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;

// Getters

public int getEmpID() {

return empID;

}

public String getEmpName() {

return empName;

}

public String getEmpDesignation() {

return empDesignation;

}

// Setters

public void setEmpID(int empID) {

this.empID = empID;

}

public void setEmpName(String empName) {

this.empName = empName;

}

public void setEmpDesignation(String empDesignation) {

this.empDesignation = empDesignation;

}

}

\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\

public class TestEmployee {

public static void main(String[] args) {

// Create two Employee objects

Employee mrBogdan = new Employee();

Employee msBird = new Employee();

// Set values using setters

mrBogdan.setEmpID(101);

mrBogdan.setEmpName("Mr. Bogdan");

mrBogdan.setEmpDesignation("Software Engineer");

msBird.setEmpID(102);

msBird.setEmpName("Ms. Bird");

msBird.setEmpDesignation("Product Manager");

// Print values using getters

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

System.out.println("Employee Name: " + mrBogdan.getEmpName()); System.out.println("Employee Designation: " + mrBogdan.getEmpDesignation()); System.out.println("\nEmployee ID: " + msBird.getEmpID());

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

System.out.println("Employee 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() ); }

}

((2 + 1) \* 3) = 9

((2 + 1) + 3) = 6

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 |

// Person class

class Person {

private String name;

private int id;

// Constructor

public Person(String name, int id) {

this.name = name;

this.id = id;

}

// Getters and setters

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;

}

}

// Student class (subclass of Person)

class Student extends Person {

private String course;

// Constructor

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

super(name, id);

this.course = course;

}

// Getter and setter for course

public String getCourse() {

return course;

}

public void setCourse(String course) {

this.course = course;

}

}

// Lecturer class (subclass of Person)

class Lecturer extends Person {

private String programme;

// Constructor

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

super(name, id);

this.programme = programme;

}

// Getter and setter for programme

public String getProg() {

return programme;

}

public void setProg(String programme) {

this.programme = programme;

}

}

// Test class to invoke Lecturer and Student classes

public class Main {

public static void main(String[] args) {

// Creating a Student object

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

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

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

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

System.out.println();

// Creating a Lecturer object

Lecturer lecturer = new Lecturer("Jane Smith", 2001, "Mathematics");

System.out.println("Lecturer Name: " + lecturer.getName());

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

System.out.println("Lecturer Programme: " + lecturer.getProg());

}

}

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);

}

}

\\\\\\\\\\\\\

Out put:

true

true

true