class student:

    course\_name = "DBDA"            #class variable

    def \_\_init\_\_(self,name,rollno,marks1,marks2):       #method

        self.name=name              #instance variable

        self.rollno=rollno          #instance variable

        self.marks1=marks1          #instance variable

        self.marks2=marks2          #instance variable

        student.institue="IACSD"    #class variable

        print("inside \_\_init\_\_")

    def display\_info(self):                             #instance method

        print(self.name, self.marks1)

    def cal\_total\_marks(self):                          #instance method

        total = self.marks1 + self.marks2

        print("Total marks: ", total)

    def display\_institue(cls):                          #class method

        print(cls.institue)

student\_object = student("ram",123,10,20)

student\_object.display\_info()

student\_object.cal\_total\_marks()

student\_object.display\_institue()

print(student\_object.course\_name)

print(student\_object.name)

print(student\_object.rollno)

print(student\_object.marks1)

print(student\_object.marks2)

print(student\_object.institue)

student\_object.name="Shivani"

print(student\_object.name)              #update instance variable

student\_object.institue="CADC"

print(student\_object.institue)          #update class variable

class student:

    course\_name="DBDA"

    def \_\_init\_\_(self):

        print("inside init")

        self.course\_name="PG-DBDA"

obj=student()

print(obj.course\_name)

# o/p:

# inside init

# PG-DBDA

# if class variable and instance variable has same name and if we call a method and that variable is present inside

# that method as instance variable, in this case instance variable will get preferrence over class variable.

#Add attributes and delete attributes from an instance

class student:

    def \_\_init\_\_(self,marks1,marks2):         #instance method

        self.marks1=marks1

        self.marks2=marks2

obj1=student(10,20)

obj2=student(40,50)

obj2.marks3=60

print(obj1.marks1, obj1.marks2)                 #10 20

print(obj2.marks1, obj2.marks2, obj2.marks3)    #40 50 60

del obj2.marks2

print(obj1.marks1, obj1.marks2)                 #10 20

print(obj1.mark1, obj2.mark2)                   #AttributeError:

class course:

    agency\_creating\_course="CDAC"       #class variable

    def \_\_init\_\_(self, course\_id, course\_name, duration, total\_hours\_of\_syllabus, contents, staff\_count):

        self.course\_id=course\_id                                #instance variable

        self.course\_name=course\_name                            #instance variable

        self.duration=duration                                  #instance variable

        self.total\_hours\_of\_syllabus=total\_hours\_of\_syllabus    #instance variable

        self.contents=contents                                  #instance variable

        self.staff\_count=staff\_count                            #instance variable

    def display\_info(self):                                     #instance method

        print(self.course\_id,self.course\_name, self.duration, self.total\_hours\_of\_syllabus, self.contents, self.staff\_count)

    @classmethod

    def display\_agency(cls):                                    #class method

        print(cls.agency\_creating\_course)

obj1 = course('1002', 'DBDA', '6','600 hours',"pyhton, java, os, etc",'23')

obj1.display\_info()                 #1002 DBDA 6 600 hours pyhton, java, os, etc 23

obj1.display\_agency()               #CDAC

#Data Hiding

class tcs:

    def \_\_init\_\_(self, emp\_name, emp\_id):

        self.emp\_name=emp\_name

        self.emp\_id=emp\_id

        self.\_\_emp\_moble\_no = '123456789'        #hidden

t=tcs('Rohit', '1001')

print(t.emp\_name)

print(t.emp\_id)

# print(t.\_\_emp\_moble\_no)   #AttributeError: 'tcs' object has no attribute '\_\_emp\_moble\_no'.?

print(t.\_tcs\_\_emp\_moble\_no)     #to see hidden data obj.\_ClassName\_\_VarName

Single class inheritance

#Single Class Inheritance

class Planet:

    def \_\_init\_\_(self):

        print("inside Planet \_\_init\_\_(self)")

        self.shape="ellipsoid"

        self.size=10

    def display(self):

        print(f"The shape of planet is {self.shape}.")

    def increment\_size(self):

        self.size+=1

        print("inside Planet: ", self.size)

class Earth(Planet):

    def \_\_init\_\_(self):

        print("Inside earth \_\_init\_\_")

        Planet.\_\_init\_\_(self)   ##call the \_\_init\_\_ of the base class in order to access base class method, attributes

        self.c=[9,8,8]

    def display(self):

        print("inside display of Earth: ", self.size, self.c)

    def increment\_size(self):

        self.size+=5

        print("inside earth: ", self.size)

obj=Earth()

obj.display()

                    #Inside earth \_\_init\_\_

                    #Inside Planet \_\_init\_\_(self)

                    #Inside display of Earth:  10 [9, 8, 8]

obj.increment\_size()        #inside earth:  15