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.

class Employee {

private int empID;

private String empName;

private String empDesignation;

public void setEmpID(int empID) {

this.empID = empID;

}

public int getEmpID() {

return empID;

}

public void setEmpName(String empName) {

this.empName = empName;

}

public String getEmpName() {

return empName;

}

public void setEmpDesignation(String empDesignation) {

this.empDesignation = empDesignation;

}

public String getEmpDesignation() {

return empDesignation;

}

}

class TestEmployee {

public static void main(String[] args) {

Employee employee1 = new Employee();

employee1.setEmpID(1001);

employee1.setEmpName("Bogdan");

employee1.setEmpDesignation("Software Engineer");

Employee employee2 = new Employee();

employee2.setEmpID(1002);

employee2.setEmpName("Bird");

employee2.setEmpDesignation("QA Engineer");

System.out.println("Employee 1 details:");

System.out.println("Employee ID: " + employee1.getEmpID());

System.out.println("Employee Name: " + employee1.getEmpName());

System.out.println("Employee Designation: " + employee1.getEmpDesignation());

System.out.println("\nEmployee 2 details:");

System.out.println("Employee ID: " + employee2.getEmpID());

System.out.println("Employee Name: " + employee2.getEmpName());

System.out.println("Employee Designation: " + employee2.getEmpDesignation());

}

}

**The Output:**

Employee 1 details:

Employee ID: 1001

Employee Name: Bogdan

Employee Designation: Software Engineer

Employee 2 details:

Employee ID: 1002

Employee Name: Bird

Employee Designation: QA Engineer

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() ); }

}

**The code:**

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()); // prints 9

SubC c = new SubC();

c.setIt(2);

c.increase();

c.triple();

System.out.println(c.returnIt()); // prints 11

}

}

**The Output:**

9

11

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 |

**The Code:**

class Person {

protected String name;

protected int id;

public Person(String name, int id) {

this.name = name;

this.id = id;

}

public String getName() {

return name;

}

public void setName(String name) {

this.name = name;

}

public int getId() {

return id;

}

public void setId(int id) {

this.id = id;

}

}

class Student extends Person {

private String course;

public Student(String name, int id, String course) {

super(name, id);

this.course = course;

}

public String getCourse() {

return course;

}

public void setCourse(String course) {

this.course = course;

}

}

class Lecturer extends Person {

private String programme;

public Lecturer(String name, int id, String programme) {

super(name, id);

this.programme = programme;

}

public String getProgramme() {

return programme;

}

public void setProgramme(String programme) {

this.programme = programme;

}

}

class TestPerson {

public static void main(String[] args) {

Student student = new Student("John Doe", 1001, "Computer Science");

Lecturer lecturer = new Lecturer("Jane Doe", 1002, "Software Engineering");

System.out.println("Student name: " + student.getName());

System.out.println("Student ID: " + student.getId());

System.out.println("Student course: " + student.getCourse());

System.out.println("\nLecturer name: " + lecturer.getName());

System.out.println("Lecturer ID: " + lecturer.getId());

System.out.println("Lecturer programme: " + lecturer.getProgramme());

}

}

**The Output:**

Student name: John Doe

Student ID: 1001

Student course: Computer Science

Lecturer name: Jane Doe

Lecturer ID: 1002

Lecturer programme: Software Engineering

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);

}

}

**The Code:**

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); // true

System.out.println(d instanceof Mammal); // true

System.out.println(d instanceof Animal); // true

}

}

The output of the code shows that the following statements are true:

* m instanceof Animal: m is an instance of Animal because Mammal is a subclass of Animal.
* d instanceof Mammal: d is an instance of Mammal because it is an object of the Dog class, and Dog is a subclass of Mammal.
* d instanceof Animal: d is an instance of Animal because Dog is a subclass of Animal.