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.

Answers

```
public class Employee {

   private int empID;

   private String empName;

   private String empDesignation;


   // Getters and Setters for empID

   public int getEmpID() {

      return empID;

   }

   public void setEmpID(int empID) {

      this.empID = empID;

   }

   // Getters and Setters for empName

   public String getEmpName() {

      return empName;

   }

   public void setEmpName(String empName) {

      this.empName = empName;

   }
```

```
    // Getters and Setters for empDesignation
    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 mrBogdan = new Employee();
        mrBogdan.setEmpID(1);
        mrBogdan.setEmpName("Bogdan");
        mrBogdan.setEmpDesignation("Software Engineer");

        // Create another Employee object for Ms. Bird
        Employee msBird = new Employee();
        msBird.setEmpID(2);
        msBird.setEmpName("Bird");
        msBird.setEmpDesignation("HR Manager");

        // Print the details using getters
```

```
        System.out.println("Employee 1:");
        System.out.println("ID: " + mrBogdan.getEmpID());
        System.out.println("Name: " + mrBogdan.getEmpName());
        System.out.println("Designation: " + mrBogdan.getEmpDesignation());

        System.out.println("\nEmployee 2:");
        System.out.println("ID: " + msBird.getEmpID());
        System.out.println("Name: " + msBird.getEmpName());
        System.out.println("Designation: " + msBird.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() ); }
}
```

Answers

Let's discuss the code step-by-step:

1.We have two classes '**SuperB**' and **'SubC'**, each stored in separate files.

2.**'SuperB'** class:

- It has an instance variable **'x'** of type **'int'**.
- It has setter methods **'setIt(int n)'** and **'increase()'** to set the value of **'x'** and increase it by 1, respectively.
- It has a method **'triple()'** to multiply the value of **'x'** by 3.
- It has a method **'returnIt()'** to return the value of **'x'.**

3.**'SubC'** class:

- It extends the **'SuperB'** class, which means it inherits all the members (fields and methods) of **'SuperB'**.
- It overrides the **'triple()'** method from the **'SuperB'** class, so when called on a **'SubC'** object, it will add 3 to the value of **'x**' instead of multiplying it by 3.
- It has a new method **'quadruple()'** to multiply the value of **'x'** by 4.

4.TestInheritance class:

- It has the **'main'** method, where we create objects of **'SuperB'** and **'SubC'**.
- We first create a **'SuperB'** object **'b'**, set its value to 2, increase it by 1, and then triple it (multiply by 3). So, **'x'** will be (2+1) * 3 = 9.
- We print the value of **'x'** using the **'returnIt()'** method of **'SuperB'**, which will print 9.
- Next, we create a **'SubC'** object **'c',** set its value to 2, increase it by 1, and then triple it (add 3). So, **'x'** will be (2+1) + 3 = 6.
- We print the value of **'x'** using the **'returnIt()'** method of **'SubC'**, which will print 6.