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.

Program :

package Lab\_3;

//Define the 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 attributes

public void display() {

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

System.***out***.println("Age: " + age);

}

}

//Define the subclass Student

class Student extends Person {

private int studentID;

// Constructor

public Student(String name, int age, int studentID) {

super(name, age); // Call superclass constructor

this.studentID = studentID;

}

// Method Overriding

public void display() {

super.display(); // Call superclass display method

System.***out***.println("Student ID: " + studentID);

}

}

public class InheritanceExample {

public static void main(String[] args) {

Person student1 = new Student("Dinesh", 21,406844);

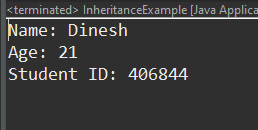
// Display all attributes of the student

student1.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.

Program :

package Lab\_3;

//Define the superclass Calculator

class Calculator {

// Method to add two integers

public int add(int a, int b) {

return a + b;

}

}

//Define the subclass AdvancedCalculator

class AdvancedCalculator extends Calculator {

// Overloaded method to add three integers

public int add(int a, int b, int c) {

return a + b + c;

}

}

public class Q2\_Calculator {

public static void main(String[] args) {

Calculator basicCalc = new Calculator();

AdvancedCalculator advCalc = new AdvancedCalculator();

// add method of Calculator

int result1 = basicCalc.add(10, 20);

System.***out***.println("Result of basicCalc.add(10, 20): " + result1);

// add method of AdvancedCalculator

int result2 = advCalc.add(10, 20, 30);

System.***out***.println("Result of advCalc.add(10, 20, 30): " + result2);

}

}

Output :

A screen shot of a computer

Description automatically generated

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.

Program :

package Lab\_3;

// superclass Vehicle

class Vehicle {

// Method to move

public void move() {

System.***out***.println("Vehicle is moving.");

}

}

//subclass Car

class Car extends Vehicle {

}

//subclass Bike

class Bike extends Vehicle {

}

public class Q3\_Vehical {

public static void main(String[] args) {

// Create objects of Car and Bike

Car car = new Car();

Bike bike = new Bike();

// Call the move() method on Car and Bike objects

car.move();

bike.move();

}

}

Output:

A screenshot of a computer

Description automatically generated

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.

Program :

package Lab\_3;

//Abstract superclass Employee

abstract class Employee {

// Abstract method to calculate pay

public abstract void calculatePay();

}

//Subclass SalariedEmployee

class SalariedEmployee extends Employee {

private String name;

private double monthlySalary;

// Constructor

public SalariedEmployee(String name, double monthlySalary) {

this.name = name;

this.monthlySalary = monthlySalary;

}

// Implementation of calculatePay() method

*@Override*

public void calculatePay() {

System.***out***.println("Calculating pay for Salaried Employee " + name);

System.***out***.println("Pay amount: " + monthlySalary);

}

}

//Subclass HourlyEmployee

class HourlyEmployee extends Employee {

private String name;

private double hoursSalary;

// Constructor

public HourlyEmployee(String name, double hoursWorked) {

this.name = name;

this.hoursSalary = hoursWorked;

}

// Implementation of calculatePay() method

*@Override*

public void calculatePay() {

System.***out***.println("Calculating pay for Hourly Employee " + name);

System.***out***.println("Pay amount: " + this.hoursSalary);

}

}

public class Q4\_Employee {

public static void main(String[] args) {

// Create objects of SalariedEmployee and HourlyEmployee

SalariedEmployee salariedEmp = new SalariedEmployee("Dinesh Chaudhary", 5000.0);

HourlyEmployee hourlyEmp = new HourlyEmployee("Dev Comm", 800.0);

// Call calculatePay() method

salariedEmp.calculatePay();

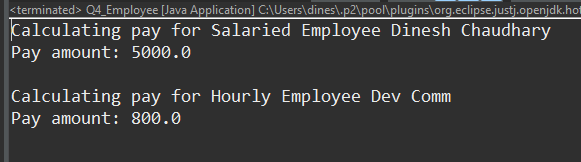
System.***out***.println();

hourlyEmp.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).

Program :

package Lab\_3;

// Define Document Class

class Document {

// Open Method

public void open() {

System.***out***.println("Document class Open method called.");

}

}

//Subclass WordDocument

class WordDocument extends Document {

*@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 {

*@Override*

public void open() {

System.***out***.println("Opening a Spreadsheet document.");

}

}

public class Q5\_CompileTimePolymorphism {

public static void main(String[] args) {

// Create instances of each document type

Document doc1 = new WordDocument();

Document doc2 = new PDFDocument();

Document doc3 = new SpreadsheetDocument();

// Demonstrate opening different types of documents

// Demonstrate Compile time polymorphism.

doc1.open();

doc2.open();

doc3.open();

}

}

Output :

A screen shot of a computer

Description automatically generated

 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.

Program :

// Calculator class with overloaded add() methods

package Lab\_3;

class Calculator2 {

// Method to add two integers

public int add(int a, int b) {

return a + b;

}

// Method to add two doubles

public double add(double a, double b) {

return a + b;

}

// Method to add three integers

public int add(int a, int b, int c) {

return a + b + c;

}

}

// Main class for testing

public class Q6\_Calculator {

public static void main(String[] args) {

Calculator2 calc = new Calculator2();

// Test each overloaded add() method

int sum1 = calc.add(10, 20);

double sum2 = calc.add(15.5, 20.3);

int sum3 = calc.add(10, 20, 30);

// Print results

System.***out***.println("Sum of 10 and 20 (int): " + sum1);

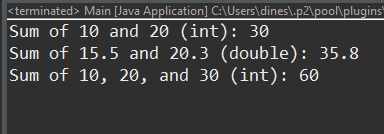
System.***out***.println("Sum of 15.5 and 20.3 (double): " + sum2);

System.***out***.println("Sum of 10, 20, and 30 (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.

Program :

package Lab\_3;

class Person {

// Properties

private String firstName;

private String lastName;

private int age;

private String email;

// No-argument constructor

public Person() {

}

// Getter and setter methods

public String getFirstName() {

return firstName;

}

public void setFirstName(String firstName) {

this.firstName = firstName;

}

public String getLastName() {

return lastName;

}

public void setLastName(String lastName) {

this.lastName = lastName;

}

public int getAge() {

return age;

}

public void setAge(int age) {

this.age = age;

}

public String getEmail() {

return email;

}

public void setEmail(String email) {

this.email = email;

}

}

public class Q7\_JavaBeanExample {

public static void main(String[] args) {

// Create an instance of Person

Person person = new Person();

// Set properties

person.setFirstName("Dinesh");

person.setLastName("Chaudhary");

person.setAge(22);

person.setEmail("dinesh.work.38@example.com");

// Print out the person's information

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 :

A screenshot of a computer

Description automatically generated

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.

Program :

package Lab\_3;

class Car {

// Properties

private String make;

private String model;

private int year;

private String color;

// No-argument constructor

public Car() {

}

// Getter and setter methods

public String getMake() {

return make;

}

public void setMake(String make) {

this.make = make;

}

public String getModel() {

return model;

}

public void setModel(String model) {

this.model = model;

}

public int getYear() {

return year;

}

public void setYear(int year) {

this.year = year;

}

public String getColor() {

return color;

}

public void setColor(String color) {

this.color = color;

}

}

public class Q8\_CarClass {

public static void main(String[] args) {

// Create an instance of Car

Car car = new Car();

// Set properties

car.setMake("Activa");

car.setModel("5G");

car.setYear(2026);

car.setColor("Black");

// Print out the car details

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 :

A screen shot of a computer

Description automatically generated