**LAB 3**

Q1.   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.

**Programe:**

**package** Demo;

**class** Person {

String name;

**int** age;

Person(String name, **int** age) {

**this**.name = name;

**this**.age = age;

}

**void** display() {

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

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

}

}

**class** Student **extends** Person {

String studentID;

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

**super**(name, age);

**this**.studentID = studentID;

}

**void** display() {

**super**.display();

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

}

}

**public** **class** Base {

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

Student student = **new** Student("Akash", 20, "S12345");

student.display();

}

}

**Output:**

Name: Akash

Age: 20

Student ID: S12345

Q2. 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.

**Programe:**

**package** Demo;

**class** Calculator {

// Method to add two integers

**int** add(**int** a, **int** b) {

**return** a + b;

}

}

**class** AdvancedCalculator **extends** Calculator {

// Overloaded method to add three integers

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

**return** a + b + c;

}

}

**public** **class** SuperclassCalculator {

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

Calculator calculator = **new** Calculator();

AdvancedCalculator advancedCalculator = **new** AdvancedCalculator();

// Using the add method from Calculator

System.***out***.println("Sum of 2 and 3: " + calculator.add(2, 3));

// Using the overloaded add method from AdvancedCalculator

System.***out***.println("Sum of 4, 5, and 6: " + advancedCalculator.add(4, 5, 6));

}

}

**Output:**

Sum of 2 and 3: 5

Sum of 4, 5, and 6: 15

Q3. 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.

**Programe:**

**package** Demo;

**class** Vehicle {

// Method to simulate movement

**void** move() {

System.***out***.println("This vehicle is moving.");

}

}

**class** Car **extends** Vehicle {

// Car-specific implementation of move, if necessary

**void** move() {

System.***out***.println("The car is moving on the road.");

}

}

**class** Bike **extends** Vehicle {

// Bike-specific implementation of move, if necessary

**void** move() {

System.***out***.println("The bike is moving on the track.");

}

}

**public** **class** BaseClass {

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

// **TODO** Auto-generated method stub

Car car = **new** Car();

Bike bike = **new** Bike();

// Call the move method on each object

car.move();

bike.move();

}

}

The car is moving on the road.

The bike is moving on the track.

Q4. 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.

**Programe:**

**package** Demo;

**abstract** **class** Employee {

**protected** String name;

**public** Employee(String name) {

**this**.name = name;

}

// Abstract method to be implemented by subclasses

**abstract** **double** calculatePay();

}

**class** SalariedEmployee **extends** Employee {

**private** **double** annualSalary;

**public** SalariedEmployee(String name, **double** annualSalary) {

**super**(name);

**this**.annualSalary = annualSalary;

}

// Implement calculatePay for salaried employees

@Override

**double** calculatePay() {

**return** annualSalary / 52; // Weekly pay

}

}

**class** HourlyEmployee **extends** Employee {

**private** **double** hourlyRate;

**private** **int** hoursWorked;

**public** HourlyEmployee(String name, **double** hourlyRate, **int** hoursWorked) {

**super**(name);

**this**.hourlyRate = hourlyRate;

**this**.hoursWorked = hoursWorked;

}

// Implement calculatePay for hourly employees

@Override

**double** calculatePay() {

**return** hourlyRate \* hoursWorked;

}

}

**public** **class** AbstactEmployee {

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

// **TODO** Auto-generated method stub

SalariedEmployee salariedEmployee = **new** SalariedEmployee("Samir Kumar", 52000);

HourlyEmployee hourlyEmployee = **new** HourlyEmployee("Samir Kumar", 20, 40);

System.***out***.println(salariedEmployee.name + "'s weekly pay: " + salariedEmployee.calculatePay());

System.***out***.println(hourlyEmployee.name + "'s weekly pay: " + hourlyEmployee.calculatePay());

}

}

**Output:**

Samir Kumar's weekly pay: 1000.0

Samir Kumar's weekly pay: 800.0

Q5. 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).

**Programe:**

**package** Demo;

//Base class

**class** Document {

**void** open() {

System.***out***.println("A generic document is opened.");

}

}

//Subclass for Word documents

**class** WordDocument **extends** Document {

@Override

**void** open() {

System.***out***.println("A Word document is opened with Microsoft Word.");

}

}

//Subclass for PDF documents

**class** PDFDocument **extends** Document {

@Override

**void** open() {

System.***out***.println("A PDF document is opened with Adobe Reader.");

}

}

//Subclass for Spreadsheet documents

**class** SpreadsheetDocument **extends** Document {

@Override

**void** open() {

System.***out***.println("A Spreadsheet document is opened with Microsoft Excel.");

}

}

//Main class to demonstrate opening different types of documents

**public** **class** DocumentPolymorphism {

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

// **TODO** Auto-generated method stub

Document wordDoc = **new** WordDocument();

Document pdfDoc = **new** PDFDocument();

Document spreadsheetDoc = **new** SpreadsheetDocument();

wordDoc.open();

pdfDoc.open();

spreadsheetDoc.open();

}

}

**Output:**

A Word document is opened with Microsoft Word.

A PDF document is opened with Adobe Reader.

A Spreadsheet document is opened with Microsoft Excel.

Q6. Create a class Calculator with overloaded methods add() that take different numbers and types of parameters: int add(int a, int b) .

**Programe:**

**package** Demo;

**public** **class** Calculator {

// Method to add two integers

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

**return** a + b;

}

// Method to add three integers

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

**return** a + b + c;

}

// Method to add two doubles

**public** **double** add(**double** a, **double** b) {

**return** a + b;

}

// Method to add a string and an integer

**public** String add(String str, **int** num) {

**return** str + num;

}

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

Calculator calculator = **new** Calculator();

System.***out***.println("Add two integers: " + calculator.add(12, 13));

System.***out***.println("Add three integers: " + calculator.add(12, 13, 15));

System.***out***.println("Add two doubles: " + calculator.add(10.5, 20.5));

System.***out***.println("Add a string and an integer: " + calculator.add("Hello, ", 2024));

}

}

**Output:**

Add two integers: 25  
Add three integers: 37  
Add two doubles: 31.0  
Add a string and an integer: Hello, 2024

Q7. 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.

**Programe:**

**i)**

**package** Demo;

**public** **class** Person {

**private** String firstName;

**private** String lastName;

**private** **int** age;

**private** String email;

// No-argument constructor

**public** Person() {}

// Getter and setter methods for each property

**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;

}

**}**

**ii)**

**package** Demo;

**public** **class** MainP {

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

// Create an instance of Person

Person person = **new** Person();

// Set the properties

person.setFirstName("Arun");

person.setLastName("Kumar");

person.setAge(30);

person.setEmail("arun.kumar@example.com");

// Print the properties

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:**

First Name: Arun  
Last Name: Kumar  
Age: 30  
Email: arun.kumar@example.com

Q8. 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.

**Programe:**

**i)**

**package** Demo;

**public** **class** Car {

**private** String make;

**private** String model;

**private** **int** year;

**private** String color;

// No-argument constructor

**public** Car() {

}

// Getters and setters for make

**public** String getMake() {

**return** make;

}

**public** **void** setMake(String make) {

**this**.make = make;

}

// Getters and setters for model

**public** String getModel() {

**return** model;

}

**public** **void** setModel(String model) {

**this**.model = model;

}

// Getters and setters for year

**public** **int** getYear() {

**return** year;

}

**public** **void** setYear(**int** year) {

**this**.year = year;

}

// Getters and setters for color

**public** String getColor() {

**return** color;

}

**public** **void** setColor(String color) {

**this**.color = color;

}

}

**ii)**

**class** Demo;

**public** **class** Main {

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

// Create an instance of Car

Car car = **new** Car();

// Set properties

car.setMake("Maruti");

car.setModel("Suzuki");

car.setYear(2020);

car.setColor("Red");

// Print car details

System.***out***.println("Car Details:");

System.***out***.println("Make: " + car.getMake());

System.***out***.println("Model: " + car.getModel());

System.***out***.println("Year: " + car.getYear());

System.***out***.println("Color: " + car.getColor());

}

}

**Output:**

Car Details:  
Make: Maruti  
Model: Suzuki  
Year: 2020  
Color: Red