# Practical 04

Exercise 01:

Create a class called “Employee” which has 3 private variables (empID, empName, empDesignation) and create getters and setters for each field. Please note that this has no main method since this is just a blueprint not a application. Now crate a test class to invoke the Employee class. Create two objects for Mr.Bogdan and Ms.Bird and set required values using setters and print them back on the console using getters.

*

```
public class Employee
 {
   private int empID;
   private String empName;
   private String empDesignation;
   public int getEmpID()
  {
      return empID;
   }
   public void setEmpID(int empID)
   {
      this.empID = empID;
   }
   public String getEmpName()
   {
      return empName;
```

```
    }
public void setEmpName(String empName)
   {
       this.empName = empName;
    }
    public String getEmpDesignation()
   {
       return empDesignation;
    }

    public void setEmpDesignation(String empDesignation)
   {
       this.empDesignation = empDesignation;
    }
}



public class TestEmployee
{
    public static void main(String[] args)
    {
       // Create an Employee object for Mr. Bogdan
       Employee bogdan = new Employee();
       bogdan.setEmpID(101);
```

```
        bogdan.setEmpName("Bogdan");
        bogdan.setEmpDesignation("Software Engineer");

        // Create another Employee object for Ms. Bird
        Employee bird = new Employee();
        bird.setEmpID(102);
        bird.setEmpName("Bird");
        bird.setEmpDesignation("Data Scientist");

         // Print the details of Mr. Bogdan and Ms. Bird
        System.out.println("Employee 1 Details:");
        System.out.println("ID: " + bogdan.getEmpID());
        System.out.println("Name: " + bogdan.getEmpName());
        System.out.println("Designation: " + bogdan.getEmpDesignation());
        System.out.println();

        System.out.println("Employee 2 Details:");
        System.out.println("ID: " + bird.getEmpID());
        System.out.println("Name: " + bird.getEmpName());
        System.out.println("Designation: " + bird.getEmpDesignation());
    }
}
```

Exercise 02:

Develop the following class execute and discuss the answer: Please note that each class stored in separate files. Write down the answer.

```
class SuperB {
   int x;
   void setIt (int n) { x=n;}
   void increase () { x=x+1;}
   void triple () {x=x*3;};
   int returnIt () {return x;}
}
class SubC extends SuperB {
   void triple () {x=x+3;} // override existing method
   void quadruple () {x=x*4;} // new method
}
public class TestInheritance {
   public static void main(String[] args) {
      SuperB b = new SuperB();
      b.setIt(2);
      b.increase();
      b.triple();
      System.out.println( b.returnIt() );
      SubC c = new SubC();
      c.setIt(2);
      c.increase();
      c.triple();
```

```
        System.out.println( c.returnIt() ); }
}
```

*Output:

9

8

Exercise 03:

Recall the following scenario discussed during the class. Develop a code base to represent the scenario. Add a test class to invoke Lecturer and Student class by creating atleast one object from each.

Note: All the common attributes and behavior stored in the super class and only the specific fields and behavior stored in subclasses.

| Student |
| --- |
| - name |
| - id |
| - course |
| + setName()/getName() |
| + setID()/getID() |
| + setCourse()/getCourse() |

| Lecturer |
| --- |
| - name |
| - id |
| - programme |
| + setName()/getName() |
| + setID()/getID() |
| + setProg()/getProg() |

| Person |
| --- |
| Identify field and attributes to be stored in this class |

Exercise 04

Develop the following class execute and discuss the answer: Please note that each public class stored in separate files. Write down the answer.

```
public class Animal{}

public class Mammal extends Animal{}

public class Reptile extends Animal{}
```

```
public class Dog extends Mammal{
  public static void main(String args[]){
    Animal a = new Animal();
    Mammal m = new Mammal();
    Dog d = new Dog();
    System.out.println(m instanceof Animal);
    System.out.println(d instanceof Mammal);
    System.out.println(d instanceof Animal);
  }
}
```

*Output:

```
true
true
true
```