

1.A. ELECTRICITY BILL

AIM:

To write a python program to calculate electricity bill

ALGORITHM

Step1: Start

Step2: Read units

Step3: check if(units<=100) then

 payAmount=units*1.5

 fixedcharge=25.00

 elif(units<=200) then

 payAmount=(100*1.5)+(units*100)*2.5

 fixedcharge=50.00

 elif (units<=300) then

 payAmount=(100*1.5)+(200-100)*2.5+(units-200)*4

 fixedcharge=100.00

 elif(units<=350) then

 payAmount=(100*1.5)+(200-100)*2.5+(300-200)*4+(units-300)*5

 fixedcharge=100.00

 else

 payAmount=0

 fixedcharge=1500.00

step4 : calculate total=payAmount+fixedcharge

Step 5: print the electricity bill

Step 6: stop

PROGRAM

```
units=int(input("please enter the number of units you consumed in a month"))
```

```
if(units<=100):
```

```
    payAmount=units*1.5
```

```
    fixedcharge=25.00
```

```
elif(units<=200):
```

```
    payAmount=(100*1.5)+(units-100)*2.5
```

```
    fixedcharge=50.00
```

```
elif(units<=300):
```

```
    payAmount=(100*1.5)+(200-100)*2.5+(units-200)*4
```

```
    fixedcharge=75.00
```

```
elif(units<=350):
```

```
    payAmount=(100*1.5)+(200-100)*2.5+(300-200)*4+(units-300)*5
```

```
    fixedcharge=100.00
```

```
else:
```

```
    payAmount=0
```

```
    fixedcharge=1500.00
```

```
Total=payAmount+fixedcharge
```

```
print("\nElectricity bill",Total)
```

RESULT:

Thus the Program was executed successfully and the output was verified.

OUTPUT:

```
please enter the number of units you consumed in a month300
```

```
Electricity bill 875.0
```

1.B. RETAIL SHOP BILLING

AIM:

To write a python program to calculate Retail shop bill

ALGORITHM:

Step 1: Start the program

Step 2 : Define a function to calculate the bill amount.

Step 3: Read items_list, reqd_list,price_list,reqd_quantity.

Step 4: Call the function to calculate bill amount.

Step 5 : Print the bill amount.

Step 6: Stop the program.

PROGRAM:

```
def calculate_bill_amount(items_list, price_list, reqd_items, reqd_quantity):  
    bill_amount=0  
    i=0  
    total=0  
    leng=len(reqd_items)  
    check = all(item in items_list for item in reqd_items)  
    if check is True:  
        while(i<leng):  
            inty=items_list.index(reqd_items[i])  
            total=total+ price_list[inty]*reqd_quantity[i]  
            bill_amount=total  
            i=i+1  
        else:  
            bill_amount=total  
    return bill_amount  
items_list=["rice","oil","sugar","soap","paste"]  
price_list=[50,200,40,30,35]  
reqd_items=["rice","oil"]  
reqd_quantity=[5,2]  
bill_amount=calculate_bill_amount(items_list, price_list, reqd_items, reqd_quantity)  
print("Total bill amount:", bill_amount)
```

RESULT:

Thus the Program was executed successfully and the output was verified.

OUTPUT:

Total bill amount: 650

1.C.SINE SERIES

AIM:

To write a python program to calculate Sine series

$$\sin x = x - x^3/3! + x^5/5! - x^7/7! + x^9/9! \dots\dots$$

ALGORITHM:

Step 1: start
Step 2: read the value of x and n
Step 3: calculate the sum of sine series
 Sign=-1
 Fact=i=1
 Sum=0
Step 4: check while i<=n then
 P=1
 Fact=1
 For j in range (1,I +1)then
 P=p*x
 fact=fact*j
 sign=-1*sign
Step 5: calculate sum=sum+sign*p/fact
 I=i+2
Step 6: print('sin(,'x,')=',sum)
Step 7:stop

PROGRAM:

```
x = int(input("Enter the value of x: "))
n = int(input("Enter the value of n: "))
sign = -1
fact = i = 1
sum = 0
while i<=n:
    p = 1
    fact = 1
    for j in range(1,i+1):
        p = p*x
        fact = fact*j
    sign = -1*sign
    sum = sum + sign* p/fact
    i = i+2
print("sin(",x,") =",sum)
```

RESULT:

Thus the Program was executed successfully and the output was verified.

OUTPUT:

```
Enter the value of x: 5
Enter the value of n: 4
sin( 5 ) = -15.833333333333332
```

1.D. WEIGHT OF BIKE

AIM:

To write a python program along with flowchart to calculate Weight of bike

ALGORITHM:

Step1: start

Step2: read the values of x,wf,wr,wb

Step3: calculate the weight of bike

$$X = \frac{wr * wb}{wf + wr}$$

Step4: print weight of bike

Step5: stop

PROGRAM:

```
Wf=10#front weight
```

```
Wr=40# rear weight
```

```
WB=50#wheelbase
```

```
X =Wr*WB/ Wf+Wr
```

```
print("Weight of bike",X)
```

RESULT:

Thus the Program was executed successfully and the output was verified.

OUTPUT:

Weight of bike 240.0

1.E.WEIGHT OF STEEL BAR

AIM:

To write a python program along with flowchart to calculate Weight of steel bar

ALGORITHM:

Step1 :start

Step2: read the values of D and L

Step 3: calculate weight of steel bar

$$W=(D^{**2})*L)/162$$

Step4: print Weight of steel bar

Step5 : stop

PROGRAM:

```
D=10
```

```
L=100
```

```
W=(D**2)*L)/162
```

```
print ("Weight of steel bar", W)
```

RESULT:

Thus the Program was executed successfully and the output was verified.

OUTPUT:

Weight of steel bar 61.72839506172839

1.F. COMPUTE ELECTRICAL CURRENT IN THREE PHASE AC CIRCUIT

AIM:

To write a python program to calculate Sine series

ALGORITHM:

Step 1: start

Step 2: read the value of R , X_L , V_L , f

Step 3: calculate the line current

$$V_{Ph} = V_L / \sqrt{3}$$

$$Z_{Ph} = \sqrt{(R^2) + (X_L^2)}$$

$$I_{Ph} = V_{Ph} / Z_{Ph}$$

$$I_L = I_{Ph}$$

Step 4: print the line current in A is , round(I_L ,2)

Step 5: calculate the power factor

$$Pf = \cos(\phi) = R_{Ph} / Z_{Ph}$$

$$R_{Ph} = R$$

$$\phi = \arccos(R_{Ph} / Z_{Ph})$$

Step 6: print the power factor is : Pf 'degree lag.

Step 7: calculate the power supplied

$$P = \sqrt{3} * V_L * I_L * \cos(\phi)$$

Step 8: print the power supplied in W is;P

Step 9: stop

PROGRAM:

```
from math import cos,acos,sqrt
R = 20. ;# in ohm
X_L = 15. ;# in ohm
V_L = 400. ;# in V
f = 50. ;# in Hz
#calculations
V_Ph = V_L/sqrt(3);# in V
Z_Ph = sqrt( (R**2) + (X_L**2) );# in ohm
I_Ph = V_Ph/Z_Ph;# in A
I_L = I_Ph;# in A
print ("The line current in A is",round(I_L,2))
# pf = cos(phi) = R_Ph/Z_Ph;
R_Ph = R;# in ohm
phi= acos(R_Ph/Z_Ph);
# Power factor
pf= cos(phi);# in radians
print ("The power factor is : ",pf,"degrees lag.")
P = sqrt(3)*V_L*I_L*cos(phi);# in W
print ("The power supplied in W is",P)
```

RESULT:

Thus the Program was executed successfully and the output was verified.

OUTPUT:

The line current in A is 9.24

The power factor is : 0.8 degrees lag.

The power supplied in W is 5120.000000000001

2.A. CIRCULATE THE VALUES OF N VARIABLES

AIM:

To write a Python program to circulate values of 'n' variables in the list

ALGORITHM:

- Step1: start
- Step2: read upper limit 'var'
- Step3: read 'var' elements using loop and store them in list
- Step4: pop out each element from list and append to list
- Step5: print list
- Step6: stop

PROGRAM:

```
Var=int(input("enter number of values"))
List1=[]
For val in range (0, var, 1):
    ele=int(input("Enter the element"))
    list1 . append(ele)
print ("circulating the elements of list",list1)
for val in range (0, var,1)
    ele = list1.pop(0)
    list1.append(ele)
    print(list1)
```

RESULT:

To write a Python program to circulate values of 'n' variables in the list.

OUTPUT:

```
enter number of values 4
enter the element12
enter the element43
enter the element11
enter the element56
circulating the elements of list [12, 43, 11, 56]
[43, 11, 56, 12]
[11, 56, 12, 43]
[56, 12, 43, 11]
[12, 43, 11, 56]
```

2.B. EXCHANGE THE VALUES OF TWO VARIABLES

AIM:

To write a Python program to exchange the values of two variables.

ALGORITHM:

Step 1: Start
Step 2: Initialize the function of swap
Step 3: Declare the variables a and b read input
Step 4: Call the function swap a,b
Step 5: The function swap perform the following operation
 temp=a
 a=b
 b=temp
step 6:Print swap number a and b
step 7:Stop

PROGRAM:

```
def swap(a,b):  
    temp=a  
    a=b  
    b=temp  
    print("the value of x after swapping:",a)  
    print("the value of y after swapping:",b)  
    return  
x=int(input("Enter value of x:"))  
y=int(input("Enter value of y:"))  
swap(x,y)
```

RESULT:

Thus the Program was executed successfully and the output was verified.

OUTPUT:

```
Enter value of x:6  
Enter value of y:8  
the value of x after swapping: 8  
the value of y after swapping: 6
```


2.C. DISTANCE BETWEEN TWO POINTS

AIM:

To write the python program to find distance between two points.

ALGORITHM:

STEP 1: start the program

STEP2: Define function distance

STEP3: Read the value a,b,c & d

STEP4: Call the function distance

STEP5: The Distance function performs following

$$d=\text{math.sqrt}(((x2-x1)**2)+((y2-y1)**2))$$

STEP6: Print distance between two points.

STEP7: Stop the program

PROGRAM:

```
import math
```

```
def distance (x1,y1,x2,y2):
```

```
    d=math.sqrt(((x2-x1)**2)+((y2-y1)**2))
```

```
    return d
```

```
a= int(input('Enter the value of x1'))
```

```
b= int(input('Enter the value of x2'))
```

```
c= int(input('Enter the value of x1'))
```

```
d= int(input('Enter the value of x2'))
```

```
print( 'The distance between the two points is', distance(a,b,c,d))
```

RESULT:

Thus the Program was executed successfully and the output was verified.

OUTPUT:

Enter the value of x13

Enter the value of x24

Enter the value of x15

Enter the value of x22

The distance between the two points is 2.8284271247461903

3.A. NUMBER SERIES

AIM:

To write a python program to print series of numbers.

ALGORITHM:

Step 1: Start the program.

Step 2: Take the input from the user by using python input () function.

Step 3: Iterate for loop with the user input number.

Step 4: Increment for loop iteration value by 1, as well as print iteration value.

PROGRAM

```
n = int(input("Please Enter any Number: "))  
  
print("The List of Natural Numbers from 1", "to", n)  
  
for i in range(1, n + 1):  
    print (i, end = ' ')
```

RESULT:

Thus the Program was executed successfully and the output was verified.

OUTPUT:

Please Enter any Number: 10

The List of Natural Numbers from 1 to 10

1 2 3 4 5 6 7 8 9 10

3.B. NUMBER PATTERN

AIM:

To write a python program to print pattern of numbers.

ALGORITHM:

Step 1: Start the program

Step 2: create a variable num.

Step 3: The first outer loop is used to handle a number of rows and the inner loop is used to handle a number of columns.

Step 4: Print (i, end="" "") is used to display numbers and the other print ("") is used for the next line after each row

Step5: Stop the program

PROGRAM

```
num = 5
```

```
for n in range(1, num):
```

```
    for i in range(1, n+1):
```

```
        print(i, end=" ")
```

```
    print("")
```

RESULT:

Thus the Program was executed successfully and the output was verified.

OUTPUT:

```
1
```

```
1 2
```

```
1 2 3
```

```
1 2 3 4
```

3.C.PYRAMID PATTERN

AIM:

To write a python program to print Pyramid pattern.

ALGORITHM:

Step 1: Start the program

Step 2: create a variable num.

Step 3: The first outer loop is used to handle a number of rows and the inner loop is used to handle a number of columns.

Step 4: Print ("*", end=" ") is used to display numbers and the other print () is used for the next line after each row

Step5: Stop the program

PROGRAM

```
num = 5
```

```
for n in range(0, num):
```

```
    for i in range(0, n+1):
```

```
        print(*, end=" ")
```

```
    print()
```

RESULT:

Thus the Program was executed successfully and the output was verified.

OUTPUT:

```
*
```

```
* *
```

```
* * *
```

```
* * * *
```

4.A. OPERATIONS OF LIST

AIM:

To write a python program to implement the operations of a list.

ALOGIRTM

Step: Start the program

Step 2: Create a list named list.

Step 3: Display the items in the list

Step 4: Append a new element to the existing list

Step 5: Delete an element from the existing list.

Step 6: Create a new list and concatenate it with the existing list.

Step 7: Repeat an item to a specified number of times.

Step 8: Stop the program.

PROGRAM:

```
print('operations of list')
list=['python programming','java complete reference','C']
print('Books available in library:')
print(list)
print('append the item in list')
newbook=input("enter the name of book to be inserted")
newlist=list.append(newbook)
print(list)
newlist=['C++ programming','C Programming']
print('Concatenation of 2 lists')
print('Concatenated list: ',(list+newlist))
print('Repetition of an item in list')
print(3*'python programming')
print("Removing an item from list")
list.remove('python programming')
print("Available books")
print(list)
```

RESULT:

Thus the python program was executed and verified successfully.

OUTPUT:

```
operations of list
Books available in library:
['python programming', 'java complete reference', 'C']
append the item in list
enter the name of book to be inserted C++
['python programming', 'java complete reference', 'C', 'C++']
Concatenation of 2 lists
Concatenated list: ['python programming', 'java complete reference', 'C', 'C++', 'C++ programming',
'C Programming']
Repetition of an item in list
python programmingpython programmingpython programming
Removing an item from list
Available books
['java complete reference', 'C', 'C++']
```

4.B. OPERATIONS OF TUPLE

AIM:

To write a python program to implement the operations of a tuple1.

ALOGIRTM

Step1: Start the program.

Step 2: Create a tuple1 named tuple1.

Step 3: Display the items in the tuple1.

Step 4: Create a new tuple1 and concatenate it with the existing tuple1.

Step 5: Repeat an item to a specified number of times.

Step6: Search the materials for construction of civil structure.

Step7:Stop the program.

PROGRAM:

```
print('operations of tuple1')
tuple1=('bricks','cement','steel')
print('Displaying the items in tuple1')
print('Materials required for construction of building:')
print(tuple1)
newtuple1=('sand','wood')
print('Concatenation of 2 tuples')
print('Concatenated tuple1: ',(tuple1+newtuple1))
print('Repetition of an item in tuple1')
print(3*'steel')
print('Searching the materials for construction of building')
if 'C' in tuple1:
    print('present')
else:
    print('not present')
```

RESULT:

Thus the python program was executed and verified successfully.

OUTPUT:

```
operations of tuple1
Displaying the items in tuple1
Materials required for construction of building:
('bricks', 'cement', 'steel')
Concatenation of 2 tuples
Concatenated tuple1: ('bricks', 'cement', 'steel', 'sand', 'wood')
Repetition of an item in tuple1
steelsteelsteel
Searching the materials for construction of building
not present
```

5.A. OPERATIONS OF SETS

AIM:

To write a python program to apply operations of set for programming language.

ALGORITHM:

Step 1: Start the program.

Step 2: Creating a Set with the use of a List and print the result.

Step 3: Addition of Languages to the Set using Update operation and print the result.

Step 4: Removing languages from Set using Remove operation and print the result.

Step5: Find the Language present or not in Set using membership operation and print the result.

Step6: Apply Union operation on Set using “|” operator and print the result.

Step7: Apply Intersection operation on set using “&” operator and print the result.

Step 8: Stop the program.

PROGRAM

```
set1 = set(["python", "C", "C++","java"])
print("\n Programming languages ")
print(set1)
set1.update(["PHP","SQL","VISUAL BASIC"])
print("\nSet after Addition of Languages using Update: ")
print(set1)
set1.remove("C")
set1.remove("PHP")
print("\nSet after Removal of two Languages: ")
print(set1)
if "C" in set1:
    print("present")
else:
    print("not present")
set2=set(["Java script","R"])
set3 = set1|set2
print("\nUnion using '|' operator")
print(set3)
set4 = set1 & set3
print("\nIntersection using '&' operator")
```

```
print(set4)
```

RESULT:

Thus the Program was executed successfully and the output was verified.

OUTPUT:

Programming languages

```
{'C', 'python', 'java', 'C++'}
```

Set after Addition of Languages using Update:

```
{'C', 'python', 'PHP', 'C++', 'java', 'SQL', 'VISUAL BASIC'}
```

Set after Removal of two Languages:

```
{'python', 'C++', 'java', 'SQL', 'VISUAL BASIC'}
```

not present

Union using '|' operator

```
{'python', 'R', 'java', 'SQL', 'Java script', 'VISUAL BASIC', 'C++'}
```

Intersection using '&' operator

```
{'python', 'C++', 'java', 'SQL', 'VISUAL BASIC'}
```


5.B. OPERATIONS OF DICTIONARIES

AIM:

To write a python program to apply operations of dictionary for components of automobile.

ALGORITHM:

Step 1: Start the program.

Step 2 Creating a Dictionary for components of automobile with Integer Keys and print the result.

Step 3: Updating existing Key's Value and print the result.

Step 4: accessing a component using key and print the result.

Step5: Deleting a key using pop and print the result.

Step6: Find the components present or not in Dictionary using membership operator and print the result.

Step7: Apply Intersection operation on set using “&” operator and print the result.

Step 8:Stop the program.

PROGRAM

```
Dict = {1: 'Engine', 2: 'Gearbox', 3: 'lights'}
print("\nComponents of automobile ")
print(Dict)
Dict[4] = 'Battery'
print("\nUpdated Components: ")
print(Dict)
print("Accessing a Component using key:")
print(Dict[1])
pop_ele = Dict.pop(2)
print('\n components after deletion: ' + str(Dict))
if "Break" in Dict.values():
    print('Present')
else:
    print('Not present')
```

RESULT:

Thus the Program was executed successfully and the output was verified.

OUTPUT:

Components of automobile

{1: 'Engine', 2: 'Gearbox', 3: 'lights'}

Updated Components:

{1: 'Engine', 2: 'Gearbox', 3: 'lights', 4: 'Battery'}

Accessing a Component using key:

Engine

components after deletion: {1: 'Engine', 3: 'lights', 4: 'Battery'}

Not present

6.A. FACTORIAL OF A NUMBER USING FUNCTION

AIM:

To write a Python program to print factorial of a number using function

ALGORITHM:

Step 1. Start the program

Step 2: Defining a recursive function

Step 3: Read the input from the user

Step 4 : if input number is negative then return an error message

Step 5: elif the input number is 0 then display 1 as output

Step6: else calculate the factorial by calling the user defined function

Step7: stop the program

PROGRAM:

```
def factorial(num):
    if num == 1:
        return num
    else:
        return num * factorial(num - 1)
num = int(input("Enter the number: "))
if num < 0:
    print("Invalid input")
elif num == 0:
    print ("Factorial of 0 is 1")
else:
    print ("Factorial of %d is %d" %(num, factorial(num)))
```

RESULT:

Thus the Program was executed successfully and the output was verified.

OUTPUT:

Enter the number: 5

Factorial of 5 is 120

6(B). FIND LARGEST NUMBER IN A LIST USING FUNCTION

AIM:

To write a Python program to find largest number in a list using function.

ALGORITHM:

Step 1.Start the program

Step 2:defining a mymax function

```
# Assume first number in list is largest
```

```
# initially and assign it to variable "max"
```

```
#Now traverse through the list and compare each number with "max" value. Whichever is largest assign that value to "max".
```

Step 3. Read the elements in the list

Step4. Print largest number by calling the user defined function

Step5:stop the program

PROGRAM:

```
def myMax(list1):
```

```
    max = list1[0]
```

```
    for x in list1:
```

```
        if x > max:
```

```
            max = x
```

```
    return max
```

```
list1 = [10, 20, 4, 45, 99]
```

```
print("Largest element is:", myMax(list1))
```

RESULT:

Thus the Program was executed successfully and the output was verified.

OUTPUT:

Largest element is: 99

6(C). FIND AREA OF SHAPES(SQUARE) USING FUNCTION

AIM:

To write a Python program to find area of shapes(square) using function

ALGORITHM:

Step 1: Start the program

Step 2: Defining a Area of square function

Step 3: Read the side of square

Step4: Print area of square by calling the user defined function

Step5: Stop the program

Program:

```
def Areaofsquare(side):  
    Area = side*side  
    return Area  
side = int(input('enter a the value of side'))  
print(Areaofsquare(side))
```

RESULT:

Thus the Program was executed successfully and the output was verified.

OUTPUT:

enter a the value of side8

64

7.A. REVERSE A STRING

AIM:

To write a Python program to reverse the strings

ALGORITHM:

Step 1: Start the program

Step 2: Defining a reversed_string function

Step 3: Read the string

Step4: Print the reversed string by calling the user defined function

Step5: Stop the program

Program:

```
def reversed_string(text):  
    if len(text) == 1:  
        return text  
    return reversed_string(text[1:] + text[:1])  
  
print(reversed_string("Python Programming!"))
```

RESULT:

Thus the Program was executed successfully and the output was verified.

OUTPUT:

!gnimmargorP nohtyP

7.B. PALINDROME

AIM:

To write a Python program to reverse the strings

ALGORITHM:

Step 1: Start the program

Step 2: Defining a isPalindrome function

Step 3: Read the string

Step4: Check the string is palindrome or not by calling the user defined function. Condition is true print yes its palindrome.

Step5: otherwise print no its not palindrome

Step5: Stop the program

Program:

```
def isPalindrome(s):
```

```
    return s == s[::-1]
```

```
s = input("enter any string :")
```

```
ans = isPalindrome(s)
```

```
if ans:
```

```
    print(s,"Yes it's a palindrome")
```

```
else:
```

```
    print(s,"No it's not a palindrome")
```

RESULT:

Thus the Program was executed successfully and the output was verified.

OUTPUT:

enter any string :malayalam

malayalam Yes it's a palindrome

7.C. CHARACTER COUNT

AIM:

To write a Python program to count the character in the strings

ALGORITHM:

Step 1: Start the program

Step 2: Defining a count_chars function

Step 3: Read the string

Step4: Print the no.of characters in a string by calling the user defined function

Step5: Stop the program

Program:

```
def count_chars(txt):  
    result = 0  
    for char in txt:  
        result += 1 # same as result = result + 1  
    return result  
  
text=input("enter any string")  
print("no.of characters in a string",count_chars(text))
```

RESULT:

Thus the Program was executed successfully and the output was verified.

OUTPUT:

enter any string welcome

no.of characters in a string 7

7.D. REPLACING CHARACTERS

AIM:

To write a Python program to replace the character in the strings

ALGORITHM:

Step 1: Start the program

Step 2: Read the string

Step3: Print the replaced string by using replace() method

Step4: Stop the program

Program:

```
string = "python is a programming language is powerful"
```

```
print(string.replace("a", "A",1))
```

```
print(string.replace("a", "A"))
```

RESULT:

Thus the Program was executed successfully and the output was verified.

OUTPUT:

```
python is A programming language is powerful
```

```
python is A progrAmming lAnguAge is powerful
```

8(A). PANDA LIBRARY MODULE

AIM:

To write a python program to implement the panda library module.

ALOGIRTM

Step1: Start the program

Step 2: **import** pandas as pd.

Step 3: creating a series as data.

Step 4: creating a series as data1

Step 5: print the series data and data1.

Step 6: Stop the program.

PROGRAM:

```
import pandas as pd
data = pd.Series([5, 2, 3,7], index=['a', 'b', 'c', 'd'])
data1 = pd.Series([1, 6, 4, 9], index=['a', 'b', 'd', 'e'])
print(data, "\n\n", data1)
```

RESULT:

Thus the python program was executed and verified successfully.

OUTPUT:

a 5

b 2

c 3

d 7

dtype: int64

a 1

b 6

d 4

e 9

dtype: int64

8(B). NUMPY LIBRARY MODULE

AIM:

To write a python program to implement the numpy library module.

ALOGIRTM

Step1: Start the program

Step 2: import numpy as np

Step 3: creating a simple array as data.

Step 4: print the simple array data.

Step 5: Stop the program.

PROGRAM:

```
import numpy as np
```

```
data = np.array([1,3,4,7])
```

```
print(data)
```

RESULT:

Thus the python program was executed and verified successfully.

OUTPUT:

```
[1 3 4 7]
```

8(C).MATPLOTLIB LIBRARY MODULE

AIM:

To write a python program to implement the matplotlib library module.

ALOGIRTM

Step1: Start the program

Step 2: import pyplot as plt

Step 3: Read x-axis values

Step 3: Read Y-axis values

Step 3: Define the function to plot

Step 4: Call the function to show the plot

Step 5: Stop the program.

PROGRAM:

```
from matplotlib import pyplot as plt
```

```
x = [5, 2, 9, 4, 7]
```

```
y = [10, 5, 8, 4, 2]
```

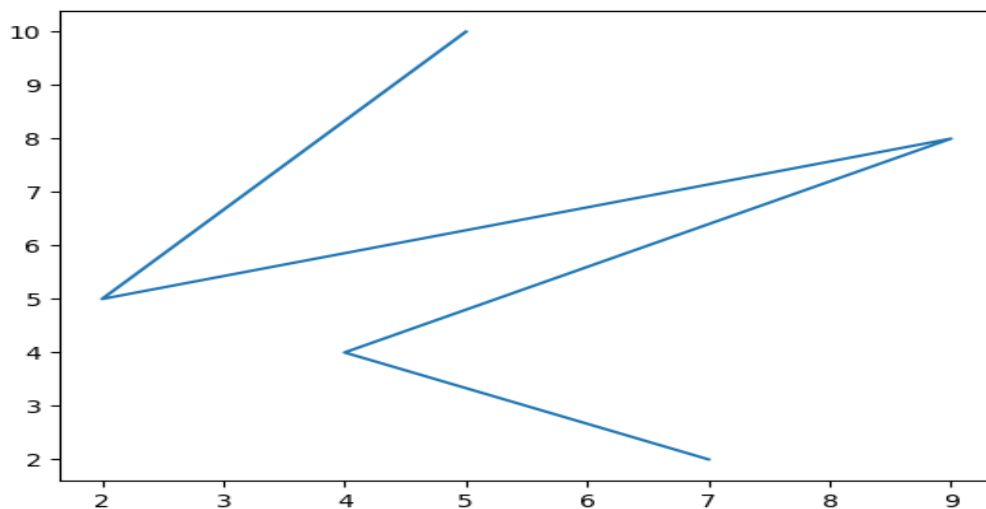
```
plt.plot(x,y)
```

```
plt.show()
```

RESULT:

Thus the python program was executed and verified successfully.

OUTPUT:



8(D).SCIPY LIBRARY MODULE

AIM:

To write a python program to implement the scipy library module.

ALOGIRTM

Step1: Start the program

Step 2: import numpy as np

Step 3: importing linalg function from scipy

Step 4: Compute the determinant of a matrix

Step 5: Stop the program.

PROGRAM:

```
import numpy as np
A = np.array([[1,2,3],[4,2,6],[7,3,9]])
from scipy import linalg
linalg.det(A)
```

RESULT:

Thus the python program was executed and verified successfully.

OUTPUT:

6.0

9.A. COPY CONTENTS OF ONE FILE TO ANOTHER FILE

AIM:

To Copy contents of one file to another file

ALGORITHM:

STEP 1: Start

STEP 2: open both files

STEP 3: Add file name as argument

STEP 4: read content from first file

STEP 5: append content to second

file

STEP 6: Stop

PROGRAM:

with open('first.txt','r') as firstfile, open('second.txt','a') as secondfile:

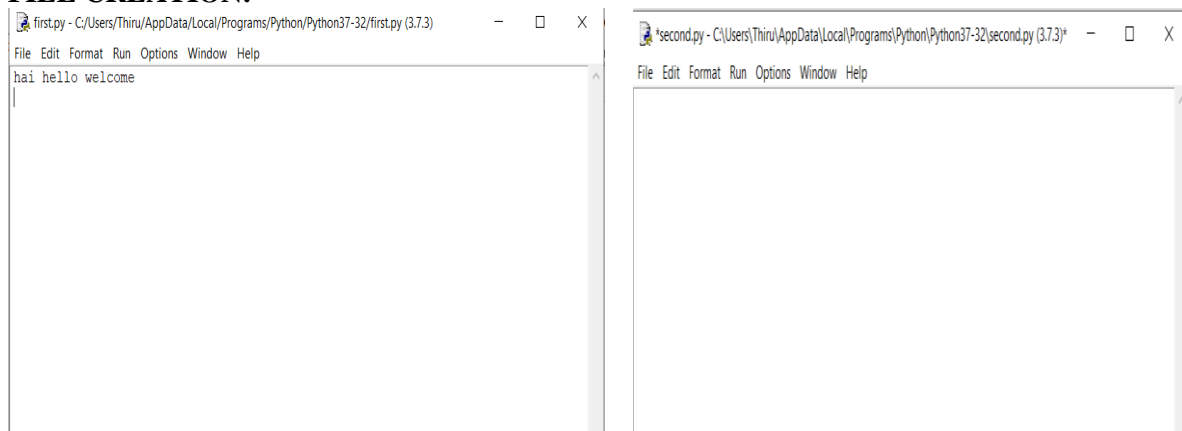
for line in firstfile:

secondfile.write(line)

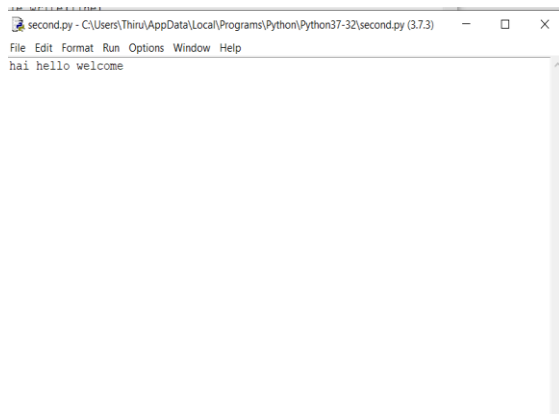
RESULT:

Thus the python program was executed and verified successfully.

FILE CREATION:



OUTPUT:



9.B.WORD COUNT

AIM:

To find the word and lines in command line arguments.

ALGORITHM:

STEP 1: Start

STEP 2: Add arguments to find the words and lines

STEP 3: Add file name as argument

STEP 4: Parse the arguments to get the values

STEP 5: Format and print the words

STEP 6: Stop

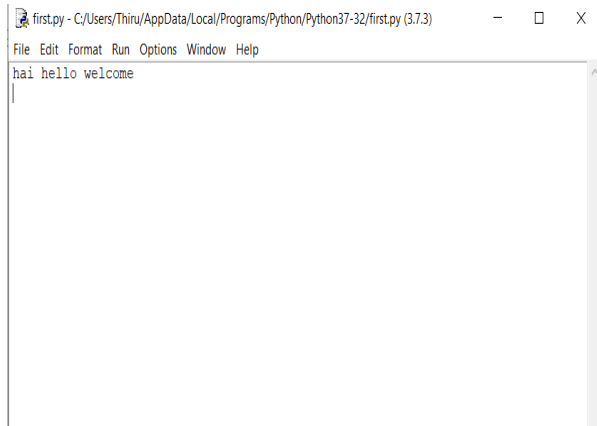
PROGRAM:

```
fname = input("Enter file name: ")
num_words = 0
with open(fname, 'r') as f:
    for line in f:
        words = line.split()
        num_words += len(words)
print("Number of words:")
print(num_words)
```

RESULT:

Thus the python program was executed and verified successfully.

FILE CREATION:

A screenshot of a Python IDE window. The title bar shows the file path: 'first.py - C:/Users/Thiru/AppData/Local/Programs/Python/Python37-32/first.py (3.7.3)'. The menu bar includes 'File', 'Edit', 'Format', 'Run', 'Options', 'Window', and 'Help'. The main text area contains the text 'hai hello welcome' on a single line.

OUTPUT:

Enter file name: first.py

Number of words:

3

9.C. LONGEST WORD

AIM:

To write a python program to find longest word from file

ALGORITHM:

STEP 1: Start

STEP 2: Open text file say 's1.py' in read mode using open function

STEP 3: Pass file name and access mode to open function

STEP 4: Read the whole content of text file using read function and store it in another variable say 'str'

STEP 5: Use split function on str object and store words in variable say 'words'

STEP 6: Find maximum word from words using len method

STEP 7: Iterate through word by word using for loop

STEP 8: Use if loop within for loop to check the maximum length of word

STEP 9: Store maximum length of word in variable say 'longest_word'

STEP 10: Display longest_word using print function

STEP 11: Stop

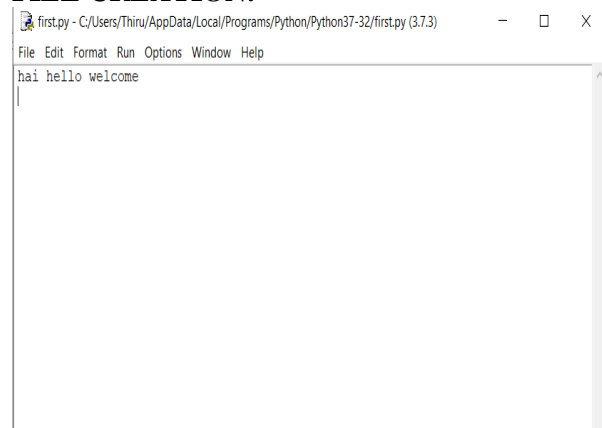
PROGRAM:

```
fin = open("first.py","r")
str = fin.read()
words = str.split()
max_len = len(max(words, key=len))
for word in words:
    if len(word)==max_len:
        longest_word =word
print(longest_word)
```

RESULT:

Thus the python program was executed and verified successfully.

FILE CREATION:



OUTPUT:

welcome

10.A. DIVIDE BY ZERO ERROR

AIM

To write a python program to implement exception handling using divide by zero error.

Algorithm

step 1: Start
Step 2: Enter the inputs a and b
Step 3: Calculate $(a+b)/(a-b)$ in try block
Step 4: if a equal to b try block throws an exception.
Step 5: The exception is caught and handled.
Step 6: stop

Program

```
a=int(input("Entre a="))
b=int(input("Entre b="))
try:
    c = ((a+b) / (a-b))
#Raising Error
    if a==b:
        raise ZeroDivisionError
#Handling of error
except ZeroDivisionError:
    print ("a/b result in 0")
else:
    print (c)
```

RESULT:

Thus the python program was executed and verified successfully.

OUTPUT:

```
Enter a=4
Enter b=4
a/b result in 0
```

10.B. VOTER'S AGE VALIDITY

AIM

To write a python program to implement exception handling using voter's age validity.

Algorithm

step 1: Start
Step 2: Enter the age
Step 3: Check whether the age is less than 18.
Step 4: If less than 18 throw an exception
Step 5: The exception is caught and handled.
Step 6: stop

Program:

```
a=int(input("Enter your age"))

try:

    c = a

#Raising Error

    if c<18:

        raise ValueError

#Handling of error

except ValueError:

    print ("not eligible for voting - enter above 18")

else:

    print (c)
```

RESULT:

Thus the python program was executed and verified successfully.

OUTPUT:

```
Enter your age15

not eligible for voting - enter above 18

Enter your age20

20
```

10.C. STUDENT MARK RANGE VALIDATION

AIM

To write a python program to implement exception handling using Students mark range.

Algorithm

step 1: Start

Step 2: Enter the mark of student

Step 3: Check whether the mark is between 0 and 100

Step 4: if the range is not between 0 and 100 then throw an exception.

Step 5: The exception is caught and handled.

Step 6: stop

Program:

```
a=int(input("Enter any marks"))
```

```
try:
```

```
    c = a
```

```
#Raising Error
```

```
    if c>0 and c>100:
```

```
        raise ValueError
```

```
#Handling of error
```

```
except ValueError:
```

```
    print ("not correct students mark range value btween 1-100")
```

```
else:
```

```
    print (c)
```

RESULT:

Thus the python program was executed and verified successfully.

OUTPUT:

```
Enter any marks102
```

```
not correct students mark range value btween 1-100
```

```
Enter any marks90
```

```
90
```

11. EXPLORING PYGAME TOOL.

Pygame

- Pygame is a cross-platform set of Python modules which is used to create video games.
- It consists of computer graphics and sound libraries designed to be used with the Python programming language.
- Pygame was officially written by **Pete Shinnners** to replace PySDL.
- Pygame is suitable to create client-side applications that can be potentially wrapped in a standalone executable.

Pygame Installation

Install pygame in Windows

Before installing Pygame, Python should be installed in the system, and it is good to have 3.6.1 or above version because it is much friendlier to beginners, and additionally runs faster. There are mainly two ways to install Pygame, which are given below:

1. Installing through pip: The good way to install Pygame is with the pip tool (which is what python uses to install packages). The command is the following:

```
py -m pip install -U pygame --user
```

2. Installing through an IDE: The second way is to install it through an IDE and here we are using Pycharm IDE. Installation of pygame in the pycharm is straightforward. We can install it by running the above command in the terminal or use the following steps:

- Open the **File** tab and click on the **Settings** option.
- **import** pygame

Simple pygame Example

```
import pygame
pygame.init()
screen = pygame.display.set_mode((400,500))
done = False
while not done:
    for event in pygame.event.get():
        if event.type == pygame.QUIT:
            done = True
    pygame.display.flip()
```

12.A. SIMULATE BOUNCING BALL USING PYGAME

AIM:

To write a python program to simulate bouncing ball using pygame.

PROGRAM/SOURCE CODE :

```
import sys, pygame

pygame.init()

size = width, height = 800, 600

speed = [1, 1]

background = 255, 255, 255

screen = pygame.display.set_mode(size)

pygame.display.set_caption("Bouncing ball")

ball = pygame.image.load("ball.png")

ballrect = ball.get_rect()

while 1:

    for event in pygame.event.get():

        if event.type == pygame.QUIT:

            sys.exit()

    ballrect = ballrect.move(speed)

    if ballrect.left < 0 or ballrect.right > width:

        speed[0] = -speed[0]

    if ballrect.top < 0 or ballrect.bottom > height:

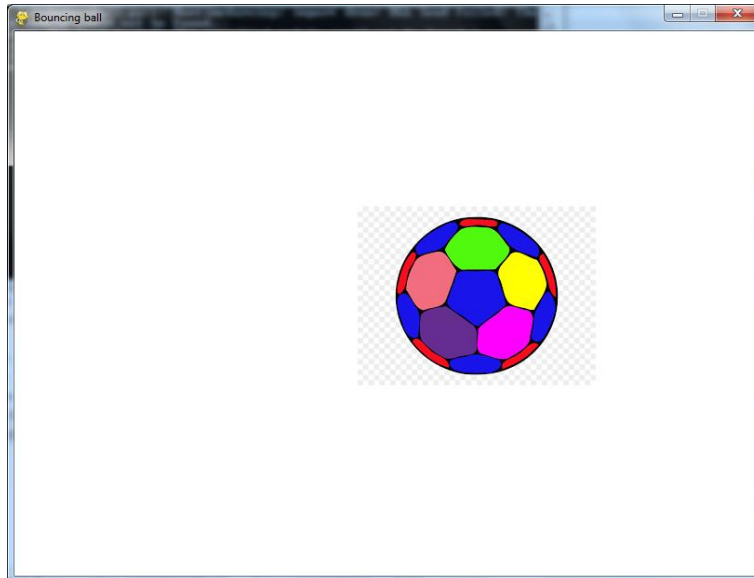
        speed[1] = -speed[1]

    screen.fill(background)

    screen.blit(ball, ballrect)

    pygame.display.flip()
```

OUTPUT



RESULT:

Thus the program to simulate bouncing ball using pygame is executed and the output is obtained.

12.B. SIMULATE CAR RACE USING PYGAME

AIM:

To write a python program to simulate car race using pygame.

PROGRAM/SOURCE CODE :

```
import random

from time import sleep

import pygame

class CarRacing:

    def __init__(self):

        pygame.init()

        self.display_width = 800

        self.display_height = 600

        self.black = (0, 0, 0)

        self.white = (255, 255, 255)

        self.clock = pygame.time.Clock()

        self.gameDisplay = None

        self.initialize()

    def initialize(self):

        self.crashed = False

        self.carImg = pygame.image.load('.\\img\\car.png')

        self.car_x_coordinate = (self.display_width * 0.45)

        self.car_y_coordinate = (self.display_height * 0.8)

        self.car_width = 49

        # enemy_car

        self.enemy_car = pygame.image.load('.\\img\\enemy_car_1.png')

        self.enemy_car_startx = random.randrange(310, 450)

        self.enemy_car_starty = -600

        self.enemy_car_speed = 5
```

```

self.enemy_car_width = 49

self.enemy_car_height = 100

# Background

self.bgImg = pygame.image.load(".\\img\\back_ground.jpg")

self.bg_x1 = (self.display_width / 2) - (360 / 2)

self.bg_x2 = (self.display_width / 2) - (360 / 2)

self.bg_y1 = 0

self.bg_y2 = -600

self.bg_speed = 3

self.count = 0

def car(self, car_x_coordinate, car_y_coordinate):

    self.gameDisplay.blit(self.carImg, (car_x_coordinate, car_y_coordinate))

def racing_window(self):

    self.gameDisplay = pygame.display.set_mode((self.display_width, self.display_height))

    pygame.display.set_caption('Car Dodge')

    self.run_car()

def run_car(self):

    while not self.crashed:

        for event in pygame.event.get():

            if event.type == pygame.QUIT:

                self.crashed = True

            # print(event)

            if (event.type == pygame.KEYDOWN):

                if (event.key == pygame.K_LEFT):

                    self.car_x_coordinate -= 50

                    print ("CAR X COORDINATES: %s" % self.car_x_coordinate)

                if (event.key == pygame.K_RIGHT):

                    self.car_x_coordinate += 50

```



```

        print ("CAR X COORDINATES: %s" % self.car_x_coordinate)

        print ("x: {x}, y: {y}".format(x=self.car_x_coordinate, y=self.car_y_coordinate))

self.gameDisplay.fill(self.black)

self.back_ground_raod()

self.run_enemy_car(self.enemy_car_startx, self.enemy_car_starty)

self.enemy_car_starty += self.enemy_car_speed

if self.enemy_car_starty > self.display_height:

    self.enemy_car_starty = 0 - self.enemy_car_height

    self.enemy_car_startx = random.randrange(310, 450)

self.car(self.car_x_coordinate, self.car_y_coordinate)

self.highscore(self.count)

self.count += 1

if (self.count % 100 == 0):

    self.enemy_car_speed += 1

    self.bg_speed += 1

if self.car_y_coordinate < self.enemy_car_starty + self.enemy_car_height:

    if self.car_x_coordinate > self.enemy_car_startx and self.car_x_coordinate <
self.enemy_car_startx + self.enemy_car_width or self.car_x_coordinate + self.car_width >
self.enemy_car_startx and self.car_x_coordinate + self.car_width < self.enemy_car_startx +
self.enemy_car_width:

        self.crashed = True

        self.display_message("Game Over !!!")

if self.car_x_coordinate < 310 or self.car_x_coordinate > 460:

    self.crashed = True

    self.display_message("Game Over !!!")

pygame.display.update()

self.clock.tick(60)

def display_message(self, msg):

    font = pygame.font.SysFont("comicansms", 72, True)

    text = font.render(msg, True, (255, 255, 255))

```

```

self.gameDisplay.blit(text, (400 - text.get_width() // 2, 240 - text.get_height() // 2))

self.display_credit()

pygame.display.update()

self.clock.tick(60)

sleep(1)

car_racing.initialize()

car_racing.racing_window()

def back_ground_raod(self):

    self.gameDisplay.blit(self.bgImg, (self.bg_x1, self.bg_y1))

    self.gameDisplay.blit(self.bgImg, (self.bg_x2, self.bg_y2))

    self.bg_y1 += self.bg_speed

    self.bg_y2 += self.bg_speed

    if self.bg_y1 >= self.display_height:

        self.bg_y1 = -600

    if self.bg_y2 >= self.display_height:

        self.bg_y2 = -600

def run_enemy_car(self, thingx, thingy):

    self.gameDisplay.blit(self.enemy_car, (thingx, thingy))

def highscore(self, count):

    font = pygame.font.SysFont("arial", 20)

    text = font.render("Score : " + str(count), True, self.white)

    self.gameDisplay.blit(text, (0, 0))

def display_credit(self):

    font = pygame.font.SysFont("lucidaconsole", 14)

    text = font.render("Thanks for playing!", True, self.white)

    self.gameDisplay.blit(text, (600, 520))

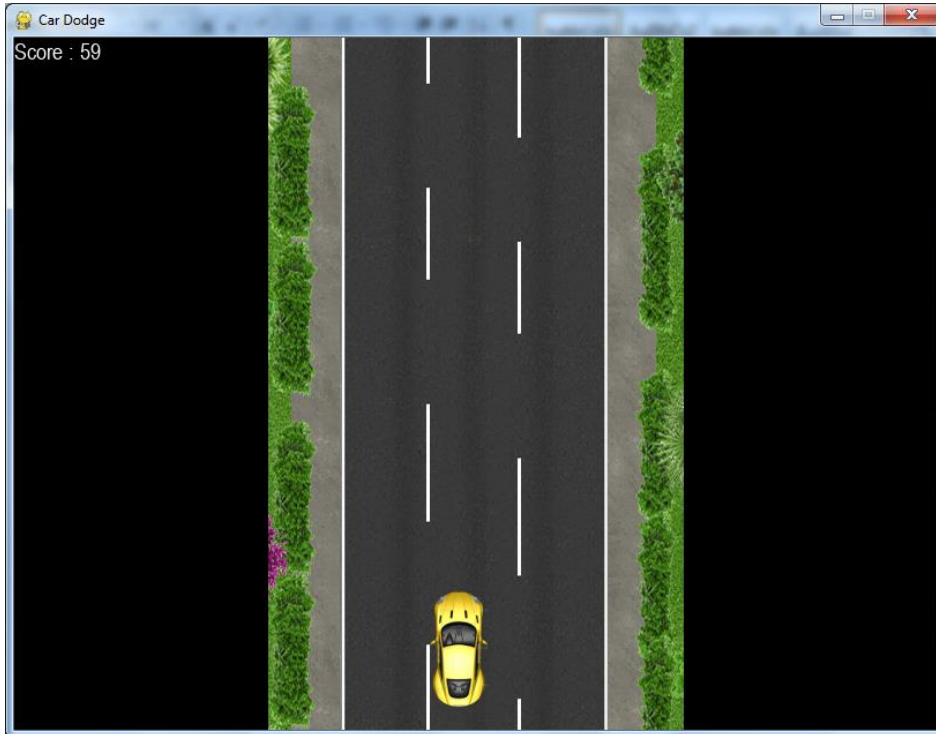
if __name__ == '__main__':

```

```
car_racing = CarRacing()
```

```
car_racing.racing_window()
```

OUTPUT



RESULT:

Thus the python program was executed and verified successfully.