EX NO: 1a)	ELECTRICITY BILLING
DATE:	

AIM:

To Develop a flow chart for Electricity billing

ALGORITHM:

STEP1: Start

STEP2: Read the previous unit and Current unit

STEP3: Calculate used unit = Current unitPrevious unit

STEP4: Calculate Electricity bill by from used unit

STEP5: Print the amount of Electricity bill

STEP6: Stop

PSEUDO CODE:

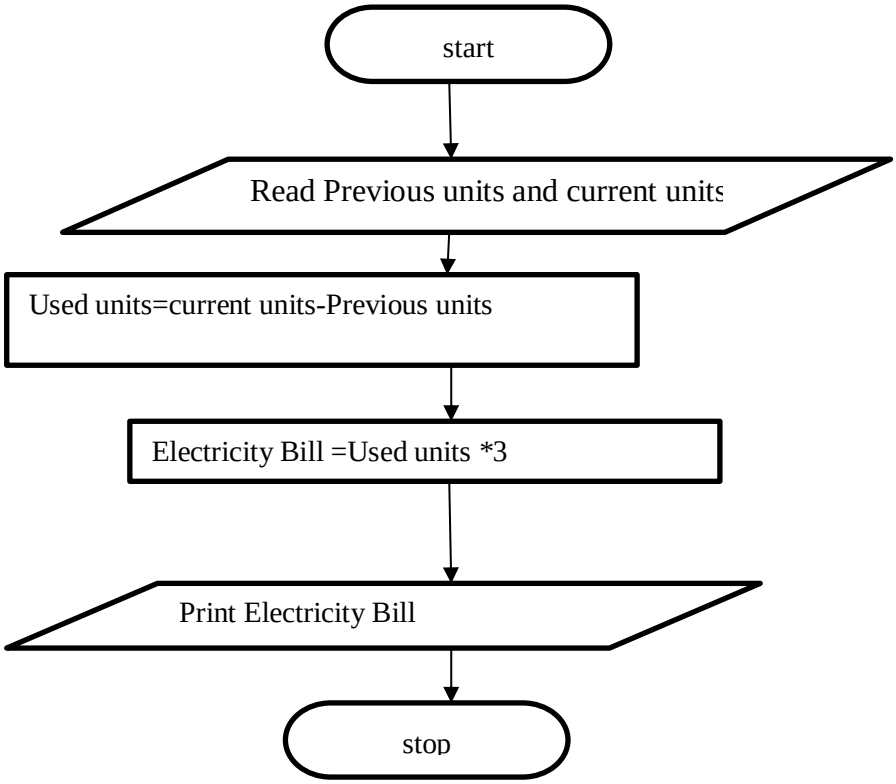
READ Previous unit ,Current unit

CACULATE used unit =Current unit-Previous unit

OBTAIN Electricity Bill = Used unit*3

PRINT Electricity Bill

FLOWCHART:


start
Read Previous units and current units
Used units=current units-Previous units
Electricity Bill =Used units *3
Print Electricity Bill
stop

RESULT:

Thus the flowchart of electric bill was developed successfully

EX NO: 1b)	REATIL SHOP BILLING
DATE:	

AIM:

To Develop a flow chart for Reatil Shop billing

ALGORITM:

STEP1: Start

STEP2: Read the barcode of the product

STEP3: Display the Product name and the amount

STEP4: Check if more product is available. go to Step2,otherwise go to step6

SETP5: Calculate the total cost of the product

STEP6: Print the total product

STEP7: Stop

PSEUDO CODE:

IF more product available

THENREAD barcode

DISPLAY Product name,

amountELSE

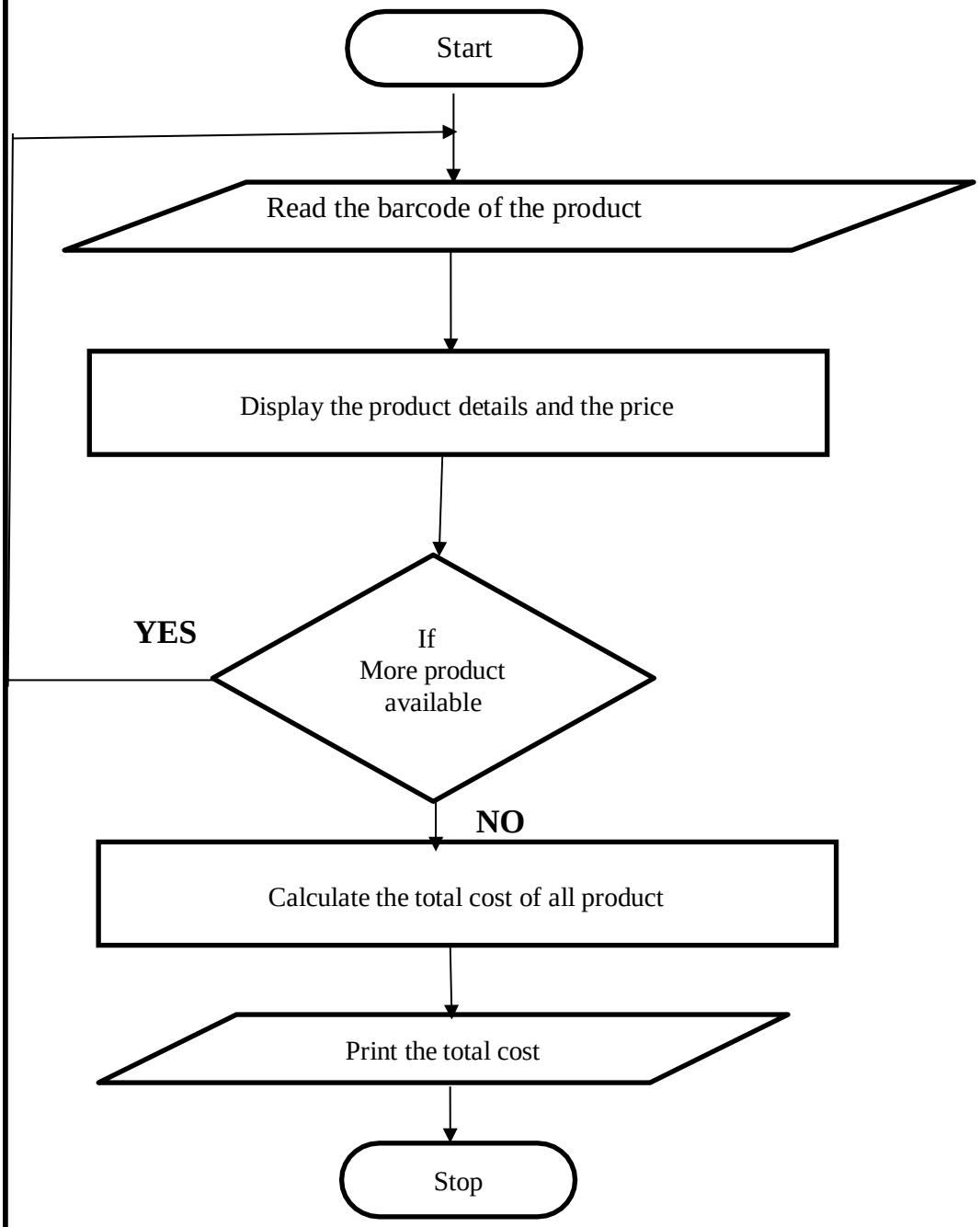
CALCULATE Total

CostPRINT TOTAL

Cost

ENDIF

FLOWCHART:


Start
Read the barcode of the product
Display the product details and the price
YES
If
More product
available
NO
Calculate the total cost of all product
Print the total cost
Stop

RESULT:

Thus the flowchart of Reatil Shop billing was developed successfully

EX NO:1c) DATE:	SIN SERIES

AIM:

To Develop a flow chart for Sin series.

Sin(x)= $x - \frac{x}{1!} + \frac{x^3}{3!} - \frac{x^5}{5!} + \frac{x^7}{7!} \ldots$

ALGORITHM:

STEP1: Start

STEP2: Read x , n

STEP3: Convert x values into radian x=3.14/180

STEP4: Substitute t=x and sum=x

STEP5: Initialize i=1STEP6: for i<n+1

STEP7: i=i+1

STEP8: Calculate t=(t*pow(-1),(2*i-1)*x*x))/(2*i*(2*i+1))

STEP9: Calculate Sum=Sum+ t

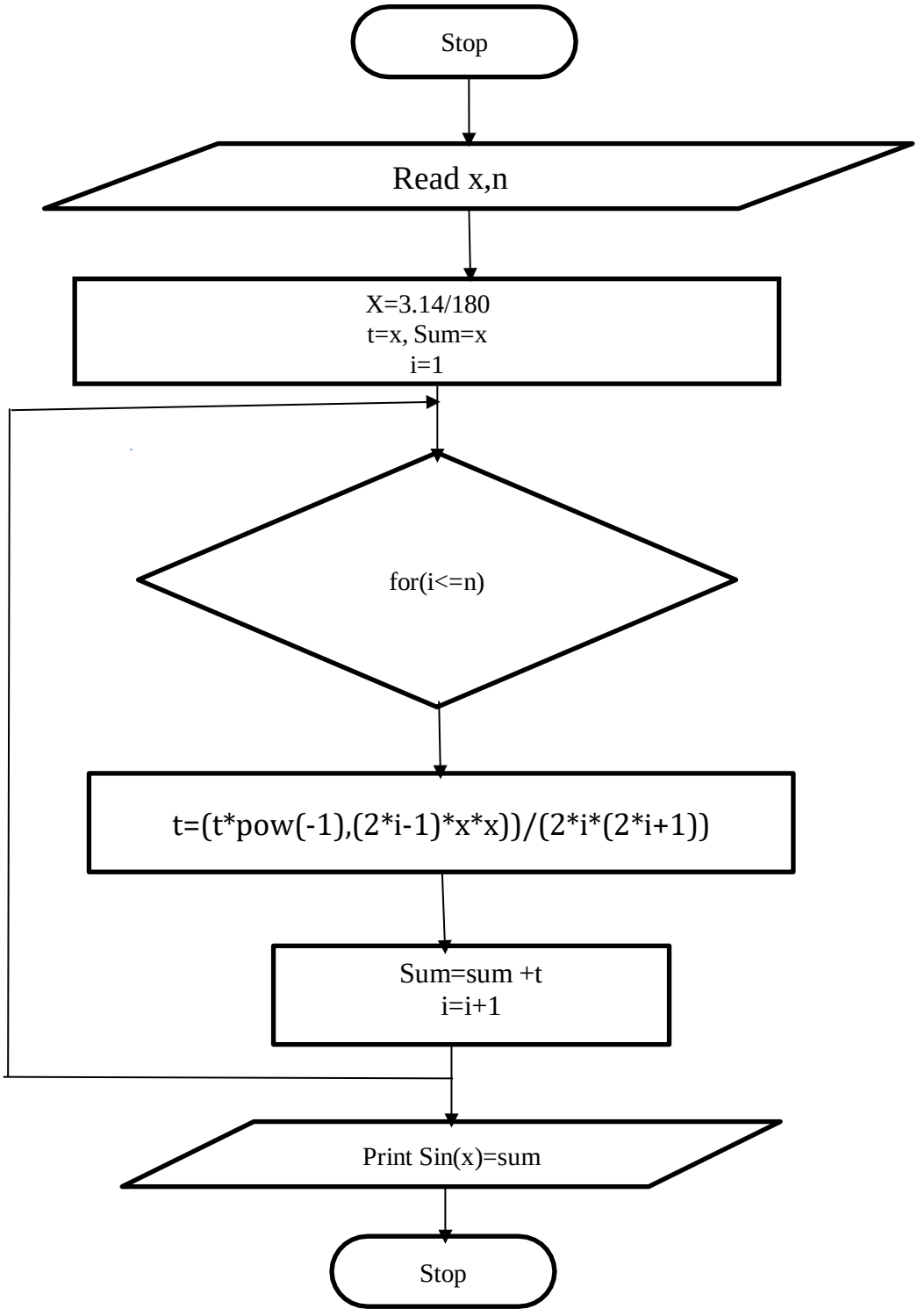
STEP10: Print Sum

STEP11: Stop

PSEUDO CODE:

```
Read x,n
CONVERT x=x*3.14/180
SUBSTITUTE t=x and sum=x
INITIALIZE i=1
FOR i<n+1
CALCULATE t=(t*pow(-1),(2*i-1)*x*x))/(2*i*(2*i+1))
CALCULATE Sum=Sum+ t
PRINT Sum
ENDFOR
```

FLOWCHART:


Stop
Read x,n
X=3.14/180
t=x, Sum=x
i=1
for(i<=n)
t=(t*pow(-1),(2*i-1)*x*x))/(2*i*(2*i+1))
Sum=sum +t
i=i+1
Print Sin(x)=sum
Stop

RESULT:

Thus the flowchart of Sin series. was developed successfully

EX NO:1d) DATE:	ELECTRICAL CURRENT IN THREE PHASE AC CIRCUIT

AIM:

To develop a flowchart for Electrical Current in Three Phase AC Circuit.

ALGORITHM:

STEP1: Start

STEP2: Get the value of Current , voltage, resistance,power factor

STEP3: Electric current= 3* Current*voltage*resistance*power factor

STEP4: Print Electric current

STEP5:Stop

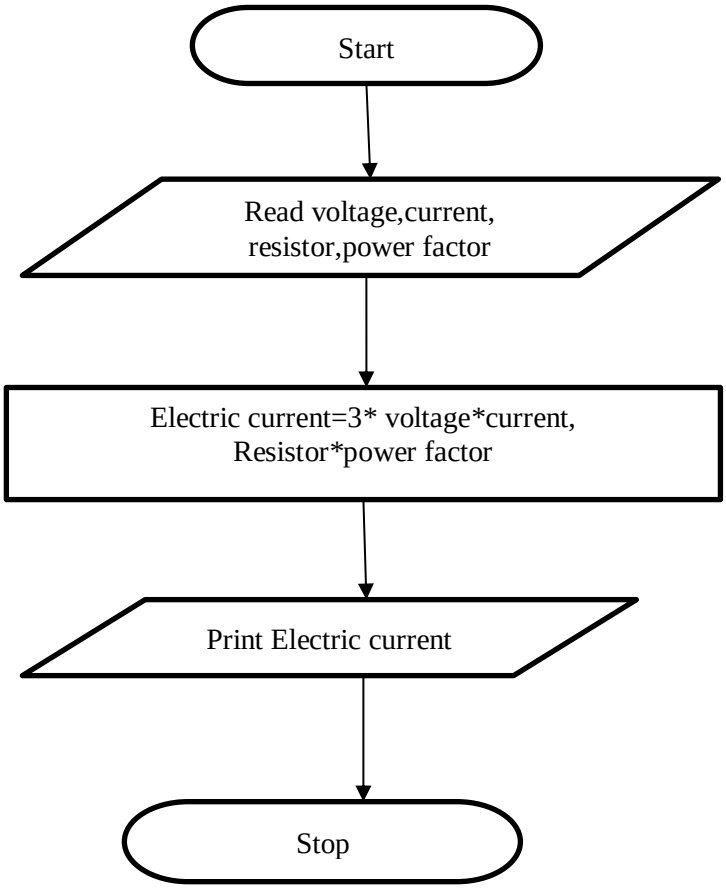
PSEUDO CODE:

READ Current , voltage, resistance, power factor

COMPUTE Electric current= 3* Current*voltage*resistance*power factor

PRINT Electric current

FLOWCHART:


Start
Read voltage,current,
resistor,power factor
Electric current=3* voltage*current,
Resistor*power factor
Print Electric current
Stop

RESULT:

Thus the flowchart of Electrical Current in Three Phase AC Circuit developed successfully

EXPT.NO 2.a	PYTHON PROGRAMMING USING SIMPLE STATEMENTS AND EXPRESSIONS - EXCHANGE THE VALUES OF TWO VARIABLES
DATE:	

AIM:

Write a python program to exchange the values of two variables

ALGORITHM:

STEP1: Declared a temporary variable a and b

STEP2: Assign the value of a and b,

STEP3: Assign the value of a to b, and b to a

STEP4: We can use, a,b = b,a

STEP5: Print the result

PROGRAM1:

```
a=10
b=20
a,b=b,a
print("The swapping of a value is=",a)
print("The swapping of b value is=",b)
```

OUTPUT:

```
The swapping of a value is= 20
The swapping of b value is= 10
```

PROGRAM2:

```
a=input("Enter the value a:")
b=input("Enter the value b:")
temp=a
a=b
b=temp
print("The swap value of a:{}".format(a))
print("The swap value of b:{}".format(b))
```

OUTPUT:

```
Enter the value a:20
Enter the value b:30
The swap value of a:30
The swap value of b:20
```

RESULT:

Thus the swapping of two numbers python program was executed successfully and verified.

EXPT.NO.2b	PYTHON PROGRAMMING USING SIMPLE STATEMENTS AND EXPRESSIONS - CIRCULATE THEVALUES OF N VARIABLES
DATE:	

AIM:

Write a python program to circulate the values of n variables

ALGORITHM:

STEP1: Circulate the values of n variables.

STEP2: Get the input from the user

STEP 3: To create the empty list then declare the conditional statements using for loop

STEP 4: Using append operation to add the element in the list and the values are rotate by using this appendoperation

STEP 5: Stop the program

PROGRAM:

```
def circulate(a,n):
    cir=a[n:]+a[:n]
    print(cir)
a=[32,46,52,63]
circulate(a,2)
```

OUTPUT:

[52, 63, 32, 46]

RESULT:

Thus the python program to circulate the values of n variables was executed successfully and verified

EXPT.NO.2.C	PYTHON PROGRAMMING USING SIMPLE STATEMENTS AND EXPRESSIONS (CALCULATE THE DISTANCE BETWEEN TWOPOINTS)
DATE:	

AIM:

Write a python program to calculate the distance between two numbers

PROCEDURE:

Step 1: Start the program.

Step 2: Read all the values of x1,x2,y1,y2.

Step 3: Calculate the result.

Step 4: Print the result.

Step 5: Stop the program

PROGRAM

```
import math
x1=int(input("enter the value of x1="))
x2=int(input("enter the value of x2="))
y1=int(input("enter the value of y1="))
y2=int(input("enter the value of y2="))
dx=x2-x1
dy=y2-y1
d=dx**2+dy**2
result=math.sqrt(d)
print("Distance is:",result)
```

OUTPUT:

```
enter the value of x1=3
enter the value of x2=7
enter the value of y1=2
enter the value of y2=8
Distance is:7.211102
```

RESULT:

Thus the distance between of two points was successfully executed and verified

EX.NO.3 (a) DATE:	**SCIENTIFIC PROBLEMS USING CONDITIONALS AND ITERATIVE LOOPS.- NUMBERSERIES**

AIM:

Write a Python program with conditional and iterative statements for Number Series

ALGORITHM:

STEP 1: Start the program.

STEP 2: Read the value of n.

STEP 3: Initialize i = 1,x=0.

STEP 4: Repeat the following until i is less than or equal to n.

STEP 4.1: x=x*2+1.

STEP 4.2: Print x.

STEP 4.3: Increment the value of i .

STEP 5: Stop the program.

PROGRAM:

```
n=int(input("Enter the number of terms for the series:"))
i=1
x=0
while(i<=n):
    x=x*2+1
    print(x)
    i=i+1
```

OUTPUT:

```
Enter the number of terms for the series:5
1
3
7
15
31
```

RESULT:

Thus the python program to print numbers patterns is executed and verified

EXPT.NO.3 b DATE:	**SCIENTIFIC PROBLEMS USING CONDITIONALS AND ITERATIVE LOOPS. –NUMBER PATTERNS**

AIM:

To write a Python program with conditional and iterative statements for Number Pattern.

ALGORITHM:

STEP 1: Start the program

STEP 2: Declare the value for rows.

STEP 3: Let i and j be an integer number

STEP 4: Repeat step 5 to 8 until all value parsed.

STEP 5: Set i in outer loop using range function,i = rows+1 and rows will initialized to i

STEP 6: Set j in inner loop using range function and i integer will be initialized to j;

STEP 7: Print i until the condition becomes false in inner loop.

STEP 8: Print new line until the condition becomes false in outer loop.

STEP 9: Stop the program.

PROGRAM:

```
n=int(input("Enter thenumbers:"))
for i in range(1,6):
        for j in range(1,i+1):
            print(i,end=" ")
        print()
```

OUTPUT:

Enter the numbers:6

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

RESULT:

Thus the python program to print numbers patterns is executed and verified

EX.NO.3 c)	SCIENTIFIC PROBLEMS USING CONDITIONALS AND ITERATIVE LOOPS. –PYRAMID
DATE:	

AIM:

Write a Python program with conditional and iterative statements for Pyramid Pattern.

ALGORITHM:

STEP 1: Start the program

STEP 2: Read the value for rows.

STEP 3: Let i and j be an integer number.

STEP 4: Repeat step 5 to 8 until all value parsed.

STEP 5: Set i in outer loop using range function, i = 0 to rows ;

STEP 6: Set j in inner loop using range function, j=0 to i+1;

STEP 7: Print * until the condition becomes false in inner loop.

STEP 8: Print new line until the condition becomes false in outer loop.

STEP 9: Stop the program.

PROGRAM:

```
n=int(input("Enter the number:"))
for i in range(0,n):
    for k in range(0,n-i-1):
        print(" ",end="")
   for j in range(0,i+1):
   print("* ",end="")
   print(" ")
```

OUTPUT:

```
Enter the number:5
        *
       * *
      * * *
     * * * *
    * * * * *
```

RESULT:

Thus the python program to print numbers pyramid patterns is executed and verified.

EXPT.NO.4(a) DATE:	**IMPLEMENTING REAL-TIME/TECHNICAL APPLICATIONS USING LISTS, TUPLES-ITEMS PRESENT INA LIBRARY**

AIM :

To Write a python program to implement items present in a library

ALGORITHM:

STEP 1: Start the program

STEP 2: Create the variable inside that variable assigned the list of elements based on the library using List and tuple

STEP 3: Using array index to print the items using list and tupel

STEP 4: To print the result using output statement

STEP 5: Stop the program.

PROGRAM:

LIST

```
library=["books", "author", "barcode" , "price"]
library[0]="python"
print(library[0])
library[1]="ramesh babu"
library[2]=1234
library[3]=230
print(library)
```

TUPLE

```
tup1 = (125,25000 )
tup2 = ('books', 'totalprice')
tup3 = tup1 + tup2;
print(tup3)
```

OUTPUT:

LIST:

```
python
['python', 'ramesh babu', 1234, 230]
```

TUPLE

```
(125, 25000, 'books', 'totalprice')
```

RESULT:

Thus the Python Program is executed successfully and the output is Verified.

EXPT.NO.4 b	IMPLEMENTING REAL-TIME/TECHNICAL APPLICATIONS USING LISTS, TUPLES -COMPONENTS OF A CAR
DATE:	

AIM:

To Write a python program to implement components of a car

ALGORITHM:

STEP 1: Start the program

STEP 2: Create the variable inside that variable assigned the list of elements based on the car using List and tuple

STEP 3:Using array index to print the items using list and tuple

STEP 4:To print the result using output statement

STEP 5: Stop the program

PROGRAM:

LIST:

```
cars = ["Nissan", "Mercedes Benz", "Ferrari", "Maserati", "Jeep", "Maruti
Suzuki"] new_list = []
for i in cars:
   if "M" in i:
      new_list.append(i)
print(new_list)
```

TUPLE:

```
cars=("Ferrari", "BMW", "Audi", "Jaguar")
print(cars)
print(cars[0])
print(cars[1])
print(cars[2])
print(cars[3])
```

OUTPUT:

LIST:

```
['Mercedes Benz', 'Maserati', 'Maruti Suzuki']
```

TUPLE:

```
('Ferrari', 'BMW', 'Audi', 'Jaguar')
Ferrari
BMW
Audi
Jaguar
```

RESULT:

Thus the Python Program is executed successfully and the output is verified.

EXPT.NO.4 C	IMPLEMENTING REAL-TIME/TECHNICAL APPLICATIONS USING LISTS, TUPLES - MATERIALS REQUIRED FOR CONSTRUCTION OF A BUILDING.
DATE:	

AIM:

To Write a python program to implement materials required for construction of building

ALGORITHM:

STEP 1: Start the program

STEP 2: Create the variable and stored the unordered list of elements based on materials Required for construction of building List and tuple

STEP 3: Using array index to print the items using list and tuple

STEP 4: To print the result using output statement

STEP 5: Stop the program

PROGRAM:

```
materials= ["cementbags", "bricks", "sand", "Steelbars",
"Paint"] materials.append("Tiles")
materials.insert(3,"pipes")
materials.remove("sand")
materials[5]="electrical"
print(materials)
```

TUPLE:

```
materials =("cementbags", "bricks", "sand", "Steelbars", "Paint")
print(materials)
print("list of element is=",materials)
print("materials[0]:", materials [0])
print("materials[1:3]:", materials [1:3])
```

OUTPUT:

LIST:

['cementbags', 'bricks', 'pipes', 'Steelbars', 'Paint', 'electrical']

TUPLE:

```
('cementbags', 'bricks', 'sand', 'Steelbars', 'Paint')
list of element is= ('cementbags', 'bricks', 'sand', 'Steelbars', 'Paint')
materials[0]: cementbags
materials[1:3]: ('bricks', 'sand')
```

RESULT:

Thus the Python Program is executed successfully and the output is verified.

EXPT.NO: 5a	IMPLEMENTING REAL-TIME/TECHNICAL APPLICATIONS USING SETS,COMPONENTS OF AN AUTOMOBILE
DATE:	

AIM

To write a python program to implement Components of an automobile using Sets and Dictionaries

ALGORITHM:

STEP 1: Start the program

STEP 2: Create the variable and stored the unordered list of elements based on component of an Automobile.

STEP 3: Using for loop to list the number of elements and using array index to print the items using set and dictionary

STEP 4: To print the result using output statement

STEP 5: Stop the program

PROGRAM:

```
cars = {'BMW', 'Honda', 'Audi', 'Mercedes', 'Honda', 'Toyota', 'Ferrari',
'Tesla'} print('Approach #1= ', cars)
print('Approach
#2') for car in
cars:
    print('Car name = {}'.format(car))
cars.add('Tata')
print('New cars set = {}'.format(cars))
cars.discard('Mercedes')
print('discard() method = {}'.format(cars))
```

OUTPUT:

```
Approach #1=  {'Mercedes', 'Honda', 'BMW', 'Ferrari', 'Audi', 'Toyota', 'Tesla'}
Approach #2
Car name = Mercedes
Car name = Honda
Car name = BMW Car
name = Ferrari Car
name = Audi
Car name = Toyota
Car name = Tesla
New cars set = {'Mercedes', 'Honda', 'BMW', 'Ferrari', 'Tata', 'Audi', 'Toyota', 'Tesla'}
discard() method = {'Honda', 'BMW', 'Ferrari', 'Tata', 'Audi', 'Toyota', 'Tesla'}
```

RESULT:

Thus the program was executed and verified successfully

EXPT.NO: 5B DATE:	IMPLEMENTING REAL-TIME/TECHNICAL APPLICATIONS USING DICTIONARY ELEMENTS OF A CIVIL STRUCTURE

AIM:

To write a Program for Elements of a civil structure using Dictionary

ALGORITHM:

STEP 1: Start the program

STEP 2: Create the variable and stored the unordered list of elements based on Element of civil structure

STEP 3: Using for loop to list the number of elements and using array index to print the items using dictionary

STEP 4: To print the result using output statement

STEP 5: Stop the program

PROGRAM:

```
# Adding elements one at a
time Dict = {}
print("Empty Dictionary: ")
print(Dict)
Dict[0] = 'BRICKS'
Dict[2] = 'CEMENT'
Dict[3] = 'BLUE
PRINT'
print("\nDictionary after adding 3
elements: ") print(Dict)
# Adding set of
values # to a single
Key
Dict['Value_set'] = 4, 5, 6
print("\nDictionary after adding 3
elements: ") print(Dict)
# Updating existing Key's
Value Dict[2] = 'STEEL'
print("\nUpdated key value: ")
print(Dict)
# Adding Nested Key value to Dictionary
Dict[5] = {'Nested': {'1': 'LIME', '2':
'SAND'}}
print("\nAdding a Nested Key:
") print(Dict)
```

OUTPUT:

Empty Dictionary:**{}**

Dictionary after adding 3 elements:
{0: 'BRICKS', 2: 'CEMENT', 3: 'BLUE PRINT'}

Dictionary after adding 3 elements:
{0: 'BRICKS', 2: 'CEMENT', 3: 'BLUE PRINT', 'Value_set': (4, 5, 6)}

Updated key value:
{0: 'BRICKS', 2: 'STEEL', 3: 'BLUE PRINT', 'Value_set': (4, 5, 6)}

Adding a Nested Key:
{0: 'BRICKS', 2: 'STEEL', 3: 'BLUE PRINT', 'Value_set': (4, 5, 6), 5: {'Nested': {'1': 'LIME', '2': 'SAND'}}}

RESULT:

Thus the program was executed and verified successfully

EXPT.NO 6 (a) DATE:	**IMPLEMENTING FACTORIAL PROGRAMS USING FUNCTIONS.**

AIM:

To Write a Python function to calculate the factorial of a number

ALGORITHM:

STEP 1: Get a positive integer input (n) from the user.

STEP 2: check if the values of n equal to 0 or not if it's zero it will return 1

Otherwise else statement can be executed

STEP 3: Using the below formula, calculate the factorial of a number n*factorial(n-1)

STEP 4: Print the output

PROGRAM:

```
def
   factorial(n)
   : if n == 0:
      return
   1 else:
      return n * factorial(n-1)
n=int(input("Input a number to compute the factiorial : "))
print(factorial(n))
```

OUTPUT:

```
Input a number to compute the factiorial: 4
24
```

RESULT:

Thus the program was executed and verified successfully

EXPT.NO 6 b DATE:	IMPLEMENTING PROGRAM LARGEST NUMBER IN A LIST USING FUNCTION

AIM:

To Write a Python program to get the largest number from a list.

ALGORITHM:

STEP 1: Declare a function that will find the largest number

STEP 2: Use max() method and store the value returned by it in a variable

STEP 3: Return the variable

STEP 4: Declare and initialize a list or take input

STEP 5:Call the function and print the value returned by it

PROGRAM:

```
def max_num_in_list( list ):
    max = list[ 0 ]
    for a in list:
        if a > max:
         max = a
    return max
print("The max number is:" ,max_num_in_list([1, 2, -8, 0]))
```

OUTPUT:

The max number is: 2

RESULT:

Thus the program was executed and verified successfully

EXPT.NO.6 (C)	IMPLEMENTING PROGRAMS USING FUNCTIONS – AREA OF SHAPE
DATE	

AIM:

To Write a python program to implement area of shape using functions

ALGORITHM:

STEP 1: Get the input from the user shape's name.

STEP 2: If it exists in our program then we will proceed to find the entered shape's area according to their respective formulas.

STEP 3: If that shape doesn't exist then we will print "Sorry!

STEP 4: Stop the program

PROGRAM:

```
def calculate_area(name):
 name = name.lower()
if name == "rectangle":
      l = int(input("Enter rectangle's length: "))
      b = int(input("Enter rectangle's breadth: "))
      rect_area = l * b
      print(f"The area of rectangle is{rect_area}.")
   elif name == "square":
      s = int(input("Enter square's side length: "))
      sqt_area = s * s
      print(f"The area of square is{sqt_area}.")
   elif name == "triangle":
      h = int(input("Enter triangle's height length: "))
      b = int(input("Enter triangle's breadth length: "))
      tri_area = 0.5 * b * h
      print(f"The area of triangle is{tri_area}.")
   elif name == "circle":
      r = int(input("Enter circle's radius length: "))
      pi = 3.14
      circ_area = pi * r * r
      print(f"The area of triangle is{circ_area}.")
   elif name == 'parallelogram':
      b = int(input("Enter parallelogram's base length: "))
      h = int(input("Enter parallelogram's height length: "))
# calculate area of
      parallelogram para_area
      = b * h
      print(f"The area of parallelogram is{para_area}.")
   else:
     print("Sorry! This shape is not available")
   if name     == "    main   " :
      print("Calculate Shape Area")
shape_name = input("Enter the name of shape whose area you want to find: ")
calculate_area(shape_name)
```

OUTPUT:

Enter the name of shape whose area you want to find:
SQUARE Enter square's side length: 6
The area of square is36.

Enter the name of shape whose area you want to find: rectangle
Enter rectangle's length: 3
Enter rectangle's breadth: 3
The area of rectangle is9.

Enter the name of shape whose area you want to find:
TRIANGLE Enter triangle's height length: 3
Enter triangle's breadth length: 5
The area of triangle is7.5.

Enter the name of shape whose area you want to find:
circle Enter circle's radius length: 2
The area of triangle is12.56.

Enter the name of shape whose area you want to find:
parallelogram Enter parallelogram's base length: 5
Enter parallelogram's height length: 6
The area of parallelogram is30.

RESULT:

Thus the python program to implement area of shape using functions was successfully executed and verified

EXPT.NO.7 a)	IMPLEMENTING PROGRAMS USING STRINGS –REVERSE
DATE	

AIM:

To Write a python program to implement reverse of a string using string functions

ALGORITHM:

STEP 1: Start the program

STEP 2: Using function string values of arguments passed in that function

STEP 3: python string to accept the negative number using slice operation

STEP4: To print the reverse string value by Using reverse method function

STEP5: print the result

PROGRAM:

```
def reverse(str):
    str = str[::-1]
    return str
str = "AMCET"
print ("The original string is :")
print (str)
print ("The reversed string is :")
print (reverse(str))
```

OUTPUT:

```
The original string is :
AMCET
The reversed string is :
TECMA
```

RESULT:

Thus the reverse of a string function python program was executed and successfully verified

EXPT.NO.7 b)	IMPLEMENTING PROGRAMS USING STRINGS -PALINDROME
DATE:	

AIM:

To write a python program to implement palindrome using string functions

ALGORITHM:

STEP 1: start by declaring the isPalindrome() function and passing the string argument.

STEP 2: Then, in the function body,

STEP 3: To get the reverse of the input string using a slice operator – string[::-1].

STEP 4: -1 is the step parameter that ensures that the slicing will start from the end of the string with one step back each time.

STEP 5:if the reversed string matches the input string, it is a palindrome Or else, it is not a palindrome.

PROGRAM:

```
def palindrome(s):
    str=s[::-1]
    return str
s=input("Enter the string:")
if(palindrome(s)==s):
    print("palindrome")
else:
    print("not palindrome")
```

OUTPUT:

```
Enter the string:MADAM
palindrome
```

RESULT:

Thus the program of palindrome by using function in python executed successfully and verified

EXPT.NO: 7 c) DATE:	IMPLEMENTING PROGRAMS USING STRINGS - CHARACTER COUNT

AIM:

To Write a python program to implement Characters count using string functions

ALGORITHM:

STEP 1: user to enter the string. Read and store it in a variable.

STEP 2: Initialize one counter variable and assign zero as its value.

STEP 3: increment this value by 1 if any character is found in the string.

STEP 4: Using one loop, iterate through the characters of the string one by one.

STEP 5: Check each character if it is a blank character or not. If it is not a blank character, increment the value of the counter variable by '1'.

STEP 6: After the iteration is completed, print out the value of the counter.

STEP 7: This variable will hold the total number of characters in the string.

PROGRAM:

```
test="Asan college of engineering and technology"
count=0
for i in test:
   if(i=="e"):
      count=count+1
print("Count of e in test is:",str(count))
```

OUTPUT:

Count of e in test is: 6

RESULT:

Thus the program of character count in string in python was executed successfully and verified

EXPT.NO: 7 (d) DATE:	IMPLEMENTING PROGRAMS USING STRINGS – REPLACING CHARACTERS

AIM:

To write a python program to implement Replacing Characters using string functions

PROCEDURE:

STEP 1: Using string.replace(old, new, count)

STEP 2 : By using string Parameters to change it old – old substring you want to replace.new – new substring which would replace the old substring.

STEP 3: count – (Optional) the number of times you want to replace the old substring with the new substring.

STEP 4: To returns a copy of the string where all occurrences of a substring are replaced with another substring.

PROGRAM:

```
string = "Welcome to  learn,read python programming"
print(string.replace("to", "our"))
print(string.replace("r", "a", 3))
```

OUTPUT:

Welcome our learn,read python programming

Welcome to leaan,aead python paogramming

RESULT:

Thus the program was executed and successfully verified

EXPT.NO: 8 (a) DATE:	**IMPLEMENTING PROGRAMS USING WRITTEN MODULES AND PYTHON STANDARD LIBRARIES–PANDAS**

AIM:

To write a python program to implement pandas modules. Pandas are denote python data structures.

ALGORITHM:

STEP 1: start the program

STEP 2: DataFrame is the key data structure in Pandas. It allows us to store And manipulate tabular data

STEP 4: Python method of DataFrame has data aligned in rows and columns like the SQL table or a spreadsheet database

STEP 3: Series: It is a 1-D size-immutable array like structure

STEP 4: Using max function method to display the maximum ages in a program

STEP 4: List of elements can be displayed by using output statement in pandas.

STEP 5: Stop the program

PROGRAM:

```
import pandas as pd
data={
"Name": ["Price","Allen","Princy"],
"Age": [22, 35, 58],
"Sex": ["male", "male", "female"]
}
df = pd.DataFrame(data)
print(df)
print(df["Age"])
ages = pd.Series([22, 35, 58],name="Age")
print(ages)
df["Age"].max()
print(ages.max()
```

OUTPUT:

```
  Name  Age     Sex
0   Price   22    male
1   Allen   35    male
2  Princy   58  female
0    22
1    35
2    58
Name: Age, dtype: int64
0    22
1    35
2    58
Name: Age, dtype: int64
58
```

RESULT:

Thus the python program to implement pandas modules. Pandas are Denote python data structures was successfully executed and verified

EXPT.NO: 8(b) DATE:	**IMPLEMENTING PROGRAMS USING WRITTENMODULES AND PYTHON STANDARD LIBRARIES– NUMPY**

AIM:

To Write a python program to implement numpy module in python .

ALGORITHM:

STEP 1:Start the program

STEP 2:To create the package of numpy in python and using array index in numpy for numerical calculation

STEP 3:To create the array index inside that index to assign the values in that dimension

STEP 4: Declare the method function of arrange statement can be used in that program

STEP 5: By using output statement we can print the result

PROGRAM:

```
import numpy as np
arr=["code","python","welcome","learn"]
print("Array:",arr)
#to print index value
print("Index value:",arr.index("welcome"))
#to insert
arr.insert(1,"welcome")
print("Insert:",arr)
#to pop
arr.pop(1)
print("pop:",arr)
#to remove
arr.remove("learn")
print("Remove:",arr)
#to extend
arr.extend(["friend","!","welcome"])
print("Extend value:",arr)
#count value
c=arr.count("welcome")
print("countvalue:",c)
```

OUTPUT:

Array: ['code', 'python', 'welcome', 'learn']

Index value: 2

Insert: ['code', 'welcome', 'python', 'welcome', 'learn']

pop: ['code', 'python', 'welcome', 'learn']

Remove: ['code', 'python', 'welcome']

Extend value: ['code', 'python', 'welcome', 'friend', '!', 'welcome']

countvalue: 2

RESULT:

Thus the python program to implement numpy module in python .Numerical python are mathematical calculations are successfully executed and verified

EXPT.NO: 8(c) DATE:	IMPLEMENTING PROGRAMS USING WRITTEN MODULES AND PYTHON STANDARD LIBRARIES–MATPLOTLIB

AIM:

To write a python program to implement matplotolib module in python. Matplotlib python are used to show the visualization entities in python.

ALGORITHM:

STEP1: Start the program

STEP2: Install Matlotlib

STEP3: Import pyplot

STEP4: Define the x and y axis value as list

STEp5: Give a title to your function using.title()

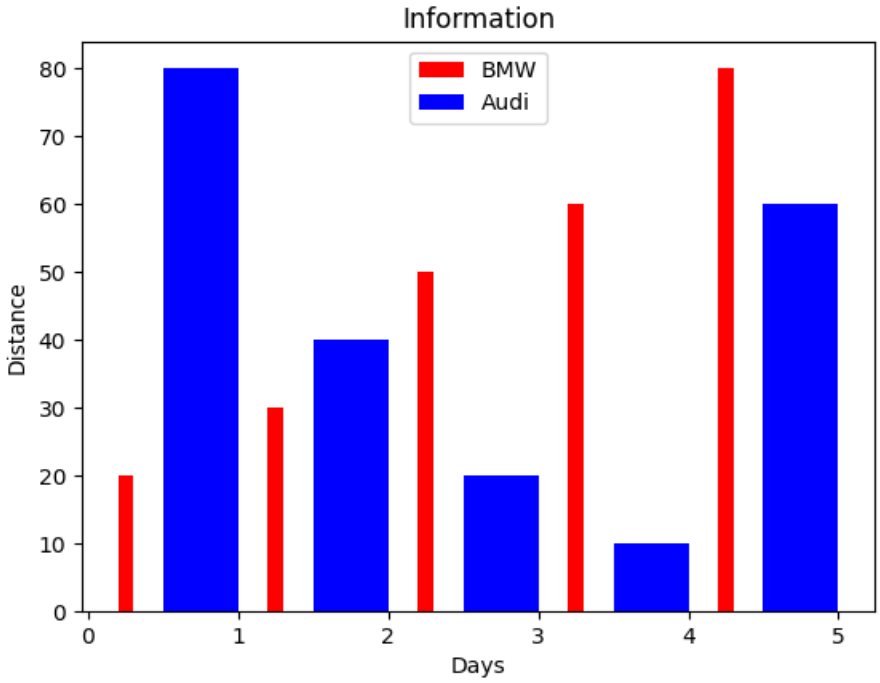
STEP6: View your plot using show() function

STEP7: Stop the prigram.

PROGRAM:

```
import matplotlib.pyplot as plt
plt.bar([0.25,1.25,2.25,3.25,4.25],[20,30,50,60,80],label='BMW',color='r',width=.1)
plt.bar([0.75,1.75,2.75,3.75,4.75],[80,40,20,10,60],label='Audi',color='b',width=.5)
plt.legend()
plt.xlabel('Days')
plt.ylabel('Distance')
plt.title('Information')
plt.show()
```

OUTPUT:



RESULT:

Thus the python program to implement matplotolib module in python. Matplotlib python are used to show the visualization entites in python was successfully executed and verified.

EXPT.NO: 8 (d) DATE:	IMPLEMENTING PROGRAMS USING WRITTEN MODULES AND PYTHON STANDARD LIBRARIES– SCIPY

AIM:

Write a python program to implement scipy module in python.
Scipy python are used to solve the scientific calculations.

ALGORITHM:

STEP1: Start the program.

STEP2: From the Scipy library import special function.

STEP3: Using Special function calculate exponential,sin,and cos values.

STEP4: Print the values.

STEP5: Stop the program.

PROGAM:

```
from scipy import special
a=special.exp10(3)
print(a)
b=special.exp2(2)
print(b)
c=special.sindg(90)
print(c)
d=special.cosdg(45)
print(d)
```

OUTPUT:

```
1000.0
4.0
1.0
0.7071067811865475
```

RESULT:

Thus the python program using scipy library was written and executed successfully.

EXPT.NO: 9 (a) DATE:	**IMPLEMENTING REAL-TIME/TECHNICAL APPLICATIONS USING FILE HANDLING - COPY FROM ONE FILE TO ANOTHER**

AIM:

To Write a python program to implement File Copying

ALGORITHM:

STEP1: Start the program

STEP2: Import the copy file module from shutill

STEP3: Enter the source and destination file names.

STEP4: Copy the content of the source file to destination file using copyfile module.

STEP5: Stop the program.

PROGRAM:

```
from shutil import copyfile
sourcefile=input("Enter the source file name:")
destinationfile=input("Enter the destination file name: ")
copyfile(sourcefile,destinationfile)
print("File copied sucessfully:")
print("content of destination file:")
fileread=open(destinationfile,"r")
print(fileread.read())
fileread.close()
```

OUTPUT:

```
Enter the source file name:first.txt
Enter the destination file name: second.txt
File copied sucessfully:
content of destination file:
Asan Memorial College Of Engineering & Technology
```

RESULT:

Thus the a python program using File Handling was written and executed Successfully .

EXPT.NO: 9 (b) DATE:	IMPLEMENTING REAL-TIME/TECHNICAL APPLICATIONS USING FILE HANDLING WORD COUNT

AIM:

To Write a python program to implement word count in File operations in python

ALGORITHM:

STEP1: Start the program

STEP 2: Open and create the txt file with some statements

STEP 3: To save that file with the extension of txt file

STEP 4: Count the the lenghth of word in a file.

STEP5: Display the word count in a target file

PROGRAM:

```
i=0
f=open("F:/new.txt","r")
data=f.read()
lines=data.split()
i+=len(lines)
print("Number of word in the file")
print(i)
```

OUTPUT:

```
Number of word in the file
6
```

RESULT:

Thus word count in File operations in python was executed successfully and verified

EXPT.NO: 9 (c) DATE:	IMPLEMENTING REAL-TIME/TECHNICAL APPLICATIONS USING FILE HANDLING - LONGEST WORD

AIM:

To Write a python program to implement longest word in File operations

ALGORITHM:

STEP1: Start the program

STEP2: Open and create the txt file with some statements

STEP3: save that file with the extension of txt file

STEP4: Now to count the longest of word in a file.

STEP5: To display the longest word in a target file

PROGRAM:

```
def longest_word(filename):
    with open(filename,'r')as infile:
        words=infile.read().split()
        max_len=len(max(words,key=len))
    return[word for word in words if len(word)==max_len]
print(longest_word('new.txt'))
```

OUTPUT:

['Engineering']

RESULT:

Thus the python program to implement longest word in File operations was executed successfully verified.

EXPT.NO: 10 (a) DATE:	IMPLEMENTING REAL-TIME/TECHNICAL APPLICATIONS USING EXCEPTION HANDLING.- DIVIDE BY ZERO ERROR.

AIM

To Write a exception handling program using python to depict the divide by zero error.

PROCEDURE:

STEP 1: start the program

STEP 2: The **try** block tests the statement of error.

STEP3: The **except** block handle the error.

STEP4: A single try statement can have multiple except statements.

STEP 5: To create the two identifier name and enter the values

STEP 6: By using division operation and if there is any error in that try block raising the error in that block

STEP 7: otherwise it display the result

STEP8: Stop the program

PROGRAM:

```
try:
   a=int(input("Enter a:"))
   b=int(input("Enter b:"))
   c=a/b
   print("Result=",c)
except Exception:
    print("con't divided by zero")
```

OUTPUT:

```
Enter a:20
Enter b:0
con't divided by zero


Enter a:20
Enter b:5
Result= 4
```

RESULT:

Thus the exception handling program using python to depict the divide by zero error. was successfully executed and verified

EXPT.NO: 10 (b) DATE:	IMPLEMENTING REAL-TIME/TECHNICAL APPLICATIONS USING EXCEPTION HANDLING.- CHECK VOTERS ELIGIBILITY

AIM:

To Write a exception handling program using python to depict the voters eligibility

ALGORITHM:

STEP1: Start the program

STEP2: Get the input from the user

STEP3: If the entered data is not integer,throw an exception

STEP4: If no exception return the message

STEP5: Stop the program

PROGRAM:

```
 try:
   age=int(input("Enter your age:"))
   if age>18:
      print("Eligible to vote")
   else:
      print("Not eligible to vote")
except:
   print("Age must be valid number")
```

OUTPUT:

```
Enter your age:32
Eligible to vote
```

RESULT:

Thus the exception handling program using python to depict the voters eligibility was successfully executed and verified.

EXPT.NO: 10 (c)	IMPLEMENTING REAL-TIME/TECHNICAL APPLICATIONS USING EXCEPTION HANDLING.- STUDENT MARK RANGE VALIDATION
DATE:	

AIM:

To Implementing real-time/technical applications using Exception handling.- student mark range validation

ALGORITHM:

STEP 1: Start the program

STEP 2: Get the input from the user

STEP 3: if statement can be used to check the mark range in the program

STEP 4: Given data is not valid it will throw an exception in the process

STEP 5: Stop the program

PROGRAM:

```
try:
   mark=int(input("enter your mark="))
   if mark>=35 and mark<101:
      print("pass and your mark is valid")
   else:
      print("fail and your mark is valid")
except ValueError:
   print("mark must be avalid number")
except IOError:
   print("enter correct valid mark")
except:
   print("An error occurred")
```

OUTPUT:

```
enter your mark=45
pass and your mark is valid
```

RESULT:

Thus the real-time/technical applications using Exception handling.- student mark range validation was successfully executed and verified

EXPT.NO: 11 DATE:	EXPLORING PYGAME TOOL.

AIM:

To Write a python program to implement pygame.

ALGORITHM:

STEP1: Start the program

STEP2: Set up the drawing window

STEP3: Run until the user ask to quit

STEP4: Fill the background with white

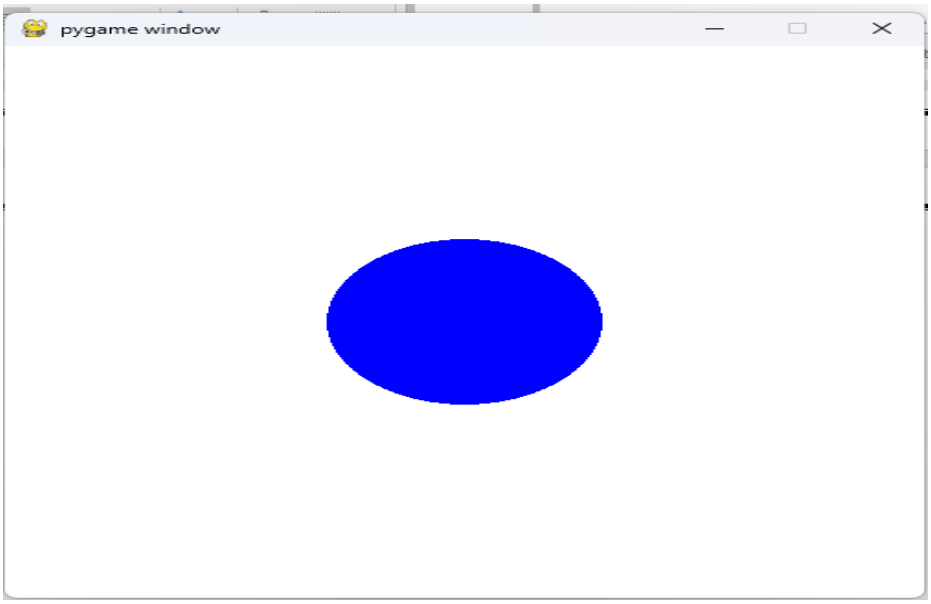
STEP5: Flip the display

STEP6: Quit pygame.

PROGRAM:

```
#import and initialize the pygame library
import pygame
pygame.init()
#set up drawing window
screen=pygame.display.set_mode([500,500])
#Run untill user quit
running=True
while running:
    #did user click the window close button
    for event in pygame.event.get():
        if event.type==pygame.QUIT:
            running=false
    #Fill the background white
    screen.fill((255,255,255))
    #Draw blue circle in center
    pygame.draw.circle(screen,(0,0,255),(250,250),75)
    #Flip the display
    pygame.display.flip()
```

OUTPUT:



RESULT:

Thus the python program to implement pygame was successfully executed and verified.

EXPT.NO: 12 DATE:	DEVELOPING A GAME ACTIVITY USING PYGAME LIKE BOUNCING BALL

AIM:

To Write a python program to implement bouncing balls using pygame tool

ALGORITHM:

STEP1: Start the program

STEP 2: The **pygame.display.set_mode()** function returns the surface object for the window This function accepts the width and height of the screen as arguments.

STEP 3: To set the caption of the window, callthe **pygame.display.set_caption()** function.

opened the image using the **pygame.image.load()** method and set the ball rectangle area boundary using the **get_rect()** method.

STEP 4: The **fill()** method is used to fill the surface with a background color.

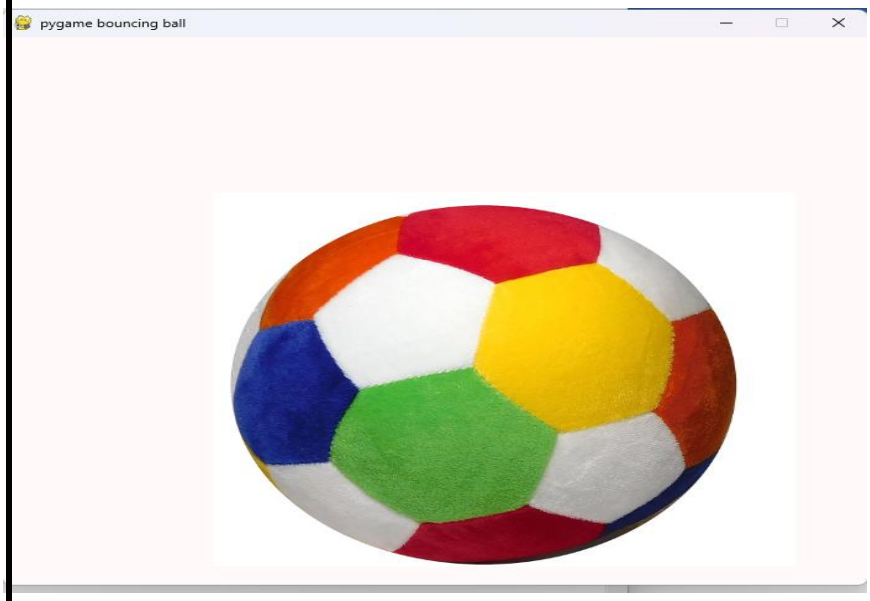
STEP 5: pygame **flip()** method to make all images visible.

STEP 6: Stop the program

PROGRAM:

```
import sys,pygame
pygame.init()
speed=[2,2]
color=[255,250,250]
width=700
height=600
scr    screen.fill(color)
een=pygame.display.set_mode((width,height))
pygame.display.set_caption("pygame bouncing ball")
ball=pygame.image.load("ball.png")
rect_boundry=ball.get_rect()
while 1:
    for event in pygame.event.get():
        rect_boundry=rect_boundry.move(speed)
        if rect_boundry.left<0 or rect_boundry.right>width:
            speed[0]=-speed[0]
        if rect_boundry.top<0 or rect_boundry.bottom>height:
            speed[1]=-speed[1]
    screen.blit(ball,rect_boundry)
    pygame.display.flip()
```

OUTPUT:



RESULT:

Thus the python program to implement bouncing balls using pygame tool was executed successfully and verified