Question 1. Write a program illustrating class definition and accessing class members.

In [8]:

```
class demo:
    '''This is a Demo Class to demontrate class definition'''
    variable="Rakshit"        # Class Variable

    def __init__(self,name=None):
        self.name=name      # This is an instance variable
        print("__init__ function called successfully")

    @staticmethod
    def static_method():
        print("Static Method called Successfully")

    @classmethod
    def class_method(cls):
        print("Class Method called Successfully")

# Now creating the object of the class demo
obj1=demo()                              #Object created successfully

# Now calling the static method
demo.static_method()

# Now calling the class Method
demo.class_method()

# Now accessing the class variable
print("The class variable is:",demo.variable)

# Now creating a second object of the demo class and passing a value to it
obj2= demo("Sangal")

# Now calling the instance variable by name of the object
print("The instance variable of the class is:",obj2.name)

# Now printing the documented String of the demo class
print("Docstring: \n",demo.__doc__)
```

```
__init__ function called successfully
Static Method called Successfully
Class Method called Successfully
The class variable is: Rakshit
__init__ function called successfully
The instance variable of the class is: Sangal
Docstring:
 This is a Demo Class to demontrate class definition
```

Question 2. Write a program to implement default constructor, parameterized constructor, and destructor.

In [11]:

```
class cons_dis:
    '''This class is created to show the use of constructor and destructor'''
    variable=10

    def __init__(self):
        '''This is a default constructoras it doesn't have any parameter.
           Python always searches for the last constructor.
           As it supports constructor overriding'''

    print("Default Constructor is called")

    def __init__(self,var):
        ''' This is a parameterized constructor'''
        cons_dis.variable+=10
        self.var=var
        print("Value of Instance Variable:",self.var)
        print("Value of Class Variable:",cons_dis.variable)

    # Implementing the destructor function
    def __del__(self):
        '''This is a distructor'''
        cons_dis.variable-=10
        print("Object with value {} is going out of scope",{self.var})

# Creating a object of cons_dis class
obj1=cons_dis(10)

# Creating another object of cons_dis class
obj2=cons_dis(20)

#Invoking the Destructor
del obj1
del obj2

# Checking if obj1 and obj2 exist or not!
print(obj1)
```

```
Default Constructor is called
Value of Instance Variable: 10
Value of Class Variable: 20
Value of Instance Variable: 20
Value of Class Variable: 30
Object with value {} is going out of scope {10}
Object with value {} is going out of scope {20}


---------------------------------------------------------------------------
NameError                                 Traceback (most recent call last)
<ipython-input-11-47f3bb7f5c5f> in <module>
     34
     35 # Checking if obj1 and obj2 exist or not!
---> 36 print(obj1)

NameError: name 'obj1' is not defined
```

Question 3. Create a Python class named Rectangle constructed by a length and width:

a.) Create a method called area which will compute the area of a rectangle

In [12]:

```
class rectangle:

    def __init__(self,l,b):
        self.l=l
        self.b=b

    def area(self):
        print("Area of Rectangle=",(self.l*self.b))

l=int(input("Enter the length of the Rectangle:"))
b=int(input("Enter the breadth of the Rectangle:"))
obj=rectangle(l,b)
obj.area()
```

```
Enter the length of the Rectangle:20
Enter the breadth of the Rectangle:30
Area of Rectangle= 600
```

Question 4. Create a class called Numbers, which has a single class attribute called MULTIPLIER, and a constructor which takes the parameters x and y (these should all be numbers):

a.) Write an instance method called add which returns the sum of the attributes x and y.

b.) Write a class method called multiply, which takes a single number parameter a and returns the product of a and MULTIPLIER.

c.) Write a static method called subtract, which takes two number objects, b and c, and returns b - c.

d.) Write a method called value which returns a tuple containing the values of x and y.

In [15]:

```
class numbers:
    multiplier=int(input("Enter the MULTIPLIER"))

    def __init__(self):
        self.x=int(input("Enter the first number:"))
        self.y=int(input("Enter the second number:"))

    def add(self):
        '''This is an Instance Method'''

        return self.x+self.y

    @classmethod
    def multiply(cls,a):
        '''This is a Class Method'''
        cls.a=a
        return cls.a*numbers.multiplier

    @staticmethod
    def subtract(e,f):
        '''This is a Static Method'''
        return e-f

    def value(self):
        '''This is an Instance Method'''
        return (self.x,self.y)

num=numbers()
print("Addition=",num.add())
print("Multiplication=",numbers.multiply(20))
print("subtraction=",numbers.subtract(30,19))
print("Values of x and y:",num.value())
```

```
Enter the MULTIPLIER-1
Enter the first number:20
Enter the second number:30
Addition= 50
Multiplication= -20
subtraction= 11
Values of x and y: (20, 30)
```

Question 5. Create a class named as Student to store the name and marks in three subjects. Use List to store the marks:

a.) Write an instance method called compute to compute total marks and average marks of a student.

b.) Write a method called display to display student information.

In [51]:

```
class student_details:
    def __init__(self):
        self.name=input("Enter The Name of the Student:")
        self.marks=[]

    def getdata(self):
        for i in range(3):
            m=int(input("Enter the marks of %s in subject %d :"%(self.name,i+1)))
            self.marks.append(m)
            print("\n")

    def compute(self):
        m=self.marks
        self.total_marks=m[0]+m[1]+m[2]
        self.average_marks=self.total_marks/len(m)

    def display(self):
        print("STUDENT DETAILS:")
        print("Name Of Student: ",self.name)
        print("Total Marks: ",self.total_marks)
        print("Average Marks: ",self.average_marks)

s=student_details()
s.getdata()
s.compute()
s.display()
```

```
Enter The Name of the Student:Rakshit Sangal
Enter the marks of Rakshit Sangal in subject 1 :99


Enter the marks of Rakshit Sangal in subject 2 :93


Enter the marks of Rakshit Sangal in subject 3 :98


STUDENT DETAILS:
Name Of Student:  Rakshit Sangal
Total Marks:  290
Average Marks:  96.66666666666667
```

Question 6. Create a class Employee that keeps a track of the number of employees in an organization and also stores their name, designation and salary details:

a.) Write a method called getdata to take input (name, designation, salary) from user.

b.) Write a method called average to find average salary of all the employees in the organization.

c.) Write a method called display to print all the information of an employee.

In [55]:

```
class Employee:
    c,s=0,0
    avg=0
    def __init__(self):
        pass

    def getdata(self):
        global name,desig,contact,sal
        name = input("Enter the Name of the Employee:")
        desig=input("Enter the Designation of the Employee:")
        contact=int(input("Enter the Contact Number"))
        sal=float(input("Enter the Salary of the Employee:"))
        Employee.c+=1

    def Average(self):
        Employee.s+=sal
        Employee.avg=(Employee.s/Employee.c)

    def Display(self):
        print("\n Name = {} , Designation = {} ,Contact Number = {}, Salary = {} \n ".forma


n=int(input("Enter the number of Employees:"))
for i in range(n):
    obj=Employee()
    obj.getdata()
    obj.Average()
    obj.Display()
print("Average Salary=",Employee.avg)
```

```
Enter the number of Employees:2
Enter the Name of the Employee:Rakshit Sangal
Enter the Designation of the Employee:Senior Developer
Enter the Contact Number9045068087
Enter the Salary of the Employee:97000

 Name = Rakshit Sangal , Designation = Senior Developer ,Contact Number = 90
45068087, Salary = 97000.0

Enter the Name of the Employee:Nikunj
Enter the Designation of the Employee:Manager
Enter the Contact Number9045068087
Enter the Salary of the Employee:49000

 Name = Nikunj , Designation = Manager ,Contact Number = 9045068087, Salary
= 49000.0

Average Salary= 73000.0
```

Question 7. Create a Python class named Circle constructed by a radius. Use a class variable to define the value of constant PI:

a.) Write two methods to be named as area and circum to compute the area and the perimeter of a circle respectively by using class variable PI.

b.) Write a method called display to print area and perimeter.

In [57]:

```
class circle:
    pi=3.14159
    def __init__(self,radius=None):
        #Constructed by radius
        self.radius=float(input("Enter length of radius in centimeters:"))

    def circum(self):
        return 2*circle.pi*self.radius

    def area(self):
        return circle.pi*self.radius*self.radius

    def display(self):
        print("The Area of Circle =",self.area(),"sq cm")
        print("The Circumference of Circle =",self.circum(),"cm")

#Creating object of Circle class
obj=circle()
#Calling the display Method
obj.display()
```

```
Enter length of radius in centimeters:9
The Area of Circle = 254.46879 sq cm
The Circumference of Circle = 56.54862 cm
```

Question 8. Create a class called String that stores a string and all its status details such as number of uppercase letters, lowercase letters, vowels,consonants and space in instance variables:

a.) Write methods named as count_uppercase, count_lowercase, count_vowels, count_consonants and count_space to count corresponding values.

b.) Write a method called display to print string along with all the values computed by methods in (a)

In [61]:

```
class String:
    def __init__(self):
        self.vowels=0
        self.consonants=0
        self.spaces=0
        self.uppercase=0
        self.lowercase=0
        self.string=str(input("Enter a String: "))

    def count_vowels(self):
        for i in self.string:
            if i in ('a','e','i','o','u','A','E','I','O','U'):
                self.vowels+=1

    def count_consonants(self):
        for i in self.string:
            if i not in ('a','e','i','o','u','A','E','I','O','U',' '):
                self.consonants+=1

    def count_uppercase(self):
        for i in self.string:
            if i.isupper():
                self.uppercase+=1

    def count_lowercase(self):
        for i in self.string:
            if i.islower():
                self.lowercase+=1

    def count_spaces(self):
        for i in self.string:
            if i==' ':
                self.spaces+=1

    def compute(self):
        self.count_vowels()
        self.count_consonants()
        self.count_spaces()
        self.count_uppercase()
        self.count_lowercase()

    def display(self):
        print('Number of Vowels=',self.vowels)
        print('Number of Consonants=',self.consonants)
        print('Number of Spaces=',self.spaces)
        print('Number of Uppercase Letters=',self.uppercase)
        print('Number of Lowercase Letters=',self.lowercase)

s=String()
s.compute()
s.display()
```

```
Enter a String: PythOn ProgRamminG Is A Fun
Number of Vowels= 7
Number of Consonants= 16
Number of Spaces= 4
Number of Uppercase Letters= 8
Number of Lowercase Letters= 15
```

Question 9. Write a program that has a class called Fraction with attributes numerator and denominator:

a.) Write a method called getdata to enter the values of the attributes.

b.) Write a method show to print the fraction in simplified form.

In [70]:

```
class fraction:
    def get_data(self):
        self.num=int(input("Numenator: "))
        self.deno=int(input("Denomenator: "))
        if self.deno==0:
            print("ZeroDivisonEror")
            exit()

    def GCD(self,a,b):
        if b==0:
            return a
        else:
            return self.GCD(b,a%b)

    def simplify(self):
        common_divisor=self.GCD(self.num,self.deno)
        self.num/=common_divisor
        self.deno/=common_divisor

    def show_data(self):
        self.simplify()
        print("Simplified Form=",int(self.num),"/",int(self.deno))

obj=fraction()
obj.get_data()
obj.show_data()
```

```
Numenator: 28
Denomenator: 21
Simplified Form= 4 / 3
```

Question 10. Write a method show to print the fraction in simplified form:

a.) Write a method called insert_element that takes values from user.

b.) Write a class method called find_max to find and print largest value in the list.

In [86]:

```
class number:
    def __init__(self):
        self.mylist=[]

    def insert_element(self):
        value=int(input("Enter the number of terms you wanted to input:"))
        for i in range(value):
            values=int(input("Enter the number:"))
            self.mylist.append(values)

    def find_max(self):
        max=self.mylist[1]
        for i in self.mylist:
            if (i>max):
                max=i
        print("The Maxium Value in the list is=",max)

#Creating Object of Number class
num=number()
#calling the insert element method
num.insert_element()
#Now calling the findmax method
num.find_max()
```

```
Enter the number of terms you wanted to input:3
Enter the number:33
Enter the number:45
Enter the number:89
The Maxium Value in the list is= 89
```

Question 11. Write a program that has a class Point with attributes x and y:

a.) Write a method called midpoint that returns a midpoint of a line joining two points.

b.) Write a method called length that returns the length of a line joining two points.

In [91]:

```
class point:
    def __init__(self):
        self.x1=int(input("Enter the value of x1: "))
        self.y1=int(input("Enter the value of y1: "))
        self.x2=int(input("Enter the value of x2: "))
        self.y2=int(input("Enter the value of y2: "))

    def midself(self):
        return (self.x1+self.x2)/2,(self.y1+self.y2)/2

    def length(self):
        return (((self.x2-self.x1)*2)+((self.y2-self.y1)*2))*0.5

obj=point()
print("The Mid point of line =",obj.midself())
print("The Length of line = %.2f" %obj.length(),"units")
```

```
Enter the value of x1: 45
Enter the value of y1: 67
Enter the value of x2: -34
Enter the value of y2: -96
The Mid point of line = (5.5, -14.5)
The Length of line = -242.00 units
```

Question 12. Create a class called Complex. Write a menu driven program to read, display, add and subtract two complex numbers by creating corresponding instance methods.

In [97]:

```
class Complex:
    def read(self):
        self.real1=int(input("Enter real part 1: "))
        self.comp1=int(input("Enter comp part 1: "))
        self.real2=int(input("Enter real part 2: "))
        self.comp2=int(input("Enter comp part 2: "))

    def display(self):
        print("First Complex number C1=",str(self.real1)+"+"+str(self.comp1)+"i")
        print("Second Complex number C2=",str(self.real2)+"+"+str(self.comp2)+"i")
        print("\n")

    def add(self):
        real_sum=self.real1+self.real2
        comp_sum=self.comp1+self.comp2
        print("The Sum of Complex Number is :",str(real_sum)+"+"+str(comp_sum)+"i")
        print("\n")

    def subtract(self):
        real_sub=self.real1-self.real2
        comp_sub=self.comp1-self.comp2
        if comp_sub<0:
            print("The Subtraction of Complex Number is :",str(real_sub)+str(comp_sub)+"i")
        else:
            print("The Subtraction of Complex Number is :",str(real_sub)+"+"+str(comp_sub)+
        print("\n")

C=Complex()
while True:
    print("* "*17)
    print('* 1. Read Complex Number          *')
    print('* 2. Display Complex Number       *')
    print('* 3. Add Two Complex Number       *')
    print('* 4. Subtract Two Complex Number  *')
    print(" *"*17)
    choice=int(input("Enter Your choice from above or Press any key to terminate: "))
    print("\n")

    if choice==1:
        C.read()

    elif choice==2:
        C.display()

    elif choice==3:
        C.add()

    elif choice==4:
        C.subtract()

    else:
        print("Oops! It is an invalid input. Logging Off....")
        break
```

```
* * * * * * * * * * * * * * * * * *
* 1. Read Complex Number          *
* 2. Display Complex Number       *
* 3. Add Two Complex Number       *
```

```
* 4. Subtract Two Complex Number  *
 * * * * * * * * * * * * * * * * * *
Enter Your choice from above or Press any key to terminate: 1


Enter real part 1: 4
Enter comp part 1: 8
Enter real part 2: 9
Enter comp part 2: 19
* * * * * * * * * * * * * * * * * *
* 1. Read Complex Number          *
* 2. Display Complex Number       *
* 3. Add Two Complex Number       *
* 4. Subtract Two Complex Number  *
 * * * * * * * * * * * * * * * * * *
Enter Your choice from above or Press any key to terminate: 5


Oops! It is an invalid input. Logging Off....
```

Question 13. Write a Program to illustrate the use of *str*(), *repr*(), *new*, *doc*, *dict*, *name* and *bases* methods.

In [101]:

```
class Builtin:
    '''I am the documentation of class'''
    def __new__(cls,var1,var2):           #Use of static method __new()
        print("new() magic method is called")
        inst=object.__new__(cls)
        return inst

    def __init__(self,var1,var2):           #Use of __init_ and _new() method
        print("init() magic method is called")
        self.var1=var1
        self.var2=var2

    def __repr__(self):
        return "var1=%s,var2=%s"%(self.var1,self.var2)

    def __str__(self):
        return "var1 is %s and var2 is %s"%(self.var1,self.var2)

B=Builtin(10,20)
print("I am __str() method",B)           #This will call __str() method.
print("I am __repr() method",[B])        #This will call __repr() method.
print("Hello, i am __doc_ method Called by object.__doc__: ",B.__doc__)
print("Hello, i am __dict_ method Called by object.__dict__: ",B.__dict__)
print("Hello, i am __name_ method Called by class.__name__: ",Builtin.__name__)
print("Hello, i am __bases_ method Called by class.__bases__: ",Builtin.__bases__)
```

```
new() magic method is called
init() magic method is called
I am __str() method var1 is 10 and var2 is 20
I am __repr() method [var1=10,var2=20]
Hello, i am __doc_ method Called by object.__doc__:  I am the documentation
of class
Hello, i am __dict_ method Called by object.__dict__:  {'var1': 10, 'var2':
20}
Hello, i am __name_ method Called by class.__name__:  Builtin
Hello, i am __bases_ method Called by class.__bases__:  (<class 'object'>,)
```

Question 14. Create a BankAccount class. Your class should support the following methods:

a.) init(self, account_no)

b.) deposit (self, account_no, amount)

c.) withdraw (self, account_no, amount)

d.) get_balance (self, account_no)

In [81]:

```
class bank_account:
    def __init__(self,account_no):
        self.account_no=account_no
        self.Balance=0
        print("New Account Created Successfully")

    def deposit(self,account_no,amount):
        self.Balance+=amount
        print("Your Updated Account Balace After deposit is=INR",self.Balance,"/-")

    def withdraw(self,account_no,amount):
        if (amount>self.Balance):
            print("You Don't have sufficient Balance")
            print("You have only INR %d /- rupees left in your Account"%self.Balance)
        else:
            self.Balance-=amount
            print('New Balance after withdrawl is=Rs',self.Balance)

    def get_balance(self,account_no):
        print("Current Account balance is=Rs",self.Balance)

customer=bank_account(3014522574210)
customer.deposit(3014522574210,151500)
customer.withdraw(3014522574210,51500)
customer.get_balance(3014522574210)
```

```
New Account Created Successfully
Your Updated Account Balace After deposit is=INR 151500 /-
New Balance after withdrawl is=Rs 100000
Current Account balance is=Rs 100000
```

Question 15. Write a program to illustrate the use of following built-in methods:

a.) hasattr(obj,attr)

b.) getattr(object, attribute_name [, default])

c.) setattr(object, name, value)

d.) delattr(class_name, name)

In [78]:

```
class BuiltinMethod():
    def __init__(self,var):
        self.var=var

    def Display(self):
        print("Var is = ",self.var)

obj=BuiltinMethod(10)
obj.Display()

#Using hasattr(object,attr) method
print("Check if object has attribute var ------>",hasattr(obj,'var'))

#Using getattr(object,attributename) method
getattr(obj,'var')

#Using setattr(object,name,value) method
setattr(obj,'var',50)
print("After setting value,var is = ",obj.var)

#We can also change the name in setattr
setattr(obj,'count',50)
print("New Variable count is created and the value is = ",obj.count)

#Using delattr(class_name,name) method
delattr(obj,'var')
print("After deleting,var is = ",obj.var) #It will give AttributeError
```

```
Var is =  10
Check if object has attribute var ------> True
After setting value,var is =  50
New Variable count is created and the value is =  50

---------------------------------------------------------------------------
AttributeError                            Traceback (most recent call last)
<ipython-input-78-86d3e00a0d10> in <module>
     25 #Using delattr(class_name,name) method
     26 delattr(obj,'var')
---> 27 print("After deleting,var is = ",obj.var) #It will give AttributeErr
or

AttributeError: 'BuiltinMethod' object has no attribute 'var'
```

Question 16. Write a program to create class Employee. Display the personal information and salary details of 5 employees using single inheritance.

In [75]:

```
class employee:
    empCount=0
    def __init__(self,name,deg,dep,contact,sal):
        self.name=name
        self.deg=deg
        self.dep=dep
        self.contact=contact
        self.sal=sal
        Employee.empCount+=1

class information(employee):
    def personal(self):
        print("Personal information of Employee "+str(Employee.empCount))
        print(""*20)
        print("Name of Employee: ",self.name)
        print("Degination: ",self.deg)
        print("Department: ",self.dep)
        print("Contact: ",self.contact)
        print("Salary :",self.sal)


info1=information("Rakshit Sangal","Senior Developer","Artificial Intelligence","9045068087
info1.personal()

info2=information("Deepak Garg","Chartered Accountant","Accounts","9045068087",50000)
info2.personal()

info3=information("Nikunj Garg","Junior Developer","Programming","9045068087",55000)
info3.personal()

info4=information("Abhishek","DS","Data Management","9045068087",65000)
info4.personal()

info5=information("Harsh Gupta","AI","Service","9045068087",75000)
info5.personal()

print("There are total "+str(Employee.empCount)+" Employee")
```

```
Personal information of Employee 6

Name of Employee:  Rakshit Sangal
Degination:  Senior Developer
Department:  Artificial Intelligence
Contact:  9045068087
Salary : 70000
Personal information of Employee 7

Name of Employee:  Deepak Garg
Degination:  Chartered Accountant
Department:  Accounts
Contact:  9045068087
Salary : 50000
Personal information of Employee 8

Name of Employee:  Nikunj Garg
Degination:  Junior Developer
Department:  Programming
Contact:  9045068087
```

```
Salary : 55000
Personal information of Employee 9

Name of Employee:  Abhishek
Degination:  DS
Department:  Data Management
Contact:  9045068087
Salary : 65000
Personal information of Employee 10

Name of Employee:  Harsh Gupta
Degination:  AI
Department:  Service
Contact:  9045068087
Salary : 75000
There are total 10 Employee
```

Question 17. WAP that extends the class Employee. Derive two classes Manager and Team Leader from Employee class. Display all the details of the employee working under a particular Manager and Team Leader.

In [8]:

```
class Employee:
    empCount=0
    def __init__(self,name=None,age=None,exp=None,dept=None,contact=None,qual=None):
        self.name=input("Enter name of Employee: ")
        self.age=int(input("Enter Age of Employee: "))
        self.exp=int(input("Enter experience of Employee in Year: "))
        self.dept=input("Enter department of Employee: ")
        self.contact=input("Enter Contact Number: ")
        self.qual=input("Enter Highest Qualification: ")
        Employee.empCount+=1

    def display(self):
        print("Name of Employee : ",self.name)
        print("Age : ",self.age)
        print("Experince: ",self.exp)
        print("Department: ",self.dept)
        print("Contact: ",self.contact)
        print("Qualification: ",self.qual)
        print("\n")

class Manager(Employee):
    def __init__(self,name1=None):
        self.name1=input("Enter the name of Manager: ")
        Employee.__init__(self)

    def displayData(self):
        print("Details of Employee Working Under Manager Mr "+self.name1+" are : ")
        print("*"*18)
        super().display()
        print(""*18,"\n")

class Team_Leader(Employee):
    def __init__(self,name2=None):
        self.name2=input("Enter the name of Team Leader: ")
        Employee.__init__(self)

    def DisplayData(self):
        print("Details of Employee Working Under Team Leader Mr "+self.name2+" are : ")
        print("*"*18)
        super().display()
        print("*"*18,"\n")

t=Team_Leader()
t.DisplayData()
m=Manager()
m.displayData()
print("The Total number of Employee is: ",Employee.empCount)
```

```
Enter the name of Team Leader: Rakshit Sangal
Enter name of Employee: Aggarwal
Enter Age of Employee: 38
Enter experience of Employee in Year: 8
Enter department of Employee: Developer
Enter Contact Number: 9045068087
Enter Highest Qualification: M.Tech
Details of Employee Working Under Team Leader Mr Rakshit Sangal are :
******************
Name of Employee :  Aggarwal
```

```
Age :  38
Experince:  8
Department:  Developer
Contact:  9045068087
Qualification:  M.Tech


**********


Enter the name of Manager: Celestial Interface
Enter name of Employee: Nikunj
Enter Age of Employee: 19
Enter experience of Employee in Year: 1
Enter department of Employee: Hacking
Enter Contact Number: 9045068087
Enter Highest Qualification: Intermediate
Details of Employee Working Under Manager Mr Celestial Interface are :
**********
Name of Employee :  Nikunj
Age :  19
Experince:  1
Department:  Hacking
Contact:  9045068087
Qualification:  Intermediate



The Total number of Employee is:  2
```

Question 18. Write a program that has a class Point. Define another class Location which has two objects (Location and destination) of class Point. Also, define a function in Location that prints the reflection on the y-axis.

In [9]:

```
class Point:
    def __init__(self,x,y):
        self.x=x
        self.y=y

    def get(self):
        return self.x,self.y

class Location(Point):
    def __init__(self,x1,y1,x2,y2):
        self.Location=Point(x1,y1)
        self.Destination=Point(x2,y2)

    def show(self):
        print("Location = ",self.Location.get())
        print("Destination = ",self.Destination.get())

    def Reflection(self):
        self.Destination.x=-self.Destination.x
        print("Reflection = ",self.Destination.x,self.Destination.y)

obj=Location(1,2,3,4)
obj.show()
obj.Reflection()
```

```
Location =  (1, 2)
Destination =  (3, 4)
Reflection =  -3 4
```

Question 19. WAP that create a class Student having attribute as name and age and Marks class inheriting Students class with its own attributes marks1, marks2 and marks3 as marks in 3 subjects. Also, define the class Result that inherits the Marks class with its own attribute total. Every class has its own display() method to display the corresponding details. Use init() and super() to implement the above classes.

In [13]:

```
class Student:
    def __init__(self,name=None,age=None):
        self.name=input("Enter Name of student: ")
        self.age=int(input("Enter age: "))

    def display(self):
        print("Name of Student : ",self.name)
        print("Age : ",self.age)

class Marks(Student):
    def get_marks(self,m1=None,m2=None,m3=None):
        self.m1=int(input("Enter the marks in Subject 1:"))
        self.m2=int(input("Enter the marks in Subject 2:"))
        self.m3=int(input("Enter the marks in Subject 3:"))

    def display(self):
        super().display()
        print("Marks in Subject 1:",self.m1)
        print("Marks in Subject 2:",self.m2)
        print("Marks in Subject 3:",self.m3)

class Result(Marks):
    def display(self,total=None,avg=None):
        self.total=self.m1+self.m2+self.m3
        self.avg=(self.total)/3
        print("-"*20)
        super().display()
        print("Total Marks Obtained By Student is = ",self.total)
        print("Average Marks Of Student is = ",self.avg)
        print("-"*20)

obj=Result()
obj.get_marks()
obj.display()
```

```
Enter Name of student: Rakshit Sangal
Enter age: 19
Enter the marks in Subject 1:99
Enter the marks in Subject 2:92
Enter the marks in Subject 3:97
--------------------
Name of Student :  Rakshit Sangal
Age :  19
Marks in Subject 1: 99
Marks in Subject 2: 92
Marks in Subject 3: 97
Total Marks Obtained By Student is =  288
Average Marks Of Student is =  96.0
--------------------
```

Question 20. Write a program that create a class Distance with members km and metres. Derive classes School and office which store the distance from your house to school and office along with other details.

In [40]:

```
class Distance:
    def __init__(self,km=None,metre=None,speed=None,time=None,choice=None):
        self.speed = int(input("Enter speed in km/h :"))
        self.time = float(input("Enter time in hour :"))
        self.km = self.speed*self.time
        self.metre = ((5/18)*self.speed)*(self.time*3600)

    def display(self):
        print(round(self.km,3),"KM or",round(self.metre,3),"metre")

class School(Distance):
    def __init__(self):
        print("Calculating Distance of School from House")
        super().__init__()

    def display(self):
        print("The Distance of School from House is : ",end='')
        super().display()
        print("**"*30)

class Office(Distance):
    def __init__(self):
        print("Calculating Distance of Office: ")
        super().__init__()

    def display(self):
        print("The Distance of Office from House is: ",end='')
        super().display()
        print("**"*30)

obj=Office()
obj.display()
s=School()
s.display()
```

```
Calculating Distance of Office:
Enter speed in km/h :80
Enter time in hour :0.5
The Distance of Office from House is: 40.0 KM or 40000.0 metre
************************************************************
Calculating Distance of School from House
Enter speed in km/h :45
Enter time in hour :1.2
The Distance of School from House is : 54.0 KM or 54000.0 metre
************************************************************
```

Question 21. Write a program to create an abstract class Vehicle. Derive three classes Car, Motorcycle and Truck from it. Define appropriate methods and print the details of vehicle.

In [33]:

```
class Vehicle:
    def vehicle_number(self):
        raise NotImplementedError()

    def fuel_type(self):
        raise NotImplementedError()

    def color(self):
        raise NotImplementedError()

    def no_of_wheels(self):
        raise NotImplementedError()

    def cc(self):
        raise NotImplementedError()

class Car(Vehicle):
    def vehicle_number(self):
        return "BR06-4200"

    def fuel_type(self):
        return "Petrol"

    def color(self):
        return "White"

    def no_of_wheels(self):
        return "Four"

    def cc(self):
        return "1390cc"

class Motorcycle(Vehicle):
    def vehicle_number(self):
        return "BR06-0420"

    def fuel_type(self):
        return "Petrol"

    def color(self):
        return "white"

    def no_of_wheels(self):
        return "Two"

    def cc(self):
        return "360cc"

class Truck(Vehicle):
    def vehicle_number(self):
        return "BR06-2569"

    def fuel_type(self):
        return "Diesel"

    def color(self):
        return "Orange"

    def no_of_wheels(self):
```

```
        return "Tweleve"

    def cc(self):
        return "5005cc"

Jaguar=Car()
print("Car--------->",Jaguar.vehicle_number(),Jaguar.fuel_type(),Jaguar.color(),Jaguar.no_o
print("\n")

Bullet=Motorcycle()
print("Motorcycle-->",Bullet.vehicle_number(),Bullet.fuel_type(),Bullet.color(),Bullet.no_o
print("\n")

Turbo=Truck()
print("Truck------->",Turbo.vehicle_number(),Turbo.fuel_type(),Turbo.color(),Turbo.no_of_wh
```

```
Car---------> BR06-4200 Petrol White Four 1390cc


Motorcycle--> BR06-0420 Petrol white Two 360cc


Truck-------> BR06-2569 Diesel Orange Tweleve 5005cc
```

Question 22. Write a program that has a class Polygon. Derive two classes Rectangle and triangle from polygon and write methods to get the details of their dimensionsand hence calculate the area.

In [39]:

```
class Polygon:
    def get_data(self):
        raise NotImplementedError()

    def area(self):
        raise NotImplementedError()

class Rectangle(Polygon):
    def get_data(self):
        self.length=float(input("Length of Rectangle:"))
        self.breadth=float(input("Breadth of Rectangle:"))

    def area(self):
        return self.length*self.breadth

class Triangle(Polygon):
    def get_data(self):
        self.a=float(input("Enter side a:"))
        self.b=float(input("Enter side b:"))
        self.c=float(input("Enter side c:"))
        self.s=(self.a+self.b+self.c)/2

    def area(self):
        return (self.s*(self.s-self.a)*(self.s-self.b)*(self.s-self.c))**0.5

obj1=Rectangle()
obj1.get_data()
print("The Area of Rectangle is =",obj1.area())
print("-"*30)

obj2=Triangle()
obj2.get_data()
print("The Area of Triangle is =",obj2.area())
```

```
Length of Rectangle:20
Breadth of Rectangle:32
The Area of Rectangle is = 640.0
------------------------------
Enter side a:4
Enter side b:5
Enter side c:6
The Area of Triangle is = 9.921567416492215
```

Question 23. Write a program that extends the class Shape to calculate the area of a circle and a cone .(use super to inherit base class methods)

In [31]:

```
class Shape:
    pi=3.14
    def __init__(self,radius=None):
        self.radius=int(input("Enter Radius:"))

class Circle(Shape):
    def area(self):
        return Shape.pi*self.radius*self.radius

class Cone(Shape):
    def get_data(self):
        self.s=int(input("Enter slant height:"))

    def area(self):
        return Shape.pi*self.radius*(self.radius+self.s)

C=Circle()
print("The Area of the circle is = ",C.area())
print("--"*30)
cone=Cone()
cone.get_data()
print("The Area of the cone is = ",cone.area())
```

```
Enter Radius:28
The Area of the circle is =  2461.76
------------------------------------------------------------
Enter Radius:20
Enter slant height:16
The Area of the cone is =  2260.8
```

Question 24. Write a program to demonstrate hybrid inheritance and show MRO for each class.

In [30]:

```
class School:
    def func1(self):
        print("Inside School")

class Student1(School):
    def func2(self):
        print("Inside student 1")

class Student2(School):
    def func3(self):
        print("Inside student 2")

class Student3(Student1,Student2):
    def func4(self):
        print("Inside Student 3")

print("Showing MRO for each class")
print(School.mro())
print(Student1.mro())
print(Student2.mro())
print(Student3.mro())

#demonstration of Hybrid Inheritence
obj=Student3()
obj.func1()
obj.func2()
obj.func3()
obj.func4()
```

```
Showing MRO for each class
[<class '__main__.School'>, <class 'object'>]
[<class '__main__.Student1'>, <class '__main__.School'>, <class 'object'>]
[<class '__main__.Student2'>, <class '__main__.School'>, <class 'object'>]
[<class '__main__.Student3'>, <class '__main__.Student1'>, <class '__main__.
Student2'>, <class '__main__.School'>, <class 'object'>]
Inside School
Inside student 1
Inside student 2
Inside Student 3
```

Question 25. Write a program to overload + operator to multiply to fraction object of fraction class which contain two instance variable numerator and denominator. Also, define the instance method simplify() to simplify the fraction objects.

In [25]:

```
class Fraction:
    def __init__(self):
        self.num=0
        self.deno=1        #Since denominator cant be zero

    def get(self):
        self.num=int(input("Enter the numerator = "))
        self.deno=int(input("Enter the denomenator = "))

    def simplify(self):        #to simplyfy the fraction object
        common_divisor=Fraction.GCD(self.num,self.deno)
        self.num=self.num//common_divisor
        self.deno=self.deno//common_divisor

    @staticmethod
    def GCD(num,deno):
        if deno==0:
            return num
        else:
            return Fraction.GCD(deno,num%deno)

    def __add__(self,F):
        Temp=Fraction()
        Temp.num=(self.num*F.deno)+(F.num*self.deno)
        Temp.deno=self.deno*F.deno
        return Temp

    def display(self):
        self.simplify()
        return str(self.num)+"/"+str(self.deno)

F1=Fraction()
F1.get()
F2=Fraction()
F2.get()
F3=Fraction()
F3=F1+F2
print("Resultant Fraction is = ",F3.display())
```

```
Enter the numerator = 27
Enter the denomenator = 129
Enter the numerator = 81
Enter the denomenator = 143
Resultant Fraction is =  4770/6149
```

Question 26. Write a program to compare two-person object based on their age by overloading > operator.

In [23]:

```
print("Enter the Date of birth to compare AGE in the foramt DD/MM/YYYY")
print("*"*30)
class Person:
    def __init__(self):
        self.d=self.m=self.y=0

    def get(self):
        self.d = int(input("Enter the Day = "))
        self.m = int(input("Enter the Month = "))
        self.y = int(input("Enter the Year = "))

    def __gt__(self,P):        #gt_ > and _It_ <
        Flag = False
        if self.y>P.y:
            if self.m>P.m:
                if self.d>P.d:
                    Flag = True
                    return Flag

P1=Person()
P1.get()
print(""*20)
P2=Person()
P2.get()
print("*"*20)
print("P1 > P2",P1>P2)
```

```
Enter the Date of birth to compare AGE in the foramt DD/MM/YYYY
******************************
Enter the Day = 20
Enter the Month = 10
Enter the Year = 2020

Enter the Day = 11
Enter the Month = 06
Enter the Year = 2018
********************
P1 > P2 True
```

Question 27. Write a program to overload inoperator.

In [19]:

```
class Popular:
    def __init__(self):
        self.max_popularity = {'Python':100,"Java":90,"Ruby":70,"c++":95,"Perl":50}

    def __contains__(self,lan):
        if lan in self.max_popularity:
            return True
        else:
            return False

    def __getitem__(self,lan):
        return self.max_popularity[lan]

    def __str__(self):
        return "The Dictionary has name of Language and its popularity percentage alloted t

P=Popular()
print(str(P))
lan = input("Enter the language for which you want to know popularity : ")
if lan in P:
    print("The popularity of Language",lan,"is = ",P[lan])
```

```
The Dictionary has name of Language and its popularity percentage alloted to
them
Enter the language for which you want to know popularity : Java
The popularity of Language Java is =  90
```

Question 28. WAP to create a Complex class having real and imaginary as it attributes. Overload the +,-,/,* and += operators for objects of Complex class.

In [17]:

```
class Complex:
    def __init__(self):
        self.real = 0
        self.imag = 0

    def setValue(self,real,imag):
        self.real = real
        self.imag = imag

    #Overloading the + Operataor
    def __add__(self,C):
        Temp=Complex()
        Temp.real = self.real + C.real
        Temp.imag = self.imag + C.imag
        print("(",Temp.real,'+',Temp.imag,'i)')

    #Overloading the - Operataor
    def __sub__(self,C):
        Temp=Complex()
        Temp.real = self.real - C.real
        Temp.imag = self.imag - C.imag
        print("(",Temp.real,'-',Temp.imag,'i)')

    #Overloading the / Operataor
    def __truediv__(self,C):
        Temp=Complex()
        Temp.real = self.real / C.real
        Temp.imag = self.imag / C.imag
        print("(",round(Temp.real,2),'/',Temp.imag,'i)')

    #Overloading the * Operataor
    def __mul__(self,C):
        Temp=Complex()
        Temp.real = self.real * C.real
        Temp.imag = self.imag * C.imag
        print("(",Temp.real,'*',Temp.imag,'i*i)')

    #Overloading the += Operataor
    def __iadd__(self,C):
        Temp=Complex()
        self.real += C.real
        self.imag += self.imag + C.imag
        print("(",self.real,'+',self.imag,'i)')

    def __repr__(self):
        return self.real,self.imag

C1 = Complex()
C1.setValue(1,2)
C2 = Complex()
C2.setValue(3,4)
C3 = Complex()
C3 = C1+C2
C4 = Complex()
C4 = C1-C2
C5 = Complex()
C5 = C1/C2
C6 = Complex()
C6 = C1*C2
```

```
C1 += C2
```

```
( 4 + 6 i)
( -2 - -2 i)
( 0.33 / 0.5 i)
( 3 * 8 i*i)
( 4 + 8 i)
```

Question 29. Write a program to inspect the object using type() ,id(), isinstance(), issubclass() and callable() built-in function.

In [15]:

```
#Defining the parent class
class Vehicles:
    # Constructor
    def __init__(self):
        pass

# Defining Child class
class Car(Vehicles):
    # Constructor
    def __init__(self):
        Vehicles.__init__(self)
        print("Inspecting the object---------->")

C=Car()

#Inspecting the object using type()
print('Inspecting using type() ----> ',type(C))

#Inspecting the object using id()
print('Inspecting using id() ----> ',id(C))

#Inspecting the object using isinstance()
print('Inspecting using isinstance() ----> ',isinstance(C,Car))

#Inspecting the object using issubclass()
print('Inspecting using issubclass() ----> ',issubclass(Car,Vehicles))

#Inspecting the object using callable()
print('Inspecting using callable() ----> ',callable(C))
```

```
Inspecting the object---------->
Inspecting using type() ---->  <class '__main__.Car'>
Inspecting using id() ---->  2077635071424
Inspecting using isinstance() ---->  True
Inspecting using issubclass() ---->  True
Inspecting using callable() ---->  False
```

Question 30. WAP to inspect the program code using the functions of inspect module.

In [5]:

```
import inspect

#1. isclass()
class A(object):
    pass
print(inspect.isclass(A))

#2. ismodule()
import numpy
print(inspect.ismodule(numpy))

#3. isfunction()
def fun(a):
    return 2*a
print(inspect.isfunction(fun))

#4. ismethod()
import collections
print(inspect.ismethod(collections.Counter))

#5. getmro()
class A(object):
    pass
class B(A):
    pass
class C(B):
    pass
print(inspect.getmro(C))

#6. getmembers()
import inspect
import math
print(inspect.getmembers(math))

#7. sinnature()
import collections
print(inspect.signature(collections.Counter))

#8. stack()
def Fibonacci(n):
    if n < 0:
        return 0
    elif n == 0:
        return 0
    elif n == 1:
        return 1
    else:
        return Fibonacci(n-1)+Fibonacci(n-2)

Fibonacci(12)
print(inspect.stack())

#9. getsource()
def fun(a,b):
    return a*b
print(inspect.getsource(fun))

#10. getdoc()
import inspect
```

```
from tkinter import *
root = Tk()
root.title('Rakshit Sangal')
print(inspect.getdoc(root))
```

```
True
True
True
False
(<class '__main__.C'>, <class '__main__.B'>, <class '__main__.A'>, <class 'o
bject'>)
[('__doc__', 'This module provides access to the mathematical functions\ndef
ined by the C standard.'), ('__loader__', <class '_frozen_importlib.BuiltinI
mporter'>), ('__name__', 'math'), ('__package__', ''), ('__spec__', ModuleSp
ec(name='math', loader=<class '_frozen_importlib.BuiltinImporter'>, origin
='built-in')), ('acos', <built-in function acos>), ('acosh', <built-in funct
ion acosh>), ('asin', <built-in function asin>), ('asinh', <built-in functio
n asinh>), ('atan', <built-in function atan>), ('atan2', <built-in function
atan2>), ('atanh', <built-in function atanh>), ('ceil', <built-in function c
eil>), ('comb', <built-in function comb>), ('copysign', <built-in function c
opysign>), ('cos', <built-in function cos>), ('cosh', <built-in function cos
h>), ('degrees', <built-in function degrees>), ('dist', <built-in function d
ist>), ('e', 2.718281828459045), ('erf', <built-in function erf>), ('erfc',
<built-in function erfc>), ('exp', <built-in function exp>), ('expm1', <buil
t-in function expm1>), ('fabs', <built-in function fabs>), ('factorial', <bu
ilt-in function factorial>), ('floor', <built-in function floor>), ('fmod',
<built-in function fmod>), ('frexp', <built-in function frexp>), ('fsum', <b
uilt-in function fsum>), ('gamma', <built-in function gamma>), ('gcd', <buil
t-in function gcd>), ('hypot', <built-in function hypot>), ('inf', inf), ('i
sclose', <built-in function isclose>), ('isfinite', <built-in function isfin
ite>), ('isinf', <built-in function isinf>), ('isnan', <built-in function is
nan>), ('isqrt', <built-in function isqrt>), ('ldexp', <built-in function ld
exp>), ('lgamma', <built-in function lgamma>), ('log', <built-in function lo
g>), ('log10', <built-in function log10>), ('log1p', <built-in function log1
p>), ('log2', <built-in function log2>), ('modf', <built-in function modf>),
('nan', nan), ('perm', <built-in function perm>), ('pi', 3.141592653589793),
('pow', <built-in function pow>), ('prod', <built-in function prod>), ('radi
ans', <built-in function radians>), ('remainder', <built-in function remaind
er>), ('sin', <built-in function sin>), ('sinh', <built-in function sinh>),
('sqrt', <built-in function sqrt>), ('tan', <built-in function tan>), ('tan
h', <built-in function tanh>), ('tau', 6.283185307179586), ('trunc', <built-
in function trunc>)]
(iterable=None, /, **kwds)
[FrameInfo(frame=<frame at 0x000001E3BCAE03A0, file '<ipython-input-5-566282
92842a>', line 51, code <module>>, filename='<ipython-input-5-56628292842a
>', lineno=51, function='<module>', code_context=['print(inspect.stack())
\n'], index=0), FrameInfo(frame=<frame at 0x000001E3BAFF23B0, file 'C:\\User
s\\sanga\\anaconda3\\lib\\site-packages\\IPython\\core\\interactiveshell.p
y', line 3437, code run_code>, filename='C:\\Users\\sanga\\anaconda3\\lib\\s
ite-packages\\IPython\\core\\interactiveshell.py', lineno=3437, function='ru
n_code', code_context=['                    exec(code_obj, self.user_global_
ns, self.user_ns)\n'], index=0), FrameInfo(frame=<frame at 0x000001E3BAEE2C1
0, file 'C:\\Users\\sanga\\anaconda3\\lib\\site-packages\\IPython\\core\\int
eractiveshell.py', line 3357, code run_ast_nodes>, filename='C:\\Users\\sang
a\\anaconda3\\lib\\site-packages\\IPython\\core\\interactiveshell.py', linen
o=3357, function='run_ast_nodes', code_context=['                    if (awa
it self.run_code(code, result,  async_=asy)):\n'], index=0), FrameInfo(frame
=<frame at 0x000001E3BC1AD800, file 'C:\\Users\\sanga\\anaconda3\\lib\\site-
packages\\IPython\\core\\interactiveshell.py', line 3165, code run_cell_asyn
c>, filename='C:\\Users\\sanga\\anaconda3\\lib\\site-packages\\IPython\\core
```

\\interactiveshell.py', lineno=3165, function='run_cell_async', code_context
=['   has_raised = await self.run_ast_nodes(code_ast.body, cell
_name,\n'], index=0), FrameInfo(frame=<frame at 0x000001E3BCB65040, file
'C:\\Users\\sanga\\anaconda3\\lib\\site-packages\\IPython\\core\\async_helpe
rs.py', line 68, code _pseudo_sync_runner>, filename='C:\\Users\\sanga\\anac
onda3\\lib\\site-packages\\IPython\\core\\async_helpers.py', lineno=68, func
tion='_pseudo_sync_runner', code_context=['   coro.send(None)\n'], inde
x=0), FrameInfo(frame=<frame at 0x000001E3BAFF4330, file 'C:\\Users\\sanga
\\anaconda3\\lib\\site-packages\\IPython\\core\\interactiveshell.py', line 2
940, code _run_cell>, filename='C:\\Users\\sanga\\anaconda3\\lib\\site-packa
ges\\IPython\\core\\interactiveshell.py', lineno=2940, function='_run_cell',
code_context=['   return runner(coro)\n'], index=0), FrameInfo(fram
e=<frame at 0x000001E3BCB64040, file 'C:\\Users\\sanga\\anaconda3\\lib\\site
-packages\\IPython\\core\\interactiveshell.py', line 2894, code run_cell>, f
ilename='C:\\Users\\sanga\\anaconda3\\lib\\site-packages\\IPython\\core\\int
eractiveshell.py', lineno=2894, function='run_cell', code_context=['
result = self._run_cell(\n'], index=0), FrameInfo(frame=<frame at 0x000001E3
BCB56900, file 'C:\\Users\\sanga\\anaconda3\\lib\\site-packages\\ipykernel
\\zmqshell.py', line 536, code run_cell>, filename='C:\\Users\\sanga\\anacon
da3\\lib\\site-packages\\ipykernel\\zmqshell.py', lineno=536, function='run_
cell', code_context=['   return super(ZMQInteractiveShell, self).run_ce
ll(*args, **kwargs)\n'], index=0), FrameInfo(frame=<frame at 0x000001E3BC107
730, file 'C:\\Users\\sanga\\anaconda3\\lib\\site-packages\\ipykernel\\ipker
nel.py', line 306, code do_execute>, filename='C:\\Users\\sanga\\anaconda3
\\lib\\site-packages\\ipykernel\\ipkernel.py', lineno=306, function='do_exec
ute', code_context=['   res = shell.run_cell(code, store_histor
y=store_history, silent=silent)\n'], index=0), FrameInfo(frame=<frame at 0x0
00001E3BC2985C0, file 'C:\\Users\\sanga\\anaconda3\\lib\\site-packages\\torn
ado\\gen.py', line 234, code wrapper>, filename='C:\\Users\\sanga\\anaconda3
\\lib\\site-packages\\tornado\\gen.py', lineno=234, function='wrapper', code
_context=['   yielded = ctx_run(next, result)\n'], index=
0), FrameInfo(frame=<frame at 0x000001E3BAFF3EB0, file 'C:\\Users\\sanga\\an
aconda3\\lib\\site-packages\\ipykernel\\kernelbase.py', line 543, code execu
te_request>, filename='C:\\Users\\sanga\\anaconda3\\lib\\site-packages\\ipyk
ernel\\kernelbase.py', lineno=543, function='execute_request', code_context=
['   self.do_execute(\n'], index=0), FrameInfo(frame=<frame at 0x00
0001E3BC297D40, file 'C:\\Users\\sanga\\anaconda3\\lib\\site-packages\\torna
do\\gen.py', line 234, code wrapper>, filename='C:\\Users\\sanga\\anaconda3
\\lib\\site-packages\\tornado\\gen.py', lineno=234, function='wrapper', code
_context=['   yielded = ctx_run(next, result)\n'], index=
0), FrameInfo(frame=<frame at 0x000001E3BC176E50, file 'C:\\Users\\sanga\\an
aconda3\\lib\\site-packages\\ipykernel\\kernelbase.py', line 268, code dispa
tch_shell>, filename='C:\\Users\\sanga\\anaconda3\\lib\\site-packages\\ipyke
rnel\\kernelbase.py', lineno=268, function='dispatch_shell', code_context=['
yield gen.maybe_future(handler(stream, idents, msg))\n'], index=0), FrameInf
o(frame=<frame at 0x000001E3BC29B260, file 'C:\\Users\\sanga\\anaconda3\\lib
\\site-packages\\tornado\\gen.py', line 234, code wrapper>, filename='C:\\Us
ers\\sanga\\anaconda3\\lib\\site-packages\\tornado\\gen.py', lineno=234, fun
ction='wrapper', code_context=['   yielded = ctx_run(next,
result)\n'], index=0), FrameInfo(frame=<frame at 0x000001E3BCB4D040, file
'C:\\Users\\sanga\\anaconda3\\lib\\site-packages\\ipykernel\\kernelbase.py',
line 365, code process_one>, filename='C:\\Users\\sanga\\anaconda3\\lib\\sit
e-packages\\ipykernel\\kernelbase.py', lineno=365, function='process_one', c
ode_context=['   yield gen.maybe_future(dispatch(*args))\n'], index=0),
FrameInfo(frame=<frame at 0x000001E3BCB6CA40, file 'C:\\Users\\sanga\\anacon
da3\\lib\\site-packages\\tornado\\gen.py', line 775, code run>, filename
='C:\\Users\\sanga\\anaconda3\\lib\\site-packages\\tornado\\gen.py', lineno=
775, function='run', code_context=['   yielded = self.g
en.send(value)\n'], index=0), FrameInfo(frame=<frame at 0x000001E3BCB4CDC0,
file 'C:\\Users\\sanga\\anaconda3\\lib\\site-packages\\tornado\\gen.py', lin
e 814, code inner>, filename='C:\\Users\\sanga\\anaconda3\\lib\\site-package

```
s\\tornado\\gen.py', lineno=814, function='inner', code_context=['
self.ctx_run(self.run)\n'], index=0), FrameInfo(frame=<frame at 0x000001E3BC
B5DDD0, file 'C:\\Users\\sanga\\anaconda3\\lib\\site-packages\\tornado\\iolo
op.py', line 741, code _run_callback>, filename='C:\\Users\\sanga\\anaconda3
\\lib\\site-packages\\tornado\\ioloop.py', lineno=741, function='_run_callba
ck', code_context=['            ret = callback()\n'], index=0), FrameInfo(fr
ame=<frame at 0x000001E3BCB4E5B0, file 'C:\\Users\\sanga\\anaconda3\\lib\\si
te-packages\\tornado\\ioloop.py', line 688, code <lambda>>, filename='C:\\Us
ers\\sanga\\anaconda3\\lib\\site-packages\\tornado\\ioloop.py', lineno=688,
function='<lambda>', code_context=['                lambda f: self._run_call
back(functools.partial(callback, future))\n'], index=0), FrameInfo(frame=<fr
ame at 0x000001E3BB161880, file 'C:\\Users\\sanga\\anaconda3\\lib\\asyncio
\\events.py', line 81, code _run>, filename='C:\\Users\\sanga\\anaconda3\\li
b\\asyncio\\events.py', lineno=81, function='_run', code_context=['
self._context.run(self._callback, *self._args)\n'], index=0), FrameInfo(fram
e=<frame at 0x000001E3BC177710, file 'C:\\Users\\sanga\\anaconda3\\lib\\asyn
cio\\base_events.py', line 1859, code _run_once>, filename='C:\\Users\\sanga
\\anaconda3\\lib\\asyncio\\base_events.py', lineno=1859, function='_run_onc
e', code_context=['                handle._run()\n'], index=0), FrameInfo(fr
ame=<frame at 0x000001E3BCB49220, file 'C:\\Users\\sanga\\anaconda3\\lib\\as
yncio\\base_events.py', line 570, code run_forever>, filename='C:\\Users\\sa
nga\\anaconda3\\lib\\asyncio\\base_events.py', lineno=570, function='run_for
ever', code_context=['                self._run_once()\n'], index=0), FrameI
nfo(frame=<frame at 0x000001E3BCB49040, file 'C:\\Users\\sanga\\anaconda3\\l
ib\\site-packages\\tornado\\platform\\asyncio.py', line 199, code start>, fi
lename='C:\\Users\\sanga\\anaconda3\\lib\\site-packages\\tornado\\platform
\\asyncio.py', lineno=199, function='start', code_context=['            sel
f.asyncio_loop.run_forever()\n'], index=0), FrameInfo(frame=<frame at 0x0000
01E3BC8B2C40, file 'C:\\Users\\sanga\\anaconda3\\lib\\site-packages\\ipykern
el\\kernelapp.py', line 612, code start>, filename='C:\\Users\\sanga\\anacon
da3\\lib\\site-packages\\ipykernel\\kernelapp.py', lineno=612, function='sta
rt', code_context=['                self.io_loop.start()\n'], index=0), Fram
eInfo(frame=<frame at 0x000001E3BB058930, file 'C:\\Users\\sanga\\anaconda3
\\lib\\site-packages\\traitlets\\config\\application.py', line 845, code lau
nch_instance>, filename='C:\\Users\\sanga\\anaconda3\\lib\\site-packages\\tr
aitlets\\config\\application.py', lineno=845, function='launch_instance', co
de_context=['        app.start()\n'], index=0), FrameInfo(frame=<frame at 0x
000001E3B8B22800, file 'C:\\Users\\sanga\\anaconda3\\lib\\site-packages\\ipy
kernel_launcher.py', line 16, code <module>>, filename='C:\\Users\\sanga\\an
aconda3\\lib\\site-packages\\ipykernel_launcher.py', lineno=16, function='<m
odule>', code_context=['    app.launch_new_instance()\n'], index=0), FrameIn
fo(frame=<frame at 0x000001E3B84A5AC0, file 'C:\\Users\\sanga\\anaconda3\\li
b\\runpy.py', line 87, code _run_code>, filename='C:\\Users\\sanga\\anaconda
3\\lib\\runpy.py', lineno=87, function='_run_code', code_context=['    exec
(code, run_globals)\n'], index=0), FrameInfo(frame=<frame at 0x000001E3B8516
FD0, file 'C:\\Users\\sanga\\anaconda3\\lib\\runpy.py', line 194, code _run_
module_as_main>, filename='C:\\Users\\sanga\\anaconda3\\lib\\runpy.py', line
no=194, function='_run_module_as_main', code_context=['    return _run_code
(code, main_globals, None,\n'], index=0)]
```

```
def fun(a,b):
    return a*b
```

```
Toplevel widget of Tk which represents mostly the main window
of an application. It has an associated Tcl interpreter.
```

Question 31. Write a program to create a new list containing the first letters of every element in an

already existing list.

In [3]:

```
A=input("Enter String : ").split()
mylist=[str(i) for i in A]
def first_letter(mylist):
    for i in mylist:
        return i[0]
newlist=list(map(first_letter,mylist))

print("Old List = ",mylist)
print("New List = ",newlist)
```

```
Enter String : Python Programming is Fun!
Old List =  ['Python', 'Programming', 'is', 'Fun!']
New List =  ['P', 'P', 'i', 'F']
```

Question 32. Write a program using reduce() function to calculate the sum of first 10 natural numbers.

In [2]:

```
from functools import reduce
def add(x,y):
    return x+y
natural_num=[int(i) for i in range(1,11)]
print("First 10 Natural Numbers are:")
print(natural_num)
result=reduce(add,natural_num)
print("The Sum of first 10 Natural Number = ",result)
```

```
First 10 Natural Numbers are:
[1, 2, 3, 4, 5, 6, 7, 8, 9, 10]
The Sum of first 10 Natural Number =  55
```

Question 33. Write a program that convert a list of temperatures in Celsius into Fahrenheit using map() function.

In [105]:

```
def Fahrenheit(temperature):
    return (temperature*(9/5)+32)

temp=input("Enter the temperature in celsius seperated by spaces").split()
temp=[float(i) for i in temp]
ftemp=list(map(Fahrenheit,temp))
print("Temperature in Celsius:")
print(temp)
print("Temperature in Fahrenhiet:")
print(ftemp)
```

```
Enter the temperature in celsius seperated by spaces36.8 90 0 100 125
Temperature in Celsius:
[36.8, 90.0, 0.0, 100.0, 125.0]
Temperature in Fahrenhiet:
[98.24, 194.0, 32.0, 212.0, 257.0]
```

Question 34. Write a program that creates an iterator to print squares of numbers.

In [48]:

```
class Squares:
    def __iter__(self):
        self.num=int(input("Enter a Number : "))
        return self

    def __next__(self):
        x=self.num
        self.num=x**2
        return self.num

obj=Squares()
myiter=iter(obj)
print("The Square of entered Number Generated by Iterator: ",next(myiter))
```

```
Enter a Number : 19
The Square of entered Number Generated by Iterator:  361
```

Question 35. Write a program that create a custom iterator to create even numbers.

In [45]:

```
class Even_number:
    def __iter__(self):
        self.num=0
        return self

    def __next__(self):
        num=self.num
        self.num+=2
        return num

num= Even_number()
i= iter(num)
n=int(input("Enter a Number of terms : "))
for x in i:
    if x>n:
        break
    else:
        print(x)
print("Operate next(i) to create even number iterator")
```

```
Enter a Number of terms : 10
0
2
4
6
8
10
Operate next(i) to create even number iterator
```

Question 36. Write a program to create a generator that starts counting from 0 and raise an exception when counter is equal to 10.

In [35]:

```
import time
def count():
    num=0
    print("Counting Begins ----> ")
    while True:
        yield num
        num=num+1

for i in count():
    if i==10:
        raise StopIteration
    else:
        print(i)
        time.sleep(0.5)
```

```
Counting Begins ---->
0
1
2
3
4
5
6
7
8
9

---------------------------------------------------------------------------
StopIteration                             Traceback (most recent call last)
<ipython-input-35-d04522c82201> in <module>
      9 for i in count():
     10     if i==10:
---> 11         raise StopIteration
     12     else:
     13         print(i)

StopIteration:
```

Question 37. Write a program to create a generator to print the Fibonacci number.

In [31]:

```
def Fibonacci(nterms):
    x,y=0,1
    print(x,end=' ')
    for _ in range(nterms):
        x,y=y,x+y
        yield x
nterms=int(input("Enter the number of terms:"))
for i in Fibonacci(nterms):
    print(i,end=' ')
#This is an Example of PipeLine Generator
```

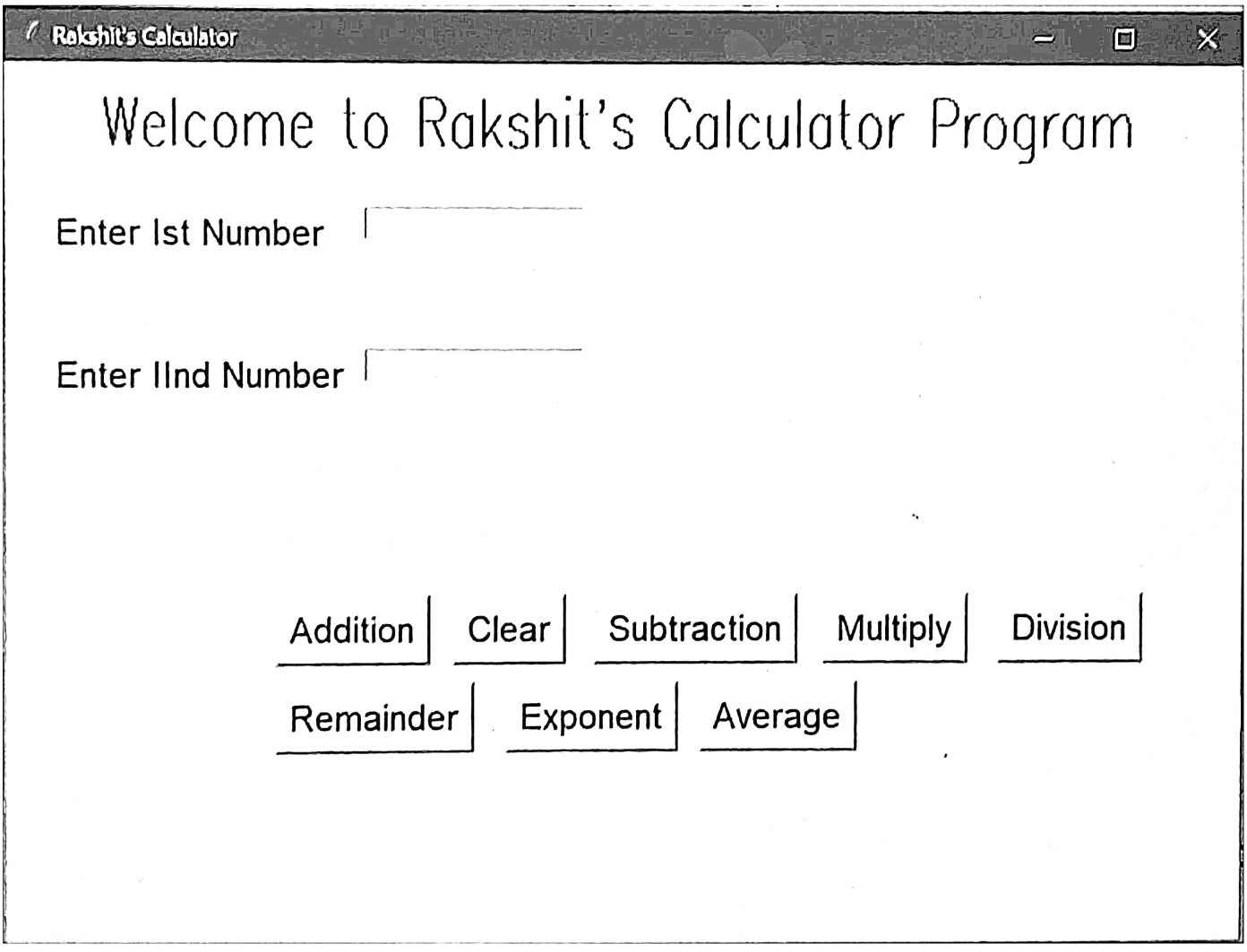
```
Enter the number of terms:10
0 1 1 2 3 5 8 13 21 34 55
```

Question 38. Write a program to create an arithmetic calculator using tkinter.

In [1]:

```
import tkinter as tk
from functools import partial
root=tk.Tk()
root.title("Rakshit's Calculator")
l=tk.Label(root,text="Welcome to Rakshit's Calculator Program",font='Modern 28 bold').place
root.geometry("700x500")

def calsum(la3,num1,num2):
    n1=int(num1.get())
    n2=int(num2.get())
    n3=n1+n2
    la3.config(fg="yellow",bg="green",height=5,width=30,text="Result=%d"%n3)

def subtract(la3,num1,num2):
    n1=int(num1.get())
    n2=int(num2.get())
    n3=n1-n2
    la3.config(fg="yellow",bg="green",height=5,width=30,text="Result=%d"%n3)

def multiply(la3,num1,num2):
    n1=int(num1.get())
    n2=int(num2.get())
    n3=n1*n2
    la3.config(fg="yellow",bg="green",height=5,width=30,text="Result=%d"%n3)

def division(la3,num1,num2):
    n1=int(num1.get())
    n2=int(num2.get())
    n3=n1/n2
    la3.config(fg="yellow",bg="green",height=5,width=30,text="Result=%d"%n3)

def remainder(la3,num1,num2):
    n1=int(num1.get())
    n2=int(num2.get())
    n3=n1%n2
    la3.config(fg="yellow",bg="green",height=5,width=30,text="Result=%d"%n3)

def exponent(la3,num1,num2):
    n1=int(num1.get())
    n2=int(num2.get())
    n3=n1**n2
    la3.config(fg="yellow",bg="green",height=5,width=30,text="Result=%d"%n3)


def average(la3,num1,num2):
    n1=int(num1.get())
    n2=int(num2.get())
    n3=(n1+n2)/2
    la3.config(fg="yellow",bg="green",height=5,width=30,text="Result=%d"%n3)



#labels
l1=tk.Label(root,text="Enter Ist Number",font='18').place(x=25,y=80)
l2= tk.Label(root,text="Enter IInd Number",font='18').place(x=25,y=160)
number1=tk.StringVar() # Hold the value of textfield in the form of String, by default it c
"200"
number2=tk.StringVar()
"300"
```

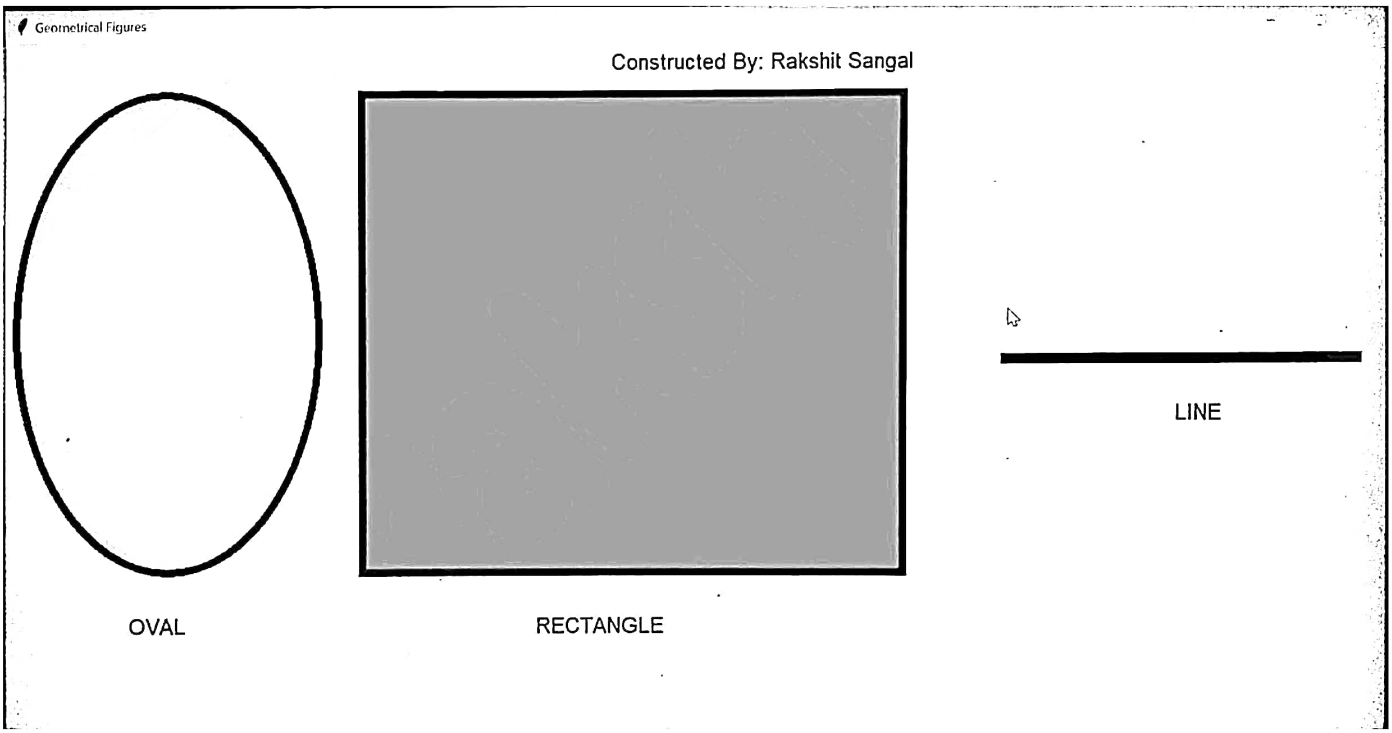
```
labelfinal=tk.Label(root)
labelfinal.place(x=250,y=200)

#textfield
t1=tk.Entry(root,textvariable=number1).place(x=200,y=80)
t2=tk.Entry(root,textvariable=number2).place(x=200,y=160)

calsum=partial(calsum,labelfinal,number1,number2)
subtract=partial(subtract,labelfinal,number1,number2)
multiply=partial(multiply,labelfinal,number1,number2)
division=partial(division,labelfinal,number1,number2)
remainder=partial(remainder,labelfinal,number1,number2)
exponent=partial(exponent,labelfinal,number1,number2)
average=partial(average,labelfinal,number1,number2)

b1=tk.Button(root,text="Addition",font='4',command=calsum).place(x=150,y=300)
b2=tk.Button(root,text="Clear",font='4',command=root.destroy).place(x=250,y=300)
b3=tk.Button(root,text="Subtraction",font='4',command=subtract).place(x=330,y=300)
b4=tk.Button(root,text="Multiply",font='4',command=multiply).place(x=460,y=300)
b5=tk.Button(root,text="Division",font='4',command=division).place(x=560,y=300)
b5=tk.Button(root,text="Remainder",font='4',command=remainder).place(x=150,y=350)
b6=tk.Button(root,text="Exponent",font='4',command=exponent).place(x=280,y=350)
b7=tk.Button(root,text="Average",font='4',command=average).place(x=390,y=350)

root.mainloop()
```



Question 39. Write a program to draw colored shapes (line, rectangle, oval) on canvas.

In [22]:

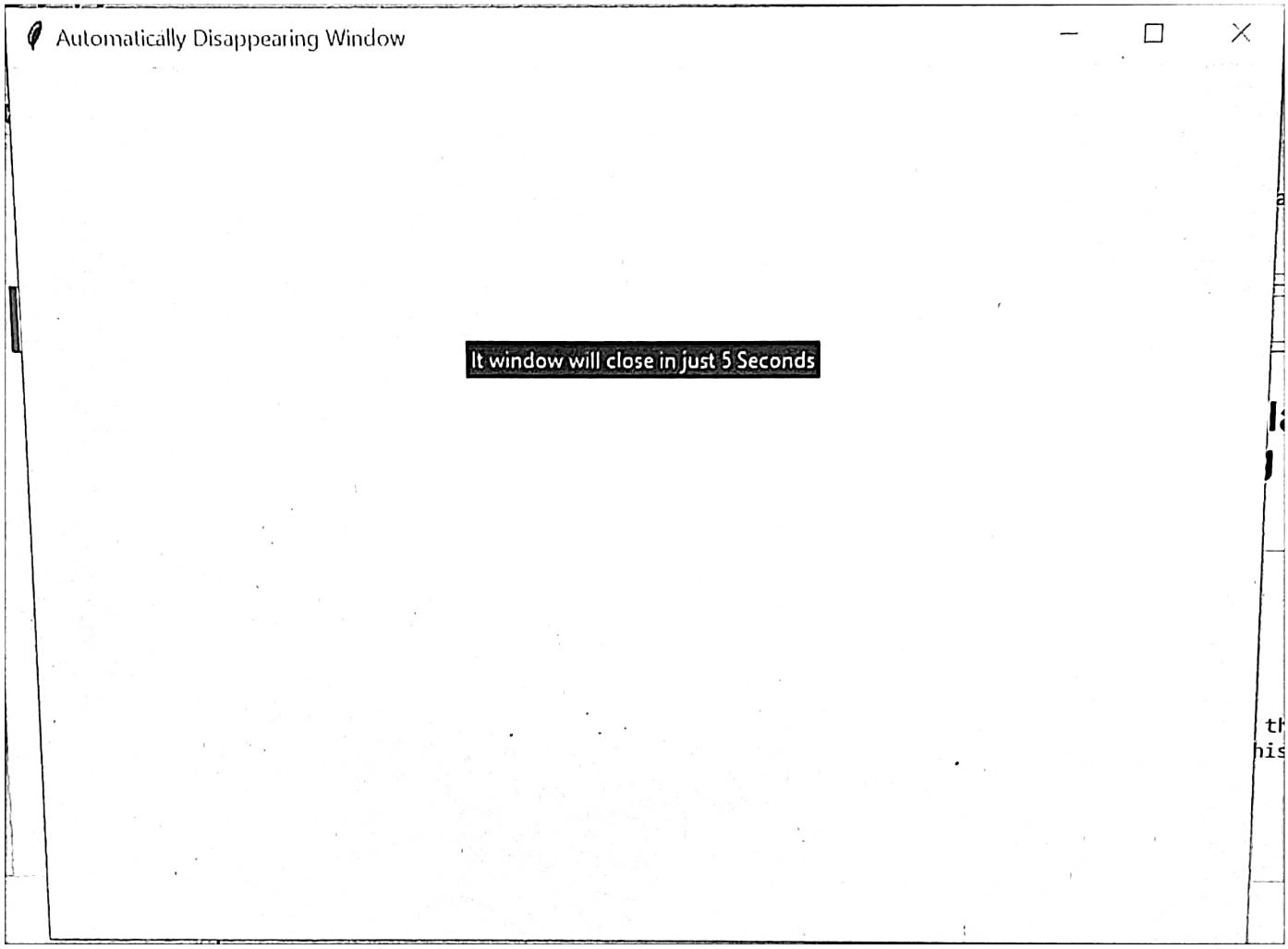
```
import tkinter as tk
top=tk.Tk()
top.title("Geometrical Figures")
canvas=tk.Canvas(top,bg='white',height=700,width=1400)

#Designing with rectangle
canvas.create_rectangle(330,500,830,50,outline='black',fill='red',width=7)

#creating Lines
canvas.create_line(920,300,1250,300,fill='black',width=10)

#creating oval
canvas.create_oval(10,50,290,500,outline='black',fill='cyan',width=7)

#Creating TextBox
canvas.create_text(700,20,text='Constructed By: Rakshit Sangal',fill='black',font='bold')
canvas.create_text(140,550,text='OVAL',fill='black',font='bold')
canvas.create_text(550,550,text='RECTANGLE',fill='black',font='bold')
canvas.create_text(1100,350,text='LINE',fill='black',font='bold')
canvas.pack()
top.mainloop()
```



Question 40. Write a program to create a window that disappears automatically after 5 seconds.

In [26]:

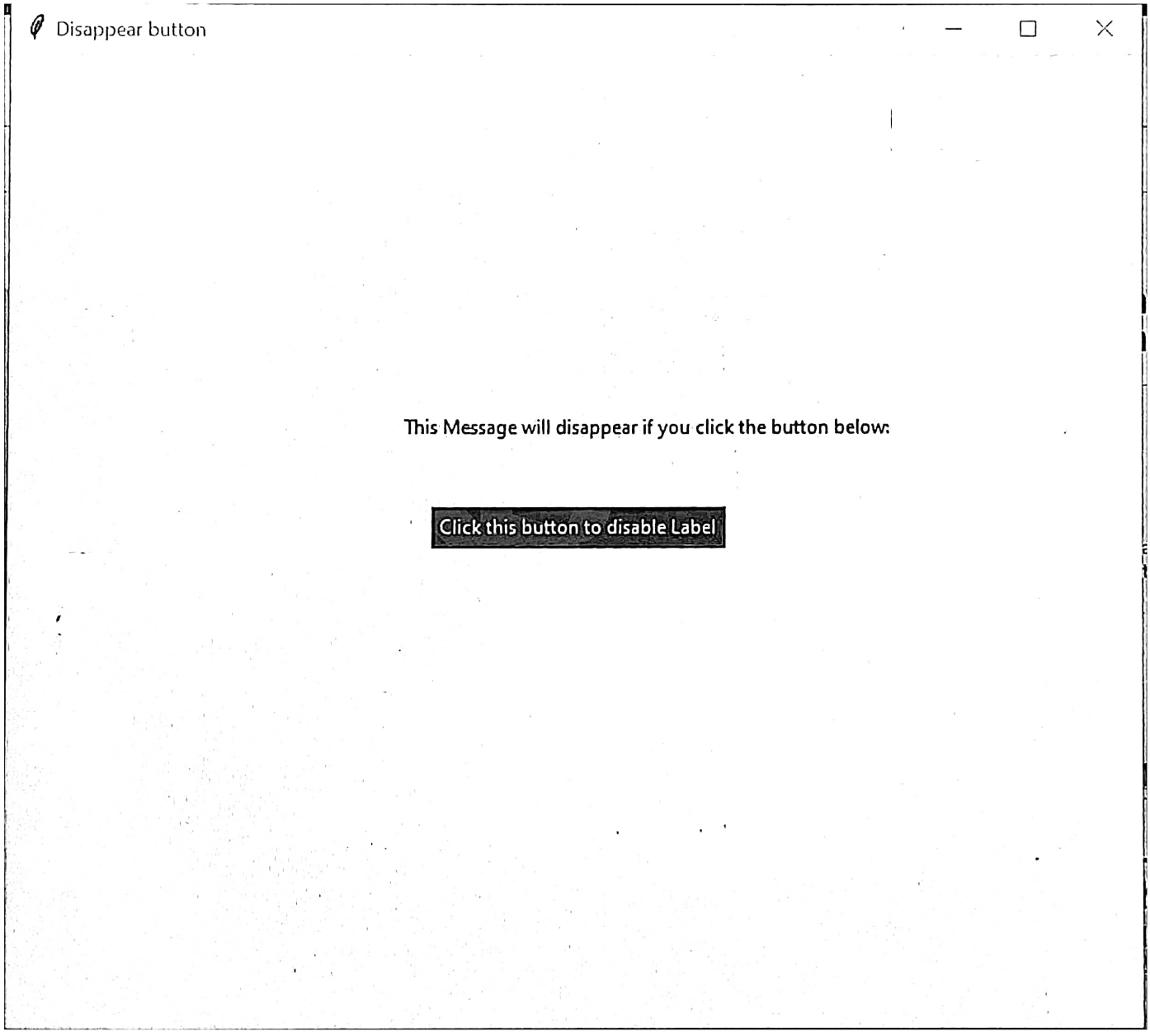
```
import tkinter as tk
root = tk.Tk()
root.title("Automatically Disappearing Window")
root.geometry("700x500")
tk.Label(root, text="It window will close in just 5 Seconds",bg='black',fg='orange').place(
root.after(5000, lambda: root.destroy())    # time in ms 5 sec=5000
root.mainloop()
```



Question 41. Write a program to create a button and a label inside the frame widget. Button should change the color upon hovering over the button and label should disappear on clicking the button.

In [28]:

```
import tkinter as tk
class Test():
    def __init__(self):
        self.root = tk.Tk()
        self.root.title("Disappear button")
        self.root.geometry("700x600")
        self.label=tk.Label(self.root, text = "This Message will disappear if you click the
        self.buttonForget = tk.Button(self.root,fg='orange',bg='black', text = 'Click this

        self.buttonForget.pack()
        self.buttonForget.place(x=260,y=280)
        self.label.pack()
        self.label.place(x=240,y=220)
        self.root.mainloop()
app = Test()
```



Question 42. Write a program to create radio-buttons

(Male, Female, and Transgender) and a label. Default selection should be on Female and the label must display the current selection made by user.

In [31]:

```
from IPython.display import display
import ipywidgets as widgets
y=widgets.Label()
x=widgets.RadioButtons( options=['Male','Female','Transgender'], value='Female', descriptio
display(x,y)
pythonlink=widgets.link((x,'value'),(y,'value'))
```

```
RadioButtons(description='Gender', index=1, options=('Male', 'Female', 'Tran
sgender'), value='Female')
```

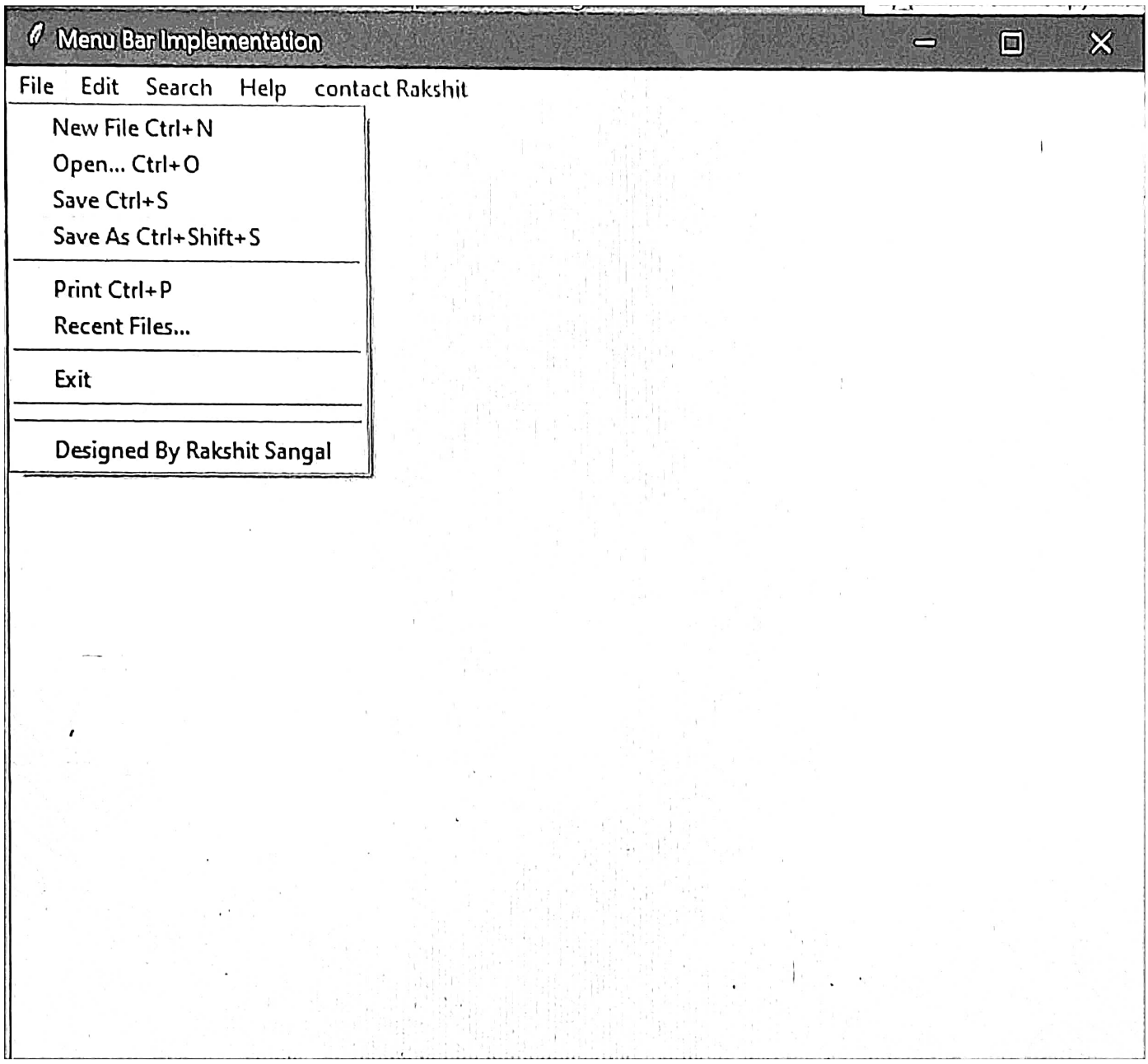
```
Label(value='')
```

Question 43. Write a program to display a menu on the menu bar.

In [29]:

```
from tkinter import *
from tkinter.ttk import *
from time import strftime

# creating tkinter window
root = Tk()
root.title('Menu Bar Implementation')
root.geometry("600x500")

# Creating Menubar
menubar = Menu(root)

# Adding File Menu and commands
file = Menu(menubar, tearoff = 0)
menubar.add_cascade(label ='File', menu = file)
file.add_command(label ='New File Ctrl+N', command = None)
file.add_command(label ='Open... Ctrl+O', command = None)
file.add_command(label ='Save Ctrl+S', command = None)
file.add_command(label ='Save As Ctrl+Shift+S', command = None)
file.add_separator()
file.add_command(label ='Print Ctrl+P', command = None)
file.add_command(label ='Recent Files...', command = None)
file.add_separator()
file.add_command(label ='Exit', command = root.destroy)
file.add_separator()
file.add_separator()
file.add_command(label ='Designed By Rakshit Sangal', command = None)

#Adding Edit Menu and commands
edit = Menu(menubar, tearoff = 0)
menubar.add_cascade(label ='Edit', menu = edit)
edit.add_command(label ='Cut', command = None)
edit.add_command(label ='Copy', command = None)
edit.add_command(label ='Paste', command = None)
edit.add_command(label ='Select All', command = None)
edit.add_separator()
edit.add_command(label ='Find...', command = None)
edit.add_command(label ='Find again', command = None)

# Adding Search Menu and commands
search = Menu(menubar, tearoff = 0)
menubar.add_cascade(label ='Search', menu = search)

# Adding Help Menu
help_ = Menu(menubar, tearoff = 0)
menubar.add_cascade(label ='Help', menu = help_)
help_.add_command(label ='Tk Help', command = None)
help_.add_command(label ='Demo', command = None)
help_.add_separator()
help_.add_command(label ='About Tk', command = None)

# Adding name Menu and commands
user = Menu(menubar, tearoff = 0)
menubar.add_cascade(label ='contact Rakshit', menu = user)
user.add_command(label = "contact",command=None)
user.add_command(label = "mail",command=None)
user.add_command(label = "terminate",command=root.destroy)

# display Menu
```

```
root.config(menu = menubar)
mainloop()
```



Question 44. Write a NumPy program to create an array of (3, 4) shape, multiply every element value by 3 and display the new array.

In [25]:

```
import numpy as np
arr1=np.arange(1,13)
arr1=arr1.reshape(3,4)
print("Array of dimension(3,4)---->>>\n",arr1)
print("\n")
newarr=arr1*3
print("Array after multiplying every element by 3 -->>>>\n",newarr)
```

```
Array of dimension(3,4)---->>>
 [[ 1  2  3  4]
 [ 5  6  7  8]
 [ 9 10 11 12]]


Array after multiplying every element by 3 -->>>>
 [[ 3  6  9 12]
 [15 18 21 24]
 [27 30 33 36]]
```

Question 45. Write a NumPy program to compute the multiplication of two given matrixes.

In [22]:

```
import numpy as np
m1 = [[1, 0, 1], [0, 1, 1],[1,1,1]]
m2 = [[1, 2], [3, 4],[5,6]]

#Showing original Matrix
print("Original Matrix:")
print(m1)
print(m2)

result = np.dot(m1,m2)
print("Result of the Matrix Multiplication:")
print(result)
```

```
Original Matrix:
[[1, 0, 1], [0, 1, 1], [1, 1, 1]]
[[1, 2], [3, 4], [5, 6]]
Result of the Matrix Multiplication:
[[ 6  8]
 [ 8 10]
 [ 9 12]]
```

Question 46. Write a Program to create a series from a list, numpy array and dict.

In [20]:

```
import pandas as pd
import numpy as np

#sample List
mylist = ['Celestial','interface','Rakshit','Aggarwal']
list2Series = pd.Series(mylist,index=['a','b','c','d'])
print("Creating Series from list : ")
print("Original List : ",mylist)
print(list2Series)
print('---'*10+">>>>>>>>>>"+'---'*10)

#Sample Numpy Array
myarr = np.array(['M','O','R','P','H','O'])
arr2Series = pd.Series(myarr)
print("Creating Series from Numpy Array : ")
print("Original Array : ",myarr)
print(arr2Series)
print('---'*10+">>>>>>>>>>"+'---'*10)

#Sample Dictionary
mydict= {'Russia' : 1, 'Canada' : 2, 'US' : 3, 'China' : 4, 'Brazil' : 5, 'India' :7}
dict2series = pd.Series(mydict,index=['a','b','c','d','e','f'])
print("Creating Series from Dictionary : ")
print("Original Dictionary : ",mydict)
print(dict2series)
print('---'*10+"**********"+'---'*10)
print('---'*10+"**********"+'---'*10)
```

```
Creating Series from list :
Original List :  ['Celestial', 'interface', 'Rakshit', 'Aggarwal']
a    Celestial
b    interface
c      Rakshit
d     Aggarwal
dtype: object
------------------------------>>>>>>>>>>------------------------------
Creating Series from Numpy Array :
Original Array :  ['M' 'O' 'R' 'P' 'H' 'O']
0    M
1    O
2    R
3    P
4    H
5    O
dtype: object
------------------------------>>>>>>>>>>------------------------------
Creating Series from Dictionary :
Original Dictionary :  {'Russia': 1, 'Canada': 2, 'US': 3, 'China': 4, 'Braz
il': 5, 'India': 7}
a   NaN
b   NaN
c   NaN
d   NaN
e   NaN
f   NaN
dtype: float64
------------------------------**********------------------------------
------------------------------**********------------------------------
```

Question 47. Write a Program to convert a numpy array to a dataframe of given shape.

In [12]:

```
import pandas as pd
import numpy as np
myarr = np.array([[10,20,30,20,10], [40,50,60,50,40], [70,80,90,80,70]])
#Converting into Dataframe
df = pd.DataFrame(data = myarr)
print(df)
```

```
    0   1   2   3   4
0  10  20  30  20  10
1  40  50  60  50  40
2  70  80  90  80  70
```

Question 48. Write a program to count number of missing values in each column.

In [13]:

```
import pandas as pd
import numpy as np
cars_data1 = pd.read_csv('C:\\Users\\sanga\\Data\\Toyota1.csv')
df = pd.DataFrame(cars_data1)
print(df)

#Showing the NaN value or boolean DataFrame
print("Show the boolean Dataframe: \n \n",df.isnull())
#Count total NaN at each column :
print("NaN value in each column: ",df.isnull().sum())
```

```
      Unnamed: 0  Price   Age     KM FuelType   HP  MetColor  Automatic    C
C  \
0              0  13500  23.0  46986   Diesel   90       1.0          0  200
0
1              1  13750  23.0  72937   Diesel   90       1.0          0  200
0
2              2  13950  24.0  41711   Diesel   90       NaN          0  200
0
3              3  14950  26.0  48000   Diesel   90       0.0          0  200
0
4              4  13750  30.0  38500   Diesel   90       0.0          0  200
0
...          ...    ...   ...    ...      ...  ...       ...        ...
...
1431        1431   7500   NaN  20544   Petrol   86       1.0          0  130
0
1432        1432  10845  72.0     ??   Petrol   86       0.0          0  130
0
1433        1433   8500   NaN  17016   Petrol   86       0.0          0  130
0
1434        1434   7250  70.0     ??      NaN   86       1.0          0  130
0
1435        1435   6950  76.0      1   Petrol  110       0.0          0  160
0

      Doors  Weight
0     three    1165
1         3    1165
2         3    1165
3         3    1165
4         3    1170
...     ...     ...
1431      3    1025
1432      3    1015
1433      3    1015
1434      3    1015
1435      5    1114

[1436 rows x 11 columns]
Show the boolean Dataframe:

       Unnamed: 0  Price    Age     KM  FuelType     HP  MetColor  Automatic
\
0           False  False  False  False     False  False     False      False
1           False  False  False  False     False  False     False      False
2           False  False  False  False     False  False      True      False
3           False  False  False  False     False  False     False      False
4           False  False  False  False     False  False     False      False
```

```
...          ...    ...    ...    ...       ...    ...       ...        ...
1431       False  False   True  False     False  False     False      False
1432       False  False  False  False     False  False     False      False
1433       False  False   True  False     False  False     False      False
1434       False  False  False  False      True  False     False      False
1435       False  False  False  False     False  False     False      False

         CC  Doors  Weight
0     False  False   False
1     False  False   False
2     False  False   False
3     False  False   False
4     False  False   False
...     ...    ...     ...
1431  False  False   False
1432  False  False   False
1433  False  False   False
1434  False  False   False
1435  False  False   False

[1436 rows x 11 columns]
NaN value in each column:  Unnamed: 0      0
Price           0
Age           100
KM              0
FuelType      100
HP              0
MetColor      150
Automatic       0
CC              0
Doors           0
Weight          0
dtype: int64
```

Question 49. Write a program to replace missing values in a column of a dataframe by the mean value of that column.

In [10]:

```
import pandas as pd
import numpy as np
#Loading the csv file
car_data=pd.read_csv('C:\\Users\\sanga\\Data\\Toyota1.csv')
#Creating a dataframe
df = pd.DataFrame(car_data)
print(df)
#Findig Means Value of column having NaN
mean_value=df['Age'].mean()
print("The Means Value of Age Column is : ",round(mean_value,3))
#Replacing NaNs in column Age with the means values in the same column
df['Age'].fillna(value=mean_value,inplace=True)
print('Updated DataFram : ')
print(df)
```

```
      Unnamed: 0  Price   Age     KM FuelType   HP  MetColor  Automatic    C
C  \
0              0  13500  23.0  46986   Diesel   90       1.0          0  200
0
1              1  13750  23.0  72937   Diesel   90       1.0          0  200
0
2              2  13950  24.0  41711   Diesel   90       NaN          0  200
0
3              3  14950  26.0  48000   Diesel   90       0.0          0  200
0
4              4  13750  30.0  38500   Diesel   90       0.0          0  200
0
...          ...    ...   ...    ...      ...  ...       ...        ...
...
1431        1431   7500   NaN  20544   Petrol   86       1.0          0  130
0
1432        1432  10845  72.0     ??   Petrol   86       0.0          0  130
0
1433        1433   8500   NaN  17016   Petrol   86       0.0          0  130
0
1434        1434   7250  70.0     ??      NaN   86       1.0          0  130
0
1435        1435   6950  76.0      1   Petrol  110       0.0          0  160
0

      Doors  Weight
0     three    1165
1         3    1165
2         3    1165
3         3    1165
4         3    1170
...     ...     ...
1431      3    1025
1432      3    1015
1433      3    1015
1434      3    1015
1435      5    1114

[1436 rows x 11 columns]
The Means Value of Age Column is :  55.672
Updated DataFram :
      Unnamed: 0  Price        Age     KM FuelType   HP  MetColor  Automatic
\
0              0  13500  23.000000  46986   Diesel   90       1.0          0
```

```
1              1  13750  23.000000  72937   Diesel   90       1.0          0
2              2  13950  24.000000  41711   Diesel   90       NaN          0
3              3  14950  26.000000  48000   Diesel   90       0.0          0
4              4  13750  30.000000  38500   Diesel   90       0.0          0
...          ...    ...        ...    ...      ...  ...       ...        ...
1431        1431   7500  55.672156  20544   Petrol   86       1.0          0
1432        1432  10845  72.000000     ??   Petrol   86       0.0          0
1433        1433   8500  55.672156  17016   Petrol   86       0.0          0
1434        1434   7250  70.000000     ??      NaN   86       1.0          0
1435        1435   6950  76.000000      1   Petrol  110       0.0          0

        CC  Doors  Weight
0     2000  three    1165
1     2000      3    1165
2     2000      3    1165
3     2000      3    1165
4     2000      3    1170
...    ...    ...     ...
1431  1300      3    1025
1432  1300      3    1015
1433  1300      3    1015
1434  1300      3    1015
1435  1600      5    1114

[1436 rows x 11 columns]
```

Question 50. Write a Pandas program to create a line plot of the opening, closing stock prices of Alphabet Inc. between two specific dates. Use the alphabet_stock_data.csv file to extract data.

In [ ]:

```
import pandas as pd
import matplotlib.pyplot as plt
df = pd.read_csv("/home/rakshit/Downloads/alphabet_stock_data.csv")
start_date = pd.to_datetime('2020-4-1')
end_date = pd.to_datetime('2020-09-30')
df['Date'] = pd.to_datetime(df['Date'])
new_df = (df['Date']>= start_date) & (df['Date']<= end_date)
df2 = df.loc[new_df]
plt.figure(figsize=(10,10))
df2.plot(x='Date', y=['Open', 'Close']);
plt.suptitle('Opening/Closing stock prices of Alphabet Inc.,\n 01-04-2020 to 30-09-2020', f
plt.xlabel("Date",fontsize=12, color='black')
plt.ylabel("$ price", fontsize=12, color='black')
plt.show()
```