**EXPERIMENT NO – 06**

***1-Class and Object***

1-#CLASS AND OBJECT IN PY

class student:

    rno=123

    name = "abc"

    branch= "cse"

    def read(self):

        print("is reading")

    def write(self):

        print("is writing")

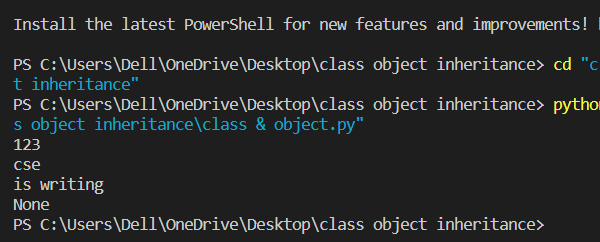
s1= student()

print(s1.rno)

print(s1.branch)

print(s1.write())

OUTPUT:



2-#CREATING OBJECT IN PYTHON

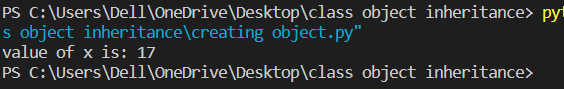
class MyClass:

    x=17;

p1 = MyClass() # p1 is a object of class MyClass

print("value of x is:",p1.x) #calling object

OUTPUT:



3-# \_\_INIT\_\_() FUNCTION IN PYTHON

class Student:

  def \_\_init\_\_(self, name, age): #self is refarnce parameter . we can give any name instead of "self"

    self.name = name

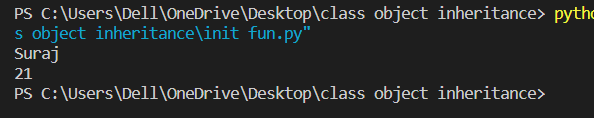
    self.age = age

p1 = Student("Suraj", 21)   #pasing argument

print(p1.name)

print(p1.age)

OUTPUT:



4-#MODIFY OBJECT IN PYHTON

class Person:

  def \_\_init\_\_(self, name, age):

    self.name = name

    self.age = age

  def myfunc(self):

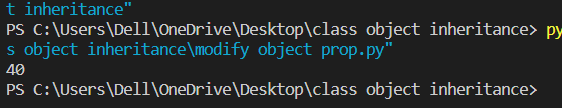
    print("Hello my name is " + self.name)

p1 = Person("John", 36)

p1.age = 40 #

print(p1.age)

OUTPUT:



***2- Inheritance:***

1-Single Inheritance

class employee:

    id1 = 123

    name = "ajay"

    department = "CSE"

    def display(self):

        print("parent class")

class xyz(employee):

    mob\_no =8468992003

    def show(self):

        print("child class")

c1=xyz()

print(c1.id1)

print(c1.name)

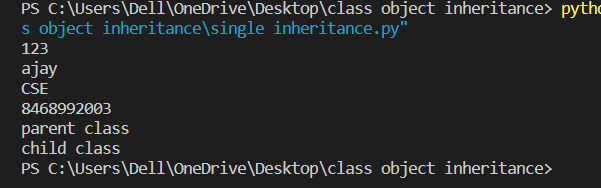
print(c1.department)

print(c1.mob\_no)

c1.display()

c1.show()

output:



2- Multi-Level inheritance

class Animal:

    def speak(self):

        print("Animal Speaking")

#The child class Dog inherits the base class Animal

class Dog(Animal):

    def bark(self):

        print("dog barking")

#The child class Dogchild inherits another child class Dog

class DogChild(Dog):

    def eat(self):

        print("Eating bread...")

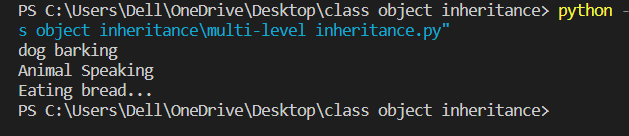
d = DogChild()

d.bark()

d.speak()

d.eat()

output:



3-Multiple inheritance

class Father():

    def Driving(self):

        print("Father Enjoys Driving")

class Mother():

    def Cooking(self):

        print("Mother Enjoys Cooking")

class Child(Father, Mother):

    def Playing(self):

        print("Child Loves Playing")

c = Child()

c.Driving()

c.Cooking()

c.Playing()

output:

