NAME – Soham Desai ROLL NO – 11

BATCH – B XIE ID - 202003021

Python – Experiment 3

AIM : Object Oriented Programming in Python

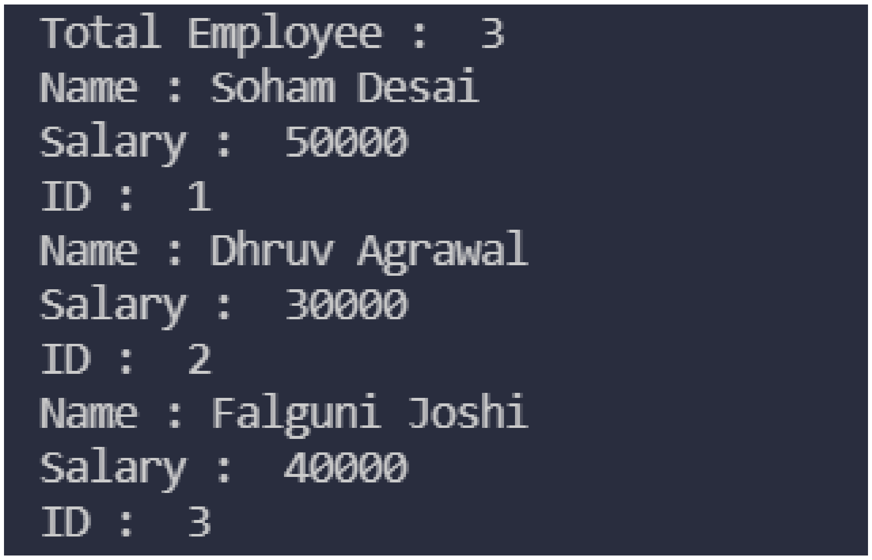
LO - 2 : Illustrate the concepts of object-oriented programming as used in Python

1. Design an person/employee / account class using python for reading & displaying the employee information.

Code:

```
class Employee:
   co = 0
   def __init__(self, name, salary,id):
      self.name = name
      self.salary = salary
      self.id = id
      Employee.co += 1

   def Display(self):
      print("Name :", self.name)
      print("Salary : ",self.salary)
      print("ID : ",self.id)
   def displayCount(self):
      print("Total Employee : ",Employee.co)

emp1 = Employee("Soham Desai",50000,1)
emp2 = Employee("Dhruv Agrawal",30000,2)
emp3 = Employee("Falguni Joshi",40000,3)

emp3.displayCount()
emp1.Display()
emp2.Display()
emp3.Display()
```

Output:



```
Total Employee :  3
Name : Soham Desai
Salary :  50000
ID :  1
Name : Dhruv Agrawal
Salary :  30000
ID :  2
Name : Falguni Joshi
Salary :  40000
ID :  3
```

2.Write python programs to understand
a) Classes, Objects, Constructors, Inner class and Static method

Code:

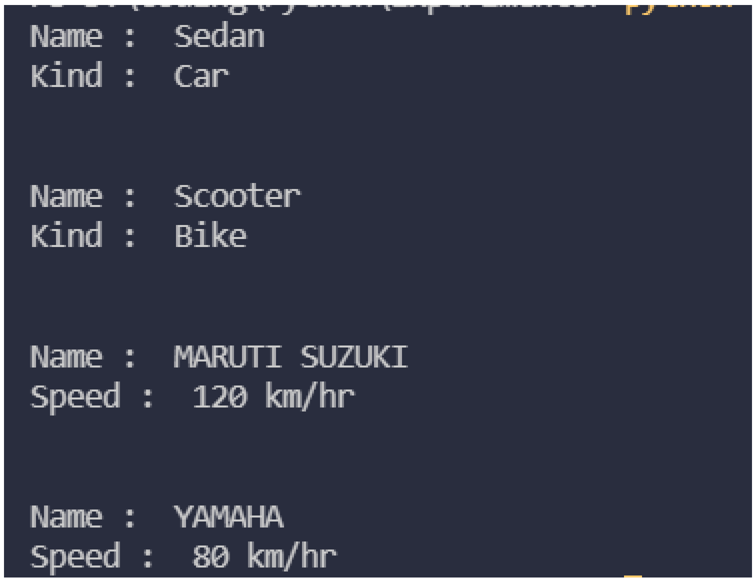
```
class Vehicle:
   def __init__(self, name, type):
      self.name = name
      self.kind = type
      self.car = self.Car()
      self.bike = self.Bike()

   def show(self):
      print("Name : ",self.name)
      print("Kind : ",self.kind)

   class Car:
      def __init__(self):
         self.name = "MARUTI SUZUKI"
         self.speed = "120 km/hr"
      def show(self):
         print("Name : ",self.name)
         print("Speed : ",self.speed)
```

NAME – Soham Desai ROLL NO – 11

BATCH – B XIE ID - 202003021

```
    class Bike:
        def __init__(self):
            self.name = "YAMAHA"
            self.speed = "80 km/hr"
        def show(self):
            print("Name : ",self.name)
            print("Speed : ",self.speed)

a = Vehicle("Sedan","Car")
b = Vehicle("Scooter","Bike")
a.show()
print("\n")
b.show()
print("\n")
c = b.Car()
c1 = b.Bike()
c.show()
print("\n")
c1.show()
```

Output:



```
Name :  Sedan
Kind :  Car


Name :  Scooter
Kind :  Bike


Name :  MARUTI SUZUKI
Speed :  120 km/hr


Name :  YAMAHA
Speed :  80 km/hr
```

NAME – Soham Desai   ROLL NO – 11

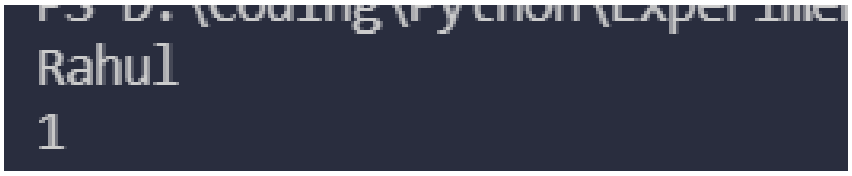
BATCH – B   XIE ID - 202003021

Write a python program to understand
b) Different types of Inheritance

Code:

```
class Person():
    def __init__(self, name, idnumber):
        self.name = name
        self.idnumber = idnumber
    def display(self):
        print(self.name)
        print(self.idnumber)

class Employee(Person):
    def __init__(self, name, idnumber, salary, post):
        self.salary = salary
        self.post = post
        Person.__init__(self, name, idnumber)

a = Employee('Rahul', 1, 20000, "Intern")
a.display()
```

Output:



```
Rahul
1
```

NAME – Soham Desai ROLL NO – 11

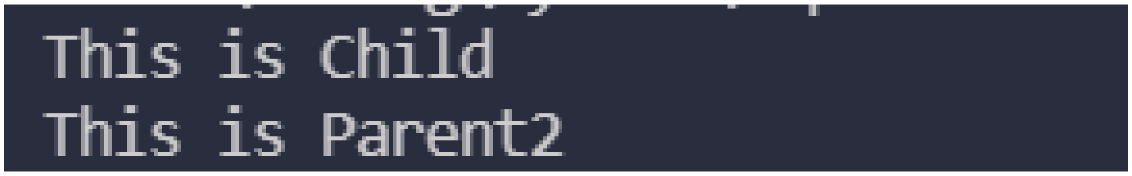
BATCH – B XIE ID - 202003021

c) Polymorphism using Operator overloading, Method overloading, Method overriding, Abstract class, Abstract method and Interfaces in Python.

Code:

```
class Parent1():
    def show(self):
        print("This is Parent1")

class Parent2():
    def display(self):
        print("This is Parent2")

class Child(Parent1, Parent2):
    def show(self):
        print("This is Child")

obj = Child()
obj.show()
obj.display()
```

Output:



```
This is Child
This is Parent2
```

Code:

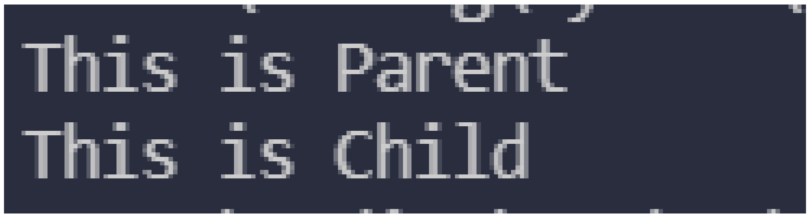
```
class Parent():
    def __init__(self):
        self.value = "This is Parent"

    def show(self):
        print(self.value)

class Child(Parent):
    def __init__(self):
        self.value = "This is Child"
```

NAME – Soham Desai ROLL NO – 11

BATCH – B XIE ID - 202003021

```
    def show(self):
        print(self.value)
obj1 = Parent()
obj2 = Child()
obj1.show()
obj2.show()
```

Output:



```
This is Parent
This is Child
```

Code:

```
from abc import ABC,abstractmethod
class Animal(ABC):
    @abstractmethod
    def move(self):
        pass
class Human(Animal):
    def move(self):
        print("I can walk and run")
class Snake(Animal):
    def move(self):
        print("I can crawl")
class Dog(Animal):
    def move(self):
        print("I can bark")
 class Lion(Animal):
    def move(self):
        print("I can roar")
 c=Animal()
```

Conclusion: From this Experiment we have learned about classes and the implementation of the classes, method overloading and method overriding and also about abstraction.