LAB3—NAMIRA MULLA

1.Create a superclass Person with attributes name and age, and a method display(). Create a subclass Student that adds an attribute studentID. Write a program to create a Student object and display all its attributes.

Code:

```
package lab3;

//Superclass Person
class Person {
 private String name;
 private int age;

 // Constructor
 public Person(String name, int age) {
     this.name = name;
     this.age = age;
 }

 // Method to display person's attributes
 public void display() {
     System.out.println("Name: " + name);
     System.out.println("Age: " + age);
 }
}

//Subclass Student
class Student extends Person {
 private String studentID;

 // Constructor
 public Student(String name, int age, String studentID) {
     super(name, age); // Call the superclass constructor
     this.studentID = studentID;
 }

 // Overriding display method
 @Override
 public void display() {
     super.display(); // Call the display method from Person
     System.out.println("Student ID: " + studentID);
 }
}

//Main class to test the implementation
public class Main {
 public static void main(String[] args) {
     // Create a Student object
     Student student = new Student("Namira", 20, "AF0407752");
```

```
        // Display all attributes
        student.display();
    }
}
```

Output:

2.Create a superclass Calculator with a method add(int a, int b). Create a subclass AdvancedCalculator that overloads the add method to handle three integers.

Code:

```
package lab3;
class Calculator {
        public int add(int a, int b) {
                return a + b;
        }
}
class AdvancedCalculator extends Calculator {
        // Overloaded method
        public int add(int a, int b, int c) {
                return a + b + c;
        }
}
public class SuperclassCalculator {
        public static void main(String[] args) {
                Calculator basicCalculator = new Calculator();
                AdvancedCalculator advancedCalculator = new
AdvancedCalculator();
                int sumTwo = basicCalculator.add(5, 10);
                System.out.println("Sum of two numbers: " + sumTwo);
                int sumThree = advancedCalculator.add(5, 10, 15);
                System.out.println("Sum of three numbers: " + sumThree);
        }
}
```

OUTPUT:

```
Problems  @ Javadoc  Declaration  Console ×
<terminated> SuperclassCalculator [Java Application] C:\Progra
Sum of two numbers: 15
Sum of three numbers: 30
```

3.Create a superclass Vehicle with a method move(). Create subclasses Car and Bike that inherit from Vehicle. Write a program to create objects of Car and Bike and call the move() method on each.

CODE:

```java
package lab3;
//Superclass Vehicle
class Vehicle {
	public void move() {
		System.out.println("The vehicle is moving");
	}
}
//Subclass Car
class Car extends Vehicle {
	@Override
	public void move() {
		System.out.println("The car is driving");
	}
}
//Subclass Bike
class Bike extends Vehicle {
	@Override
	public void move() {
		System.out.println("The bike is cycling");
	}
}
//Main class
public class SuperclassVehicle  {
	public static void main(String[] args) {
		// Create objects of Car and Bike
		Vehicle myCar = new Car();
		Vehicle myBike = new Bike();
		myCar.move();
		myBike.move();
	}
}
```

Output:

```
Problems  @ Javadoc  Declaration  Console ×
<terminated> SuperclassVehicle [Java Application] C:\Program Files\Java
The car is driving
The bike is cycling
```

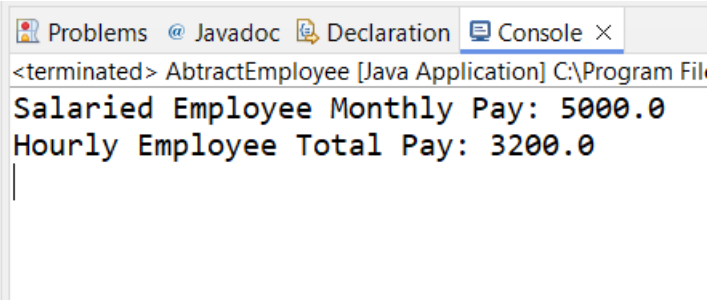
4.Create an class Employee with an abstract method calculatePay(). Create subclasses SalariedEmployee and HourlyEmployee that implement the calculatePay() method. Write a program to create objects of both subclasses and call the calculatePay() method.

Code:

```java
package lab3;
//Abstract class Employee
abstract class Employee {
	public abstract double calculatePay();
}
//Subclass SalariedEmployee
class SalariedEmployee extends Employee {
	private double annualSalary;
	// Constructor
	public SalariedEmployee(double annualSalary) {
		this.annualSalary = annualSalary;
	}
	@Override
	public double calculatePay() {
		return annualSalary / 12; // Monthly salary
	}
}
//Subclass HourlyEmployee
class HourlyEmployee extends Employee {
	private double hourlyRate;
	private int hoursWorked;
	public HourlyEmployee(double hourlyRate, int hoursWorked) {
		this.hourlyRate = hourlyRate;
		this.hoursWorked = hoursWorked;
	}
	@Override
	public double calculatePay() {
		return hourlyRate * hoursWorked; // Total pay
	}
}
//Main class
public class AbtractEmployee {
```

```
	public static void main(String[] args) {
		Employee salariedEmployee = new SalariedEmployee(60000);
// Annual salary
		Employee hourlyEmployee = new HourlyEmployee(20, 160); //
$20/hour for 160 hours
		System.out.println("Salaried Employee Monthly Pay: " +
salariedEmployee.calculatePay());
		System.out.println("Hourly Employee Total Pay: " +
hourlyEmployee.calculatePay());
	}
}
```

Output:



5.Create an class Document with an method void open(). Implement subclasses WordDocument, PDFDocument, and SpreadsheetDocument that extend Document and provide implementations for open(). Write a main class to demonstrate opening different types of documents.(implement complile time- polymorphism).

Code:

```
package lab3;
//Superclass Document
class Document {
	// Method to open the document
	public void open() {
		System.out.println("Opening a generic document.");
	}
}
//Subclass WordDocument
class WordDocument extends Document {
	// Overriding the open method
	@Override
	public void open() {
		System.out.println("Opening a Word document.");
	}
}
//Subclass PDFDocument
class PDFDocument extends Document {
```

```
	@Override
	public void open() {
		System.out.println("Opening a PDF document.");
	}
}
//Subclass SpreadsheetDocument
class SpreadsheetDocument extends Document {
	// Overriding the open method
	@Override
	public void open() {
		System.out.println("Opening a Spreadsheet document.");
	}
}
//Main class
public class Main2 {
	public static void main(String[] args) {
		Document[] documents = new Document[3];
		documents[0] = new WordDocument();
		documents[1] = new PDFDocument();
		documents[2] = new SpreadsheetDocument();
		for (Document doc : documents) {
			doc.open(); // Calls the overridden method for each
specific document type
		}
	}
}
```

Output:

```
Problems  @ Javadoc  Declaration  Console ×
<terminated> Main2 [Java Application] C:\Program Files\Java\jd
Opening a Word document.
Opening a PDF document.
Opening a Spreadsheet document.
```

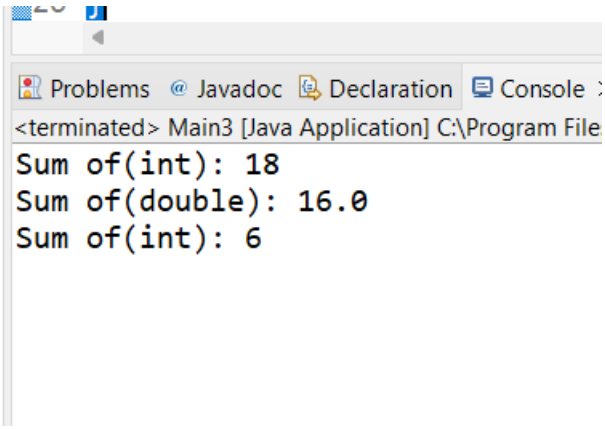
6.Create a class Calculator with overloaded methods add() that take different numbers and types of parameters: int add(int a, int b), double add(double a, double b), int add(int a, int b, int c) Write a main class to demonstrate the usage of these methods.

Code:

```
package lab3;
//Calculator class
class Calculat {
	// Method to add two integers
	public int add(int a, int b) {
```

```
            return a + b;
        }
        public double add(double a, double b) {
            return a + b;
        }
        public int add(int a, int b, int c) {
            return a + b + c;
        }
}
public class Main3 {
        public static void main(String[] args) {
            Calculat calc = new Calculat();
            // Demonstrate adding two integers
            int sum1 = calc.add(8, 10);
            System.out.println("Sum of(int): " + sum1);
            double sum2 = calc.add(5.5, 10.5);
            System.out.println("Sum of(double): " + sum2);
            int sum3 = calc.add(1, 2, 3);
            System.out.println("Sum of(int): " + sum3);
        }
}
```

Output:



7.Create a JavaBean class Person with properties firstName, lastName, age, and email. Implement the required no-argument constructor, getter and setter methods for each property. Write a main class to create an instance of Person, set its properties, and print them out.

Code:

```
package lab3;
import java.io.Serializable;
class Person1 implements Serializable {
        private String firstName;
        private String lastName;
        private int age;
```

```java
        private String email;
        // creatingg constructor
        public Person1() {
        }
        // Getter and Setter for firstName
        public String getFirstName() {
                return firstName;
        }
        public void setFirstName(String firstName) {
                this.firstName = firstName;
        }
        // Getter and Setter for lastName
        public String getLastName() {
                return lastName;
        }
        public void setLastName(String lastName) {
                this.lastName = lastName;
        }
        // Getter and Setter for age
        public int getAge() {
                return age;
        }
        public void setAge(int age) {
                this.age = age;
        }
        // Getter and Setter for email
        public String getEmail() {
                return email;
        }
        public void setEmail(String email) {
                this.email = email;
        }
}
public class InheritanceeDemo {
        public static void main(String[] args) {
                // Create an instance of Person
                Person1 person = new Person1();

                person.setFirstName("Namira");
                person.setLastName("mulla");
                person.setAge(20);
                person.setEmail("mullarukasa@gmail");

                System.out.println("First Name: " +
person.getFirstName());
                System.out.println("Last Name: " + person.getLastName());
                System.out.println("Age: " + person.getAge());
                System.out.println("Email: " + person.getEmail());
        }
```

```
}
```

Output:

```
Problems  @ Javadoc  Declaration  Console ×
<terminated> InheritanceeDemo [Java Application] C:\Progra
First Name: Namira
Last Name: mulla
Age: 20
Email: mullarukasa@gmail
```

8.Create a JavaBean class Car with properties make, model, year, and color. Implement the required no-argument constructor, getter and setter methods for each property. Write a main class to create an instance of Car, set its properties, and print the car details

Code:

```
package lab3;
import java.io.Serializable;
class Cars implements Serializable {
	private String make;
	private String model;
	private int year;
	private String color;
	public Cars() {}
	public String getMake() {
		return make;
	}
	// Setter for make
	public void setMake(String make) {
		this.make = make;
	}
	// Getter for model
	public String getModel() {
		return model;
	}
	// Setter for model
	public void setModel(String model) {
		this.model = model;
	}
	// Getter for year
	public int getYear() {
		return year;
```

```java
        }
        // Setter for year
        public void setYear(int year) {
                this.year = year;
        }
        // Getter for color
        public String getColor() {
                return color;
        }
        // Setter for color
        public void setColor(String color) {
                this.color = color;
        }
}
public class tataMoters { // main class
        public static void main(String[] args) {
                // Create an object of Car
                Cars car = new Cars();
                // Seting thepropeerties of car
                car.setMake("Tata");
                car.setModel("Nexon");
                car.setYear(2024);
                car.setColor("Blue");
                System.out.println("Car Make: " + car.getMake());
                System.out.println("Car Model: " + car.getModel());
                System.out.println("Car Year: " + car.getYear());
                System.out.println("Car Color: " + car.getColor());
        }
}
```

Output:

```
Problems  @ Javadoc  Declaration  Console >
<terminated> tataMoters [Java Application] C:\Program
Car Make: Tata
Car Model: Nexon
Car Year: 2024
Car Color: Blue
```