**NATIONAL UNIVERSITY OF TECHNOLOGY, ISLAMABAD  
  
OBJECT ORIENTED PROGRAMMING LAB  
  
MID SEMESTER EXAM**

**NUTECH ID: F24605037  
  
Submitted by: FARIAH HAJRA**

**Instructor Name: Ms. Tayyaba Kalsoom  
  
Course: CS123  
  
Session: Fall 24  
  
Submission Date: 23rd April, 2025**

# **LAB TASKS: INHERITANCE|**

**Question 01:**

**For default constructor (it is created only when no constructor is present or created):  
CODE:**

**class** Student {

**int** stdId;

    String stdName;

*// Default constructor is made hen no constructor is initiallized*

*/\**

*// Parameterized constructor*

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

*this.stdId = id;*

*this.stdName = name;*

*}*

*// Copy constructor*

*public Student(Student std1) {*

*this.stdId = std1.stdId;*

*this.stdName = std1.stdName;*

*}\*/*

}

**public** **class** StudentInfo {

**public** **static** **void** main(String[] args) {

*// Create Student objects*

        Student obj1 = new Student();

*// Student obj2 = new Student(56, "Fariah");*

*//Student obj3 = new Student(std1);*

*// Print details of Student 01*

        System.out.println("Details of Default Student 01:");

        display(obj1);

*// Print details of Student 02*

*// System.out.println("Details of Student 02:");*

*//display(obj2);*

*// Print details of Student 03*

*// System.out.println("Details of Student 03(copy costructor):");*

*//display(obj3);*

    }

**static** **void** display(Student object){

        System.out.println("Name: " + object.stdName);

        System.out.println("Student ID: " + object.stdId);

    }

}

**OUTPUT:  
  
A blue screen with white text

AI-generated content may be incorrect.Parameterized Constructor Code:**

**Question 02:**

Calculate bank account balance after interest through java. Create java class that has the following attributes:

* name (String)
* age (int)
* Private account Number (int)
* Private balance (double)

**Requirements:**

1. Create a class with the mentioned fields and apply proper **encapsulation**.
2. Add **constructors** (parameterized) to initialize account holder details.
3. Add a method deposit(), applyInterest(), calculateInterest(), showInfo of account Holder

**CODE:**

import java**.**util**.**Scanner;

**class** emp{

*//incapsulation for data hiding*

    String name;

**private** **int** accNum;

**private** **double** balance;

*//get and set method*

**void** getaccNum(**int** accNum){

        this.accNum = accNum;

    }

**int** setaccNum(){

        return accNum;

    }

**void** setBal(**double** balance){

        this.balance = balance;

    }

**double** getBal(){

        return balance;

    }

**void** depos(**double** depositAmount){

        if (depositAmount >  0){

            balance += depositAmount;

        }

        else{

            System.err.println("Error:Deposit can not be less than 0");

        }  
 }

**private** **double** calculateInterest(**int** age){

**double** interest;

        if(age > 50){

            interest = balance \* 0.5;*//50% balance*

        }

        else{

            interest = balance \* 0.10;*//10% increase in interest*

        }

        return interest;

    }

**void** applyInterest(**int** age){*//current balance*

*//retrieving private method by calling in same class get/set method is not needed*

        balance += calculateInterest(age);

    }

    emp(String name,**int** accNum, **double** balance){

        this.name =  name;

        this.accNum = accNum;

        this.balance = balance;

    }

**void** showInfo(){

        System.out.println("-------Displaying Account Information---------");

        System.out.println("Bank Account Holder Name: "+name);

        System.out.println("Bank account number: "+accNum);

        System.out.println("Balance: "+balance);

    }

}

**public** **class** BankAccount{

**public** **static** **void** main(String[] args){

        Scanner s = new Scanner(System.in);

        System.out.println("Task 02:");

        System.out.print("Enter  Name: ");

        String name = s.nextLine(); *//calling String class to store string input*

        System.out.print("Enter account Number: ");

**int** accNum = s.nextInt();

        System.out.print("Enter age: ");

**int** age = s.nextInt();

        System.out.print("Enter balance: ");

**double** balance = s.nextDouble();

        System.out.print("Enter deposit: ");

**double** deposit = s.nextDouble();

        emp e = new emp(name, accNum, balance);

        e.showInfo();

        System.out.println("\nAfter applying deposit:");

        e.depos(deposit);

        e.showInfo();

System.out.println("\nAfter applying interest:");

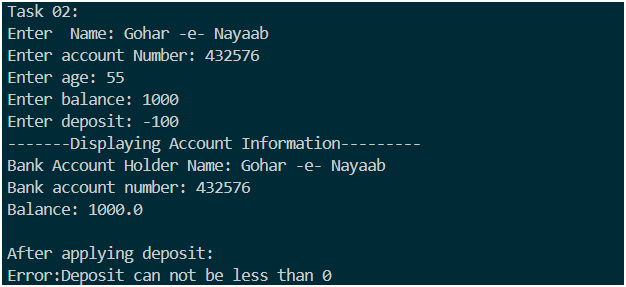
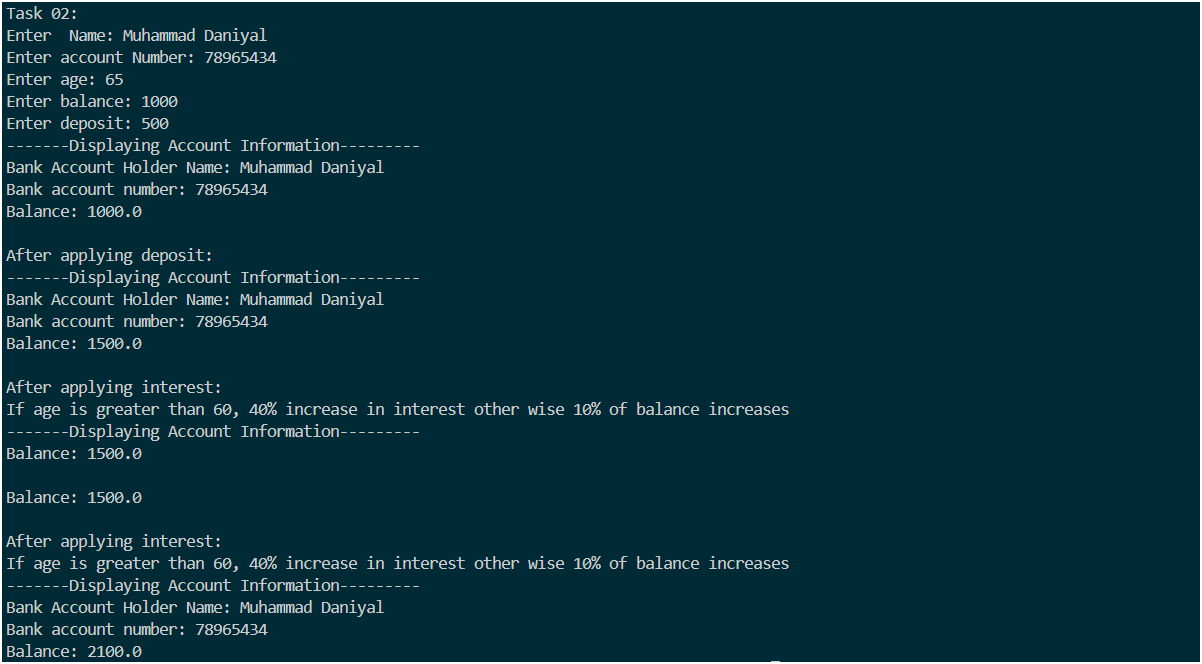
        System.out.println("If age is greater than 50% increase in interest other wise 10% of balance increases");

        e.applyInterest(age);

        e.showInfo();

    }

}

**Output:**If deposit amount is less than zero: ****Calculating Interest and Applying it on the basis of age:  


**Question 2:**

Write two Java classes: Employee and Manager.

The Employee class should have:

* A protected string variable named name.
* A private double variable named salary.
* A method named calculateBonus() that returns a double representing 10% of the salary.
* A method named displayInfo() that prints the name and salary of the employee to the console.

The Manager class should:

* Inherit from the Employee class.
* Have a private integer variable named teamSize.
* Override the calculateBonus() method to return a double representing 20% of the salary.
* Override the displayInfo() method to first call the displayInfo() method of the Employee class and then print the teamSize of the manager to the console.

In the main method of a separate class, create an instance of the Manager class and call its displayInfo() and calculateBonus() methods to demonstrate its specific behavior.

**Solution:**

class Employee {

protected String name;

private double salary;

// Constructor

Employee() {

this.name = "Unknown";

this.salary = 0;

}

Employee(String name, double salary) {

this.name = name;

this.salary = salary;

}

// 10% bonus

public double calculateBonus() {

return salary \* 0.10;

}

// Method to display employee info

public void displayInfo() {

System.out.println("\nName of employee: " + name);

System.out.println("Salary of employee: " + salary);

}

// Getter

protected double getSalary() {

return salary;

}

}

class Manager extends Employee {

private int teamSize;

// Constructor

Manager(String name, double salary, int teamSize) {

super(name, salary);

this.teamSize = teamSize;

}

// giving 20% bonus using overridden method

@Override

public double calculateBonus() {

return getSalary() \* 0.20;

}

// team size displayed using overridden method

@Override

public void displayInfo() {

super.displayInfo();

System.out.println("Team Size: " + teamSize);

}

}

public class Main2 {

public static void main(String[] args) {

System.out.println("Sarah Shahzad F24605008");

System.out.println("Question: 2");

Manager m1 = new Manager("Sarah", 9800, 7);

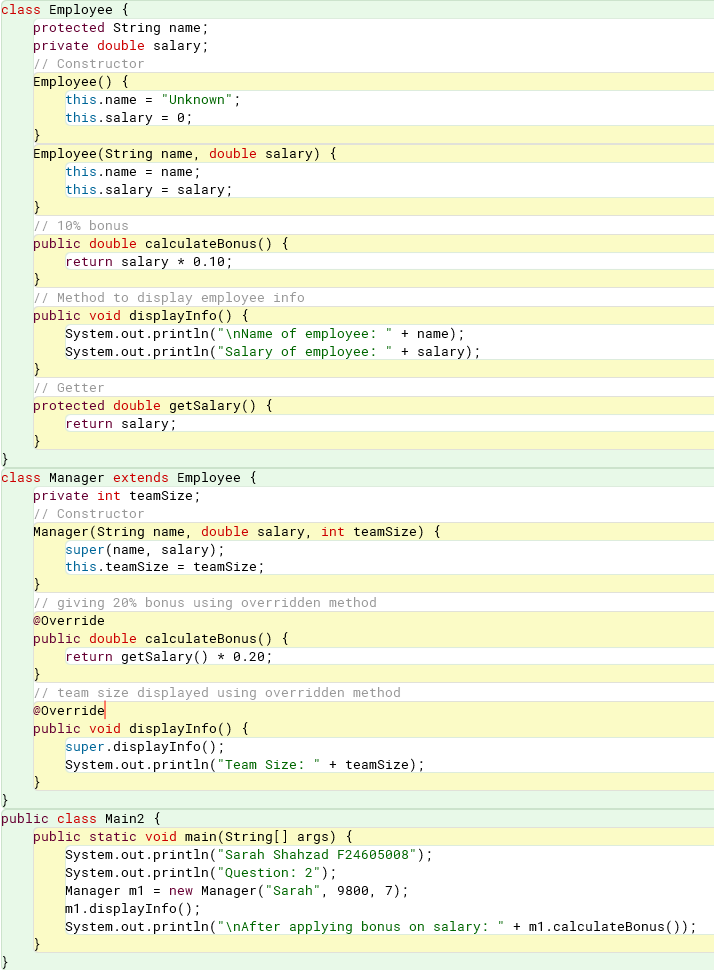
m1.displayInfo();

System.out.println("\nAfter applying bonus on salary: " + m1.calculateBonus());

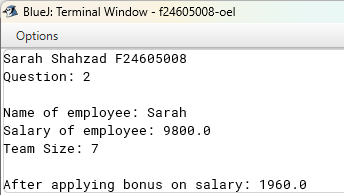
}

}

**CODE:**



**OUTPUT:**



**Question 3:**

Write the Java code to implement the Vehicle, LandVehicle, and Car classes, demonstrating multi-level inheritance and **method overriding** of the move() method. The move() method should have different implementations in LandVehicle and Car, reflecting their specific ways of moving. The displayCarInfo() method in Car should implicitly use the overridden move() method. Finally, create a Main class to instantiate a Car object and call its startEngine(), move(), and displayCarInfo() methods to showcase the inherited and overridden functionalities.

// Vehicle class

class Vehicle {

protected String type;

public Vehicle(String type) {

this.type = type;

}

public void startEngine() {

System.out.println(type + " engine has started successfully");

}

public void move() {

System.out.println(type + " is moving in Vehicle class ");

}

}

//Land vehicle class

class LandVehicle extends Vehicle {

protected int year;

public LandVehicle(String type, int year) {

super(type);

this.year = year;

}

public void move() {

System.out.println(type + " is from the year " + year );

}

}

//Car class

class Car extends LandVehicle {

private String model;

public Car(String model) {

super("Car", 2023);

this.model = model;

}

public void move() {

System.out.println(model + " is now moving in Car class");

}

public void displayCarInfo() {

startEngine();

System.out.println("Car Model: " + model);

System.out.println("Car Year: " + year);

move(); //overriden method

}

}

public class Vehicle1 {

public static void main(String[] args) {

System.out.println("Sarah Shahzad F24605008");

System.out.println("Question: 3\n");

Car car1= new Car("Honda City");

//inheritance and overriding methods

System.out.println("Vehicle 1:");

car1.startEngine();

car1.move();

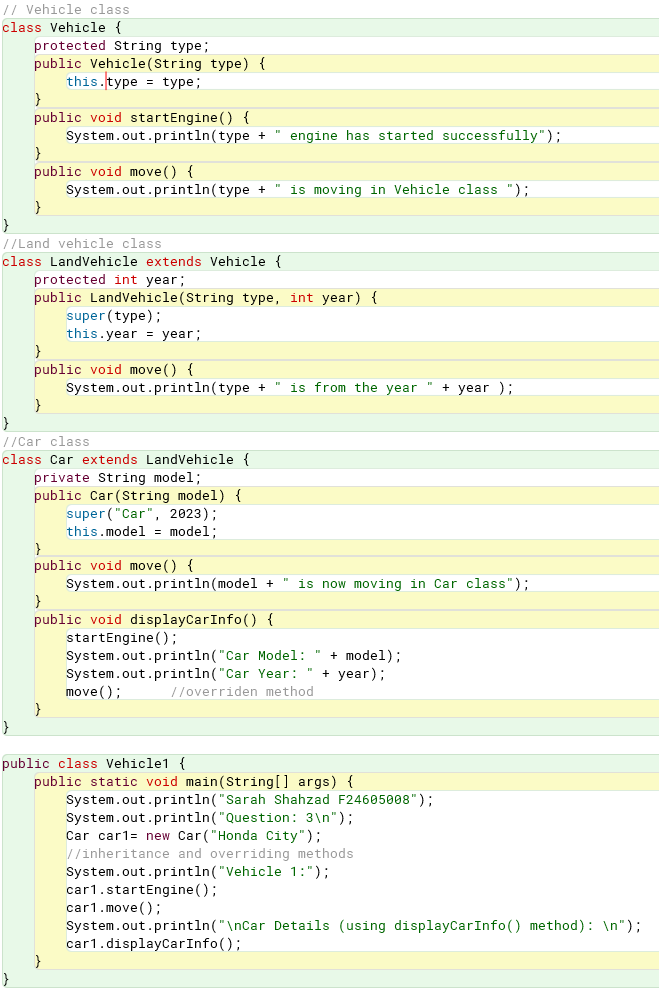
System.out.println("\nCar Details (using displayCarInfo() method): \n");

car1.displayCarInfo();

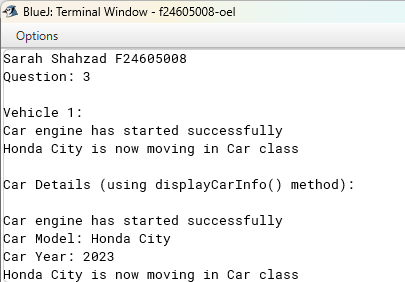
}

}

**CODE:**



**OUTPUT:**



**Question 4:**

Create a *Employee Class* with

* Attributes – name, salary, hours of work
* Constructor – to initialize the attributes
* Methods – showInfo(), addToSalary(amount), deductFromSalary(amount)
  + showInfo() – display whole information of Employee
  + addToSalary(amount) – increase salary by 20% if hours of work is greater than 8
  + deductFromSalary(amount) – deduct 5% if hours of work is less than 5
* Write main method to check the functionality of the class

class Employee1 {

String name;

double salary;

int Workhours;

Employee1(String name, double salary, int Workhours) {

this.name = name;

this.salary = salary;

this.Workhours = Workhours;

}

void showInfo() {

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

System.out.println("Salary: $" + salary);

System.out.println("Hours of Work: " + Workhours);

System.out.println();

}

void addToSalary() {

if (Workhours > 8) {

salary += salary \* 0.20;

System.out.println("Salary increased by 20% due to extra working hours.");

}

}

void deductFromSalary() {

if (Workhours < 5) {

salary -= salary \* 0.05;

System.out.println("Salary reduced by 5% due to insufficient working hours.\n");

}

}

public static void main(String[] args) {

System.out.println("Sarah Shahzad F24605008");

System.out.println("Question: 4\n");

Employee1 emp1 = new Employee1("Haneen", 5000, 12);

Employee1 emp2 = new Employee1("Humaira", 4000, 4);

System.out.println("Before Salary Adjustments:\n");

emp1.showInfo();

emp2.showInfo();

emp1.addToSalary();

emp2.deductFromSalary();

System.out.println("After Salary Adjustments:\n");

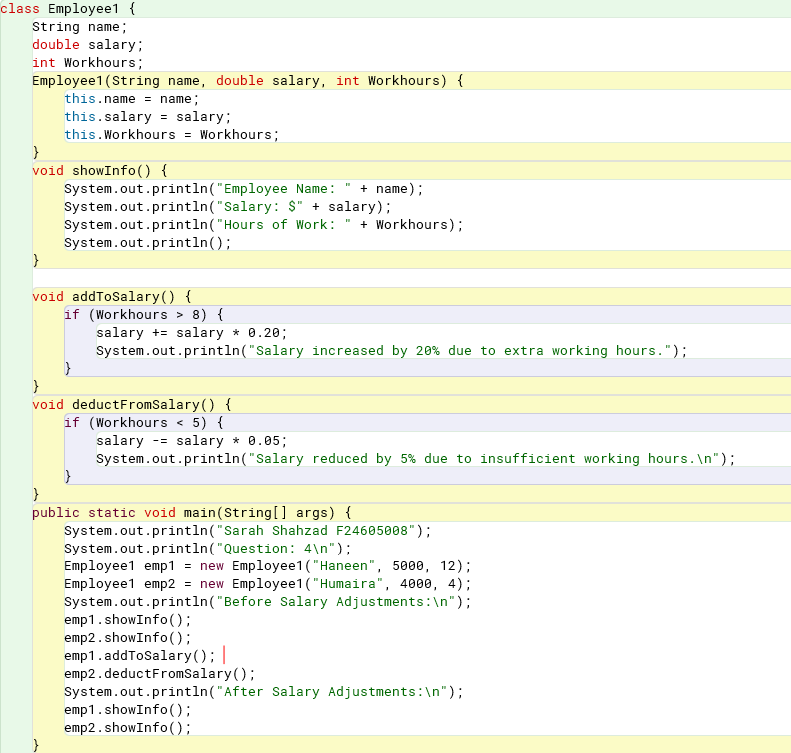
emp1.showInfo();

emp2.showInfo();

}

}

**CODE:**



**OUTPUT:**

