desktop/JAYANI NIRUPA/python/cmd line arguments.py ========
=====
Total arguments passed: 1

Name of Python script: C:/Users/subba/OneDrive/Desktop/JAYANI NIRUPA/python/cmd line arguments.py

Arguments passed:

Result: 0
>>>
```

RESULT:
The program has been compiled executed and output is verified successfully.

Expt. No:
Date :

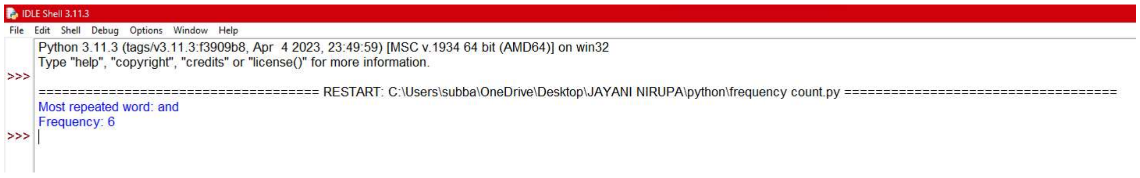
Aim

To write a python program to find the most repeated word in a text file

PROGRAM:

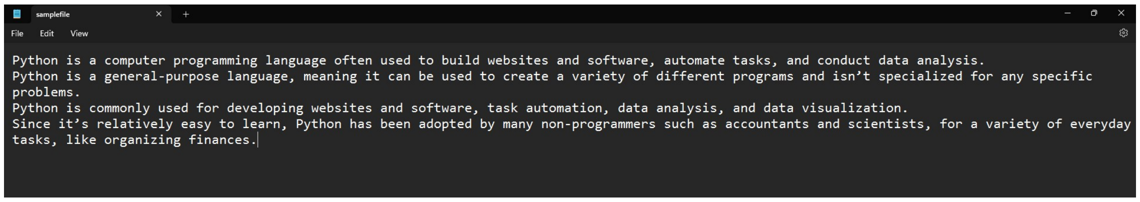
```
# Python program to find the most repeated word in a text file
# reading mode.
file = open("samplefile.txt","r")
frequent_word = ""
frequency = 0
words = []

# Traversing file line by line
for line in file:

    # splits each line into
    # words and removing spaces
    # and punctuations from the input
    line_word = line.lower().replace(',','').replace('.','').split(" ");

    # Adding them to list words
    for w in line_word:
        words.append(w);

# Finding the max occurred word
for i in range(0, len(words)):
    # Declaring count
    count = 1;

    # Count each word in the file
    for j in range(i+1, len(words)):
        if(words[i] == words[j]):
            count = count + 1;

    # If the count value is more
    # than highest frequency then
    if(count > frequency):
        frequency = count;
        frequent_word = words[i];

print("Most repeated word: " + frequent_word)
print("Frequency: " + str(frequency))
file.close();
```

OUTPUT:



```
Python 3.11.3 (tags/v3.11.3:f3909b8, Apr  4 2023, 23:49:59) [MSC v.1934 64 bit (AMD64)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>>
================================== RESTART: C:\Users\subba\OneDrive\Desktop\JAYANI NIRUPA\python\frequency count.py ==================================
Most repeated word: and
Frequency: 6
>>>
```

Text File:



```
Python is a computer programming language often used to build websites and software, automate tasks, and conduct data analysis.
Python is a general-purpose language, meaning it can be used to create a variety of different programs and isn't specialized for any specific
problems.
Python is commonly used for developing websites and software, task automation, data analysis, and data visualization.
Since it's relatively easy to learn, Python has been adopted by many non-programmers such as accountants and scientists, for a variety of everyday
tasks, like organizing finances.
```

RESULT:
The program has been compiled executed and output is verified successfully.

Expt. No:
Date :

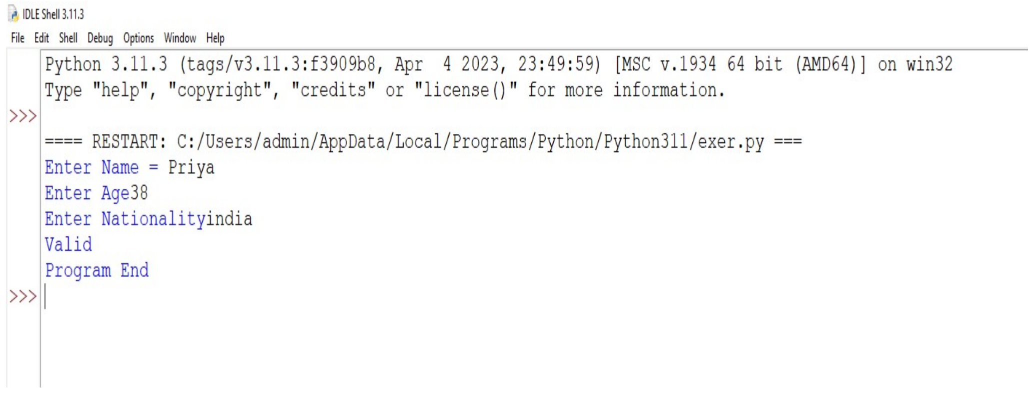
Aim

To write a python program to demonstrate user defined exception handling for License Registration Process

PROGRAM:

```
class LicenseException(Exception):
        message = None
        def _init_(obj,m):
                obj.message = m

try:
        name = input("Enter Name = ")
        age = int(input("Enter Age"))
        nation =input("Enter Nationality")
        if(age>=18 and age<=60 and nation=="india"):
                print("Valid")
        elif(age<=17 and age >=61):
                error = LicenseException("Invalid Age")
                raise error
        else:
                        error = LicenseException("Invalid Nation")
                        raise error


except LicenseException:
                        print(error.message)

finally:
                        print('Program End')
```

OUTPUT:



```
Python 3.11.3 (tags/v3.11.3:f3909b8, Apr  4 2023, 23:49:59) [MSC v.1934 64 bit (AMD64)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>>
==== RESTART: C:/Users/admin/AppData/Local/Programs/Python/Python311/exer.py ===
Enter Name = Priya
Enter Age38
Enter Nationalityindia
Valid
Program End
>>>
```

RESULT:
The program has been compiled executed and output is verified successfully.

Expt. No:
Date :

Aim

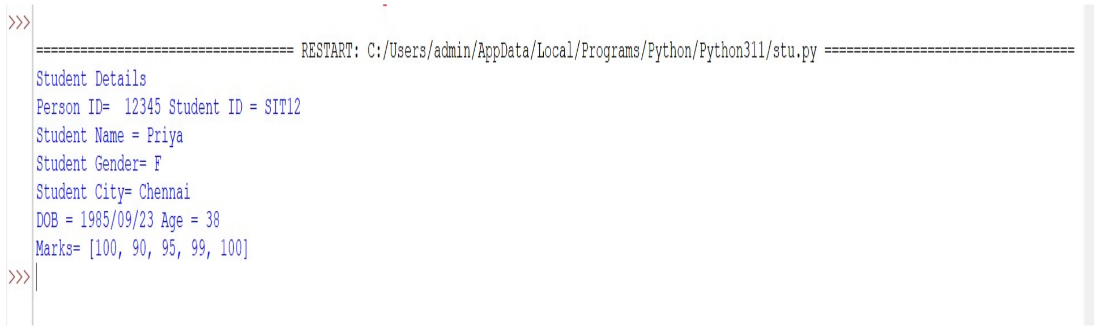
To write a Python program to implement class and object - Student Class

PROGRAM:

```
class Person:
   PID = None
   PName = None
   PGender = None
   PCity = None
   PDOB = None
   def __init__(obj,idd,name,gender,city,dob):
      obj.PID = idd
      obj.PName = name
      obj.PGender = gender
      obj.PCity = city
      obj.PDOB = dob

   def calcAge(obj):
      return 2023 - int(obj.PDOB[0:4])

class Student(Person):
   SID = None
   SMarks = None
   SAge = None
   def __init__(obj,idd,name,gender,city,dob,sid,marks):
      obj.PID = idd
      obj.PName = name
      obj.PGender = gender
      obj.PCity = city
      obj.PDOB = dob
      obj.SAge = obj.calcAge()
      obj.SID = sid
      obj.SMarks = marks
   def printStudent(obj):
      print("Student Details")
      print("Person ID= ",obj.PID,"Student ID =",obj.SID)
```

```
        print("Student Name =",obj.PName)
        print("Student Gender=",obj.PGender)
        print("Student City=",obj.PCity)
        print("DOB =",obj.PDOB,"Age =",obj.SAge)
        print("Marks=",obj.SMarks)

s1 =
Student(12345,"Priya","F","Chennai","1985/09/23","SIT12",eval("[100,9
0,95,99,100]"))
s1.printStudent()
```

OUTPUT:



```
>>>
================================ RESTART: C:/Users/admin/AppData/Local/Programs/Python/Python311/stu.py ================================
Student Details
Person ID=  12345 Student ID = SIT12
Student Name = Priya
Student Gender= F
Student City= Chennai
DOB = 1985/09/23 Age = 38
Marks= [100, 90, 95, 99, 100]
>>>
```

RESULT:

The program has been compiled executed and output is verified successfully.

Expt. No:
Date :

Aim

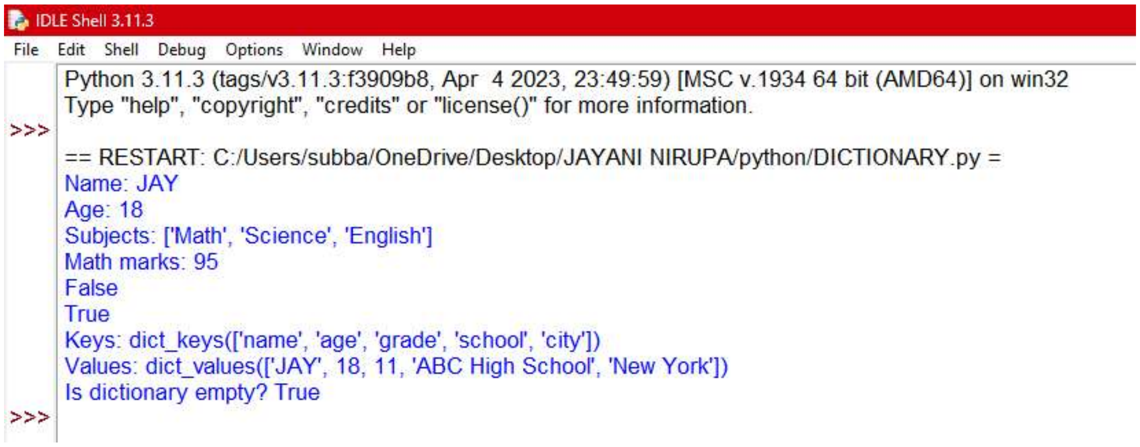
To write a Python program to illustrate various dictionary functions

PROGRAM:

```
# Create a dictionary
student = {
    "name": "JAY",
    "age": 18,
    "grade": 12,
    "subjects": ["Math", "Science", "English"],
    "marks": {
        "Math": 95,
        "Science": 88,
        "English": 92
    }
}

# Accessing dictionary elements
print("Name:", student["name"])
print("Age:", student.get("age"))
print("Subjects:", student["subjects"])
print("Math marks:", student["marks"]["Math"])
# Modifying dictionary elements
student["age"] = 18
student["grade"] = 11
student["marks"]["Science"] = 90
```

```
# Adding new key-value pairs
student["school"] = "ABC High School"
student["city"] = "New York"
# Removing key-value pairs
del student["subjects"]
student.pop("marks")
# Checking existence of key
print("City" in student)
print("grade" in student)
# Getting keys and values
print("Keys:", student.keys())
print("Values:", student.values())
# Clearing the dictionary
student.clear()
# Checking if the dictionary is empty
print("Is dictionary empty?", len(student) == 0)
```

OUTPUT:



```
Python 3.11.3 (tags/v3.11.3:f3909b8, Apr  4 2023, 23:49:59) [MSC v.1934 64 bit (AMD64)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>>
== RESTART: C:/Users/subba/OneDrive/Desktop/JAYANI NIRUPA/python/DICTIONARY.py =
Name: JAY
Age: 18
Subjects: ['Math', 'Science', 'English']
Math marks: 95
False
True
Keys: dict_keys(['name', 'age', 'grade', 'school', 'city'])
Values: dict_values(['JAY', 18, 11, 'ABC High School', 'New York'])
Is dictionary empty? True
>>>
```

RESULT:
The program has been compiled executed and output is verified successfully.

Expt. No:
Date   :

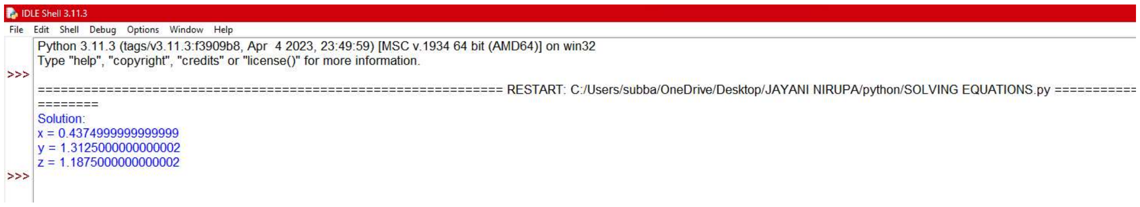
Aim

To write a Python program to solve linear system of equations

PROGRAM:

```
import numpy as np
# Coefficient matrix
A = np.array([[2, 1, -1],
              [4, -1, 3],
              [1, 3, -2]])

# Right-hand side vector
b = np.array([1, 4, 2])
# Solve the linear system
x = np.linalg.solve(A, b)
# Print the solution
print("Solution:")
print("x =", x[0])
print("y =", x[1])
print("z =", x[2])
```

OUTPUT:



```
Python 3.11.3 (tags/v3.11.3:f3909b8, Apr  4 2023, 23:49:59) [MSC v.1934 64 bit (AMD64)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>>
============================================================== RESTART: C:/Users/subba/OneDrive/Desktop/JAYANI NIRUPA/python/SOLVING EQUATIONS.py ==========
========
Solution:
x = 0.4374999999999999
y = 1.3125000000000002
z = 1.1875000000000002
>>>
```

RESULT:
The program has been compiled executed and output is verified successfully.

Expt. No:
Date :

Aim

To write a Python program to plot the following graphs

a. Histogram

b. Scatter Plot

c. Simple Plot

d. Box Plot

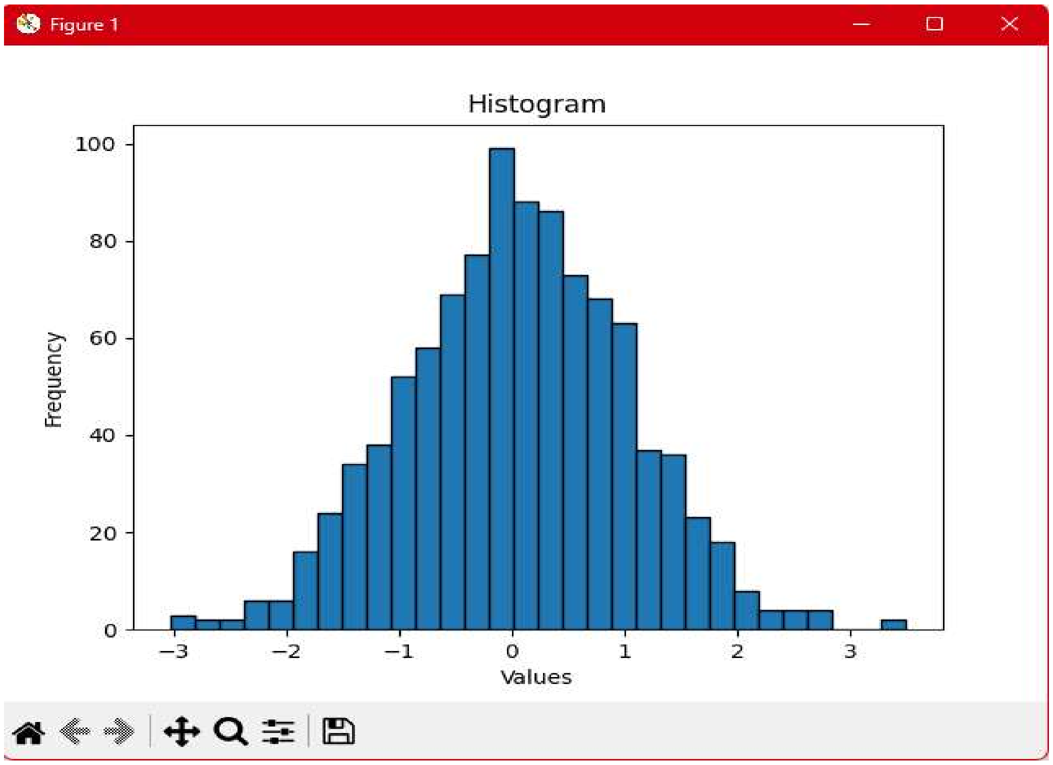
e. Pie Chart

PROGRAM:

a)HISTOGRAM

```
 import matplotlib.pyplot as plt
import numpy as np
# Generate random data
data = np.random.randn(1000)
# Create histogram
plt.hist(data, bins=30, edgecolor='black')
# Set labels and title
plt.xlabel('Values')
plt.ylabel('Frequency')
plt.title('Histogram')
# Display the plot
plt.show()
```

OUTPUT:



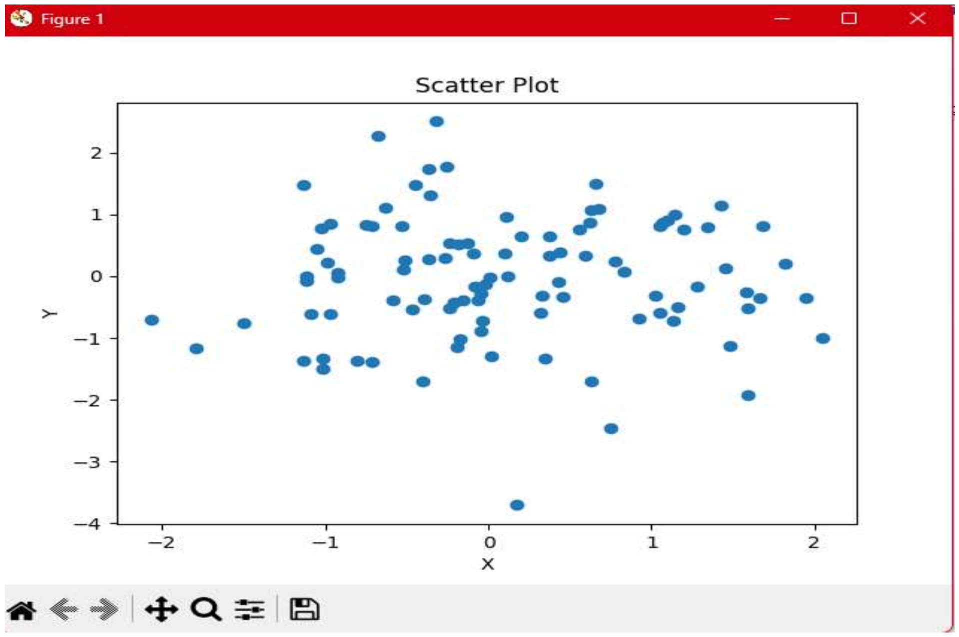
b) SCATTER PLOT

PROGRAM:

```
import matplotlib.pyplot as plt

import numpy as np

# Generate random data

x = np.random.randn(100)

y = np.random.randn(100)

# Create scatter plot

plt.scatter(x, y)

# Set labels and title

plt.xlabel('X')

plt.ylabel('Y')

plt.title('Scatter Plot')

# Display the plot
```

```
plt.show()
```

OUTPUT:

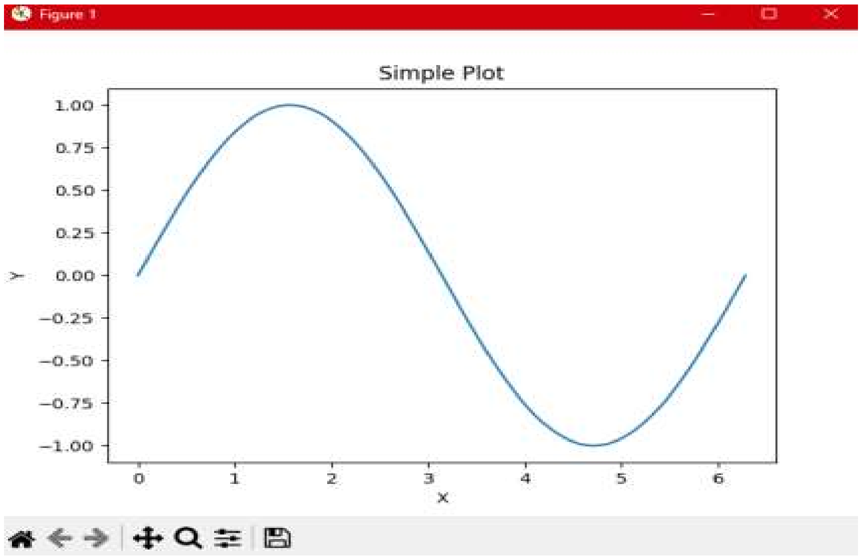


c)SIMPLE PLOT

PROGRAM:

```
import matplotlib.pyplot as plt
import numpy as np
# Generate data
x = np.linspace(0, 2*np.pi, 100)
y = np.sin(x)
# Create plot
plt.plot(x, y)
# Set labels and title
plt.xlabel('X')
plt.ylabel('Y')
plt.title('Simple Plot')
# Display the plot
plt.show()
```

OUTPUT:

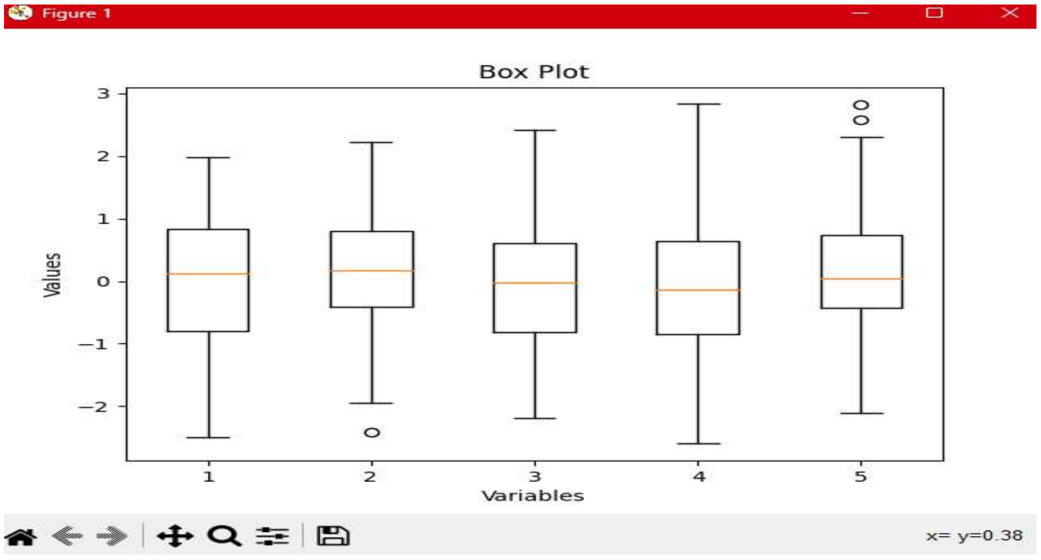


d)BOX PLOT

PROGRAM:

```
import matplotlib.pyplot as plt
import numpy as np
# Generate random data
data = np.random.randn(100, 5)
# Create box plot
plt.boxplot(data)
# Set labels and title
plt.xlabel('Variables')
plt.ylabel('Values')
plt.title('Box Plot')
# Display the plot
plt.show()
```

OUTPUT:

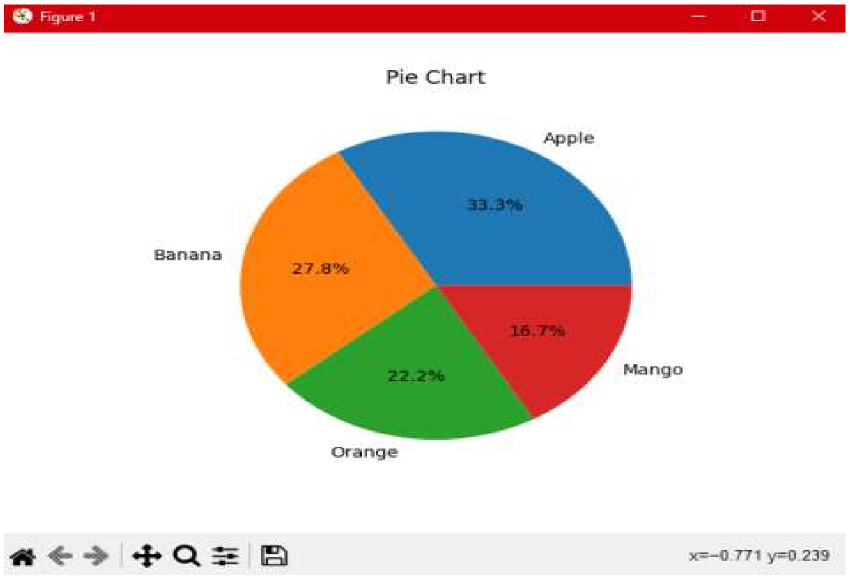


e) PIE CHART

PROGRAM:

```
import matplotlib.pyplot as plt
# Data for the pie chart
labels = ['Apple', 'Banana', 'Orange', 'Mango']
sizes = [30, 25, 20, 15]
# Create pie chart
plt.pie(sizes, labels=labels, autopct='%1.1f%%')
# Set title
plt.title('Pie Chart')
# Display the plot
plt.show()
```

OUTPUT:



RESULT:
The program has been compiled executed and output is verified successfully.

Expt. No:
Date :

AIM:

To write a Python program to open a CSV file using Pandas

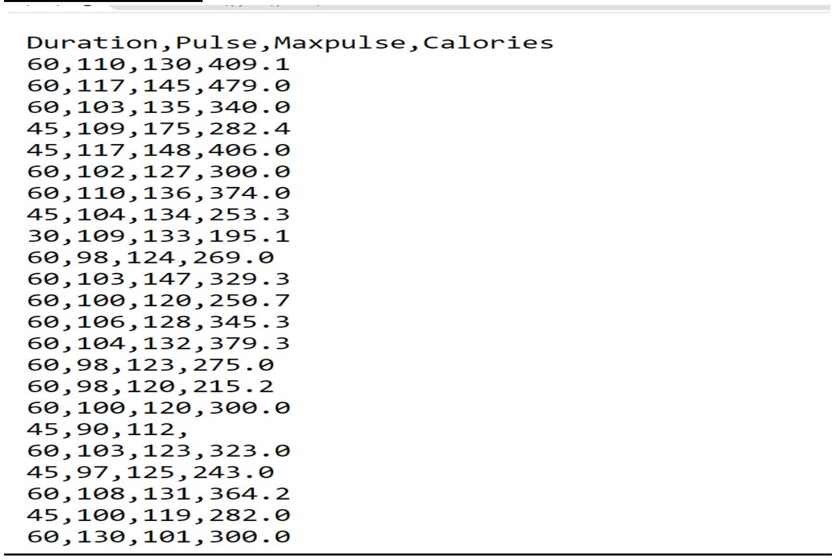
- Read CSV Files
- A simple way to store big data sets is to use CSV files (comma separated files).

- CSV files contains plain text and is a well know format that can be read by everyone including Pandas.

- In our examples we will be using a CSV file called 'data.csv'.

- Download data.csv. or Open data.csv

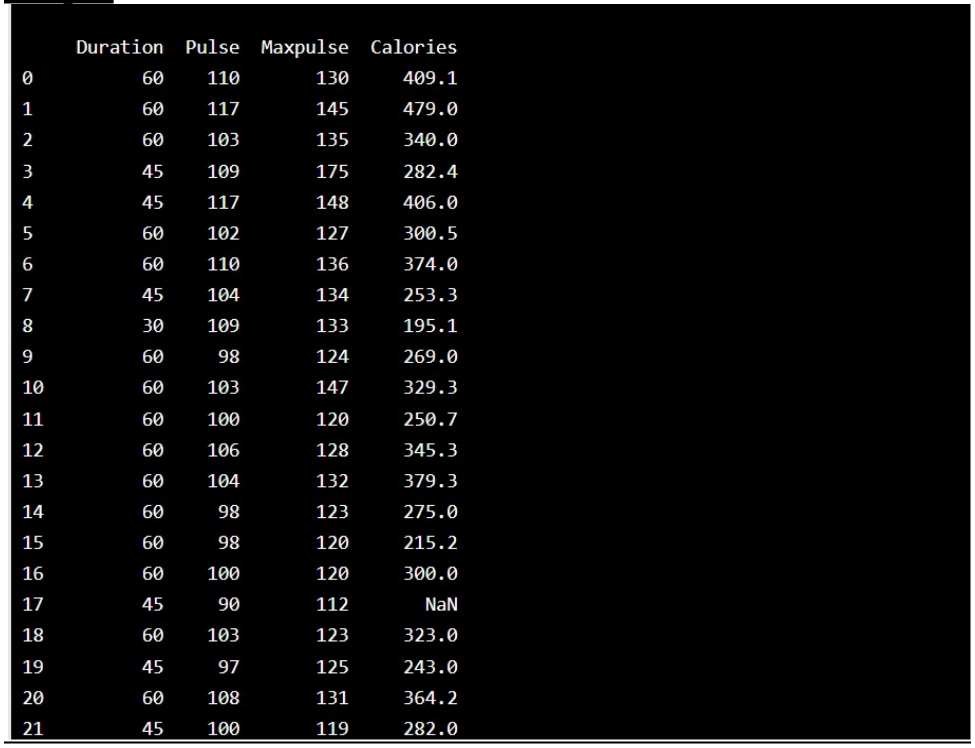
data.csv - File



```
Duration,Pulse,Maxpulse,Calories
60,110,130,409.1
60,117,145,479.0
60,103,135,340.0
45,109,175,282.4
45,117,148,406.0
60,102,127,300.0
60,110,136,374.0
45,104,134,253.3
30,109,133,195.1
60,98,124,269.0
60,103,147,329.3
60,100,120,250.7
60,106,128,345.3
60,104,132,379.3
60,98,123,275.0
60,98,120,215.2
60,100,120,300.0
45,90,112,
60,103,123,323.0
45,97,125,243.0
60,108,131,364.2
45,100,119,282.0
60,130,101,300.0
```

```
import pandas as pd

df = pd.read_csv('data.csv')

print(df.to_string())
```

Output



```
    Duration  Pulse  Maxpulse  Calories
0         60    110       130     409.1
1         60    117       145     479.0
2         60    103       135     340.0
3         45    109       175     282.4
4         45    117       148     406.0
5         60    102       127     300.5
6         60    110       136     374.0
7         45    104       134     253.3
8         30    109       133     195.1
9         60     98       124     269.0
10        60    103       147     329.3
11        60    100       120     250.7
12        60    106       128     345.3
13        60    104       132     379.3
14        60     98       123     275.0
15        60     98       120     215.2
16        60    100       120     300.0
17        45     90       112       NaN
18        60    103       123     323.0
19        45     97       125     243.0
20        60    108       131     364.2
21        45    100       119     282.0
```

RESULT:
The program has been compiled executed and output is verified successfully

Content Beyond Syllabus

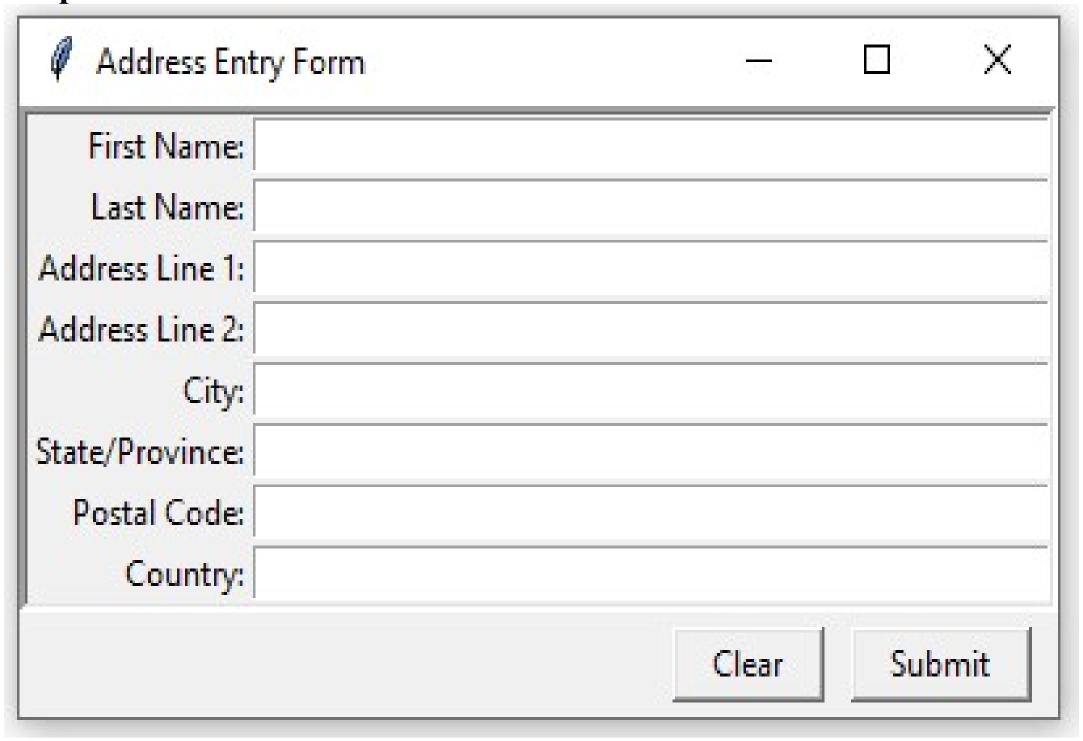
Tkinter Example Program - "Address Entry Form"

```
import tkinter as tk

# Create a new window with the title "Address Entry Form"
window = tk.Tk()
window.title("Address Entry Form")

# Create a new frame `frm_form` to contain the Label
# and Entry widgets for entering address information
frm_form = tk.Frame(relief=tk.SUNKEN, borderwidth=3)
# Pack the frame into the window
frm_form.pack()

# Create the Label and Entry widgets for "First Name"
lbl_first_name = tk.Label(master=frm_form, text="First Name:")
ent_first_name = tk.Entry(master=frm_form, width=50)
# Use the grid geometry manager to place the Label and
# Entry widgets in the first and second columns of the
# first row of the grid
lbl_first_name.grid(row=0, column=0, sticky="e")
ent_first_name.grid(row=0, column=1)

# Create the Label and Entry widgets for "Last Name"
lbl_last_name = tk.Label(master=frm_form, text="Last Name:")
ent_last_name = tk.Entry(master=frm_form, width=50)
# Place the widgets in the second row of the grid
lbl_last_name.grid(row=1, column=0, sticky="e")
ent_last_name.grid(row=1, column=1)

# Create the Label and Entry widgets for "Address Line 1"
lbl_address1 = tk.Label(master=frm_form, text="Address Line 1:")
ent_address1 = tk.Entry(master=frm_form, width=50)
# Place the widgets in the third row of the grid
lbl_address1.grid(row=2, column=0, sticky="e")
ent_address1.grid(row=2, column=1)

# Create the Label and Entry widgets for "Address Line 2"
lbl_address2 = tk.Label(master=frm_form, text="Address Line 2:")
ent_address2 = tk.Entry(master=frm_form, width=50)
# Place the widgets in the fourth row of the grid
lbl_address2.grid(row=3, column=0, sticky=tk.E)
ent_address2.grid(row=3, column=1)
```

```
# Create the Label and Entry widgets for "City"
lbl_city = tk.Label(master=frm_form, text="City:")
ent_city = tk.Entry(master=frm_form, width=50)
# Place the widgets in the fifth row of the grid
lbl_city.grid(row=4, column=0, sticky=tk.E)
ent_city.grid(row=4, column=1)

# Create the Label and Entry widgets for "State/Province"
lbl_state = tk.Label(master=frm_form, text="State/Province:")
ent_state = tk.Entry(master=frm_form, width=50)
# Place the widgets in the sixth row of the grid
lbl_state.grid(row=5, column=0, sticky=tk.E)
ent_state.grid(row=5, column=1)

# Create the Label and Entry widgets for "Postal Code"
lbl_postal_code = tk.Label(master=frm_form, text="Postal Code:")
ent_postal_code = tk.Entry(master=frm_form, width=50)
# Place the widgets in the seventh row of the grid
lbl_postal_code.grid(row=6, column=0, sticky=tk.E)
ent_postal_code.grid(row=6, column=1)

# Create the Label and Entry widgets for "Country"
lbl_country = tk.Label(master=frm_form, text="Country:")
ent_country = tk.Entry(master=frm_form, width=50)
# Place the widgets in the eight row of the grid
lbl_country.grid(row=7, column=0, sticky=tk.E)
ent_country.grid(row=7, column=1)

# Create a new frame `frm_buttons` to contain the
# Submit and Clear buttons. This frame fills the
# whole window in the horizontal direction and has
# 5 pixels of horizontal and vertical padding.
frm_buttons = tk.Frame()
frm_buttons.pack(fill=tk.X, ipadx=5, ipady=5)

# Create the "Submit" button and pack it to the
# right side of `frm_buttons`
btn_submit = tk.Button(master=frm_buttons, text="Submit")
btn_submit.pack(side=tk.RIGHT, padx=10, ipadx=10)

# Create the "Clear" button and pack it to the
# right side of `frm_buttons`
```

```
btn_clear = tk.Button(master=frm_buttons, text="Clear")
btn_clear.pack(side=tk.RIGHT, ipadx=10)

# Start the application
window.mainloop()
```

Output


Address Entry Form
First Name:
Last Name:
Address Line 1:
Address Line 2:
City:
State/Province:
Postal Code:
Country:
Clear
Submit