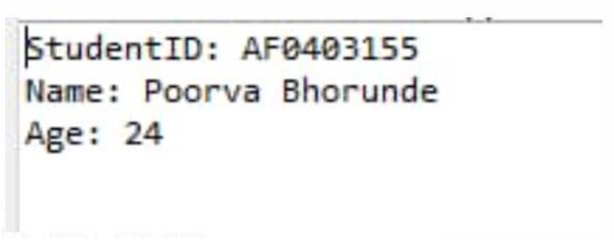
1. Create a superclass Person with attributes name and age, and a method display(). Create a subclass Student that adds an attribute studentID. Write a program to create a Student object and display all its attributes.

Program :

```
package demo;
public class person {// superclass person
	String name;// attributes of person class
	int age;
	public person(String name, int age) {
		this.name=name;
		this.age=age;
	}
	public static void main(String[] args) {
	}
}
//student.java
package demo;

public class Student  {//attribute for student class
	String studentID,name;
	int age;
	public Student(String name, int age, String studentID) {
		this.studentID=studentID;
		this.name=name;
		this.age=age;
	}
	public String getStudentID() {//get method
		return studentID;
	}
	public void display() {
		System.out.println("StudentID: "+studentID);
		System.out.println("Name: "+name);
		System.out.println("Age: "  +age);
	}
	public static void main(String[] args) {
		// creating student object
	  Student student = new Student("Poorva", 24, "AF0403155");
	  student.display();
	}
}
```

Output :



```
StudentID: AF0403155
Name: Poorva Bhorunde
Age: 24
```

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.

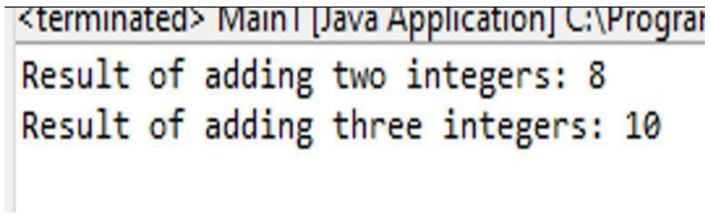
Program :

```
package demo;
//Superclass Calculator with a method add(int a, int b)
class Calculator {
 // Method to add two integers
public int add(int a, int b) {
return a + b;
 } }
package demo;
//Subclass AdvancedCalculator that inherits from Calculator class
AdvancedCalculator extends Calculator {
 // Overloading the add method to handle three integers
public int add(int a, int b, int c) {      return a +
b + c;
 } }
package demo;
//Main class to demonstrate the use of the Calculator classes
public class Main1 {
 public static void main(String[] args) {
     // Creating an instance of AdvancedCalculator
     AdvancedCalculator advancedCalculator = new AdvancedCalculator();

     // Using the add method of Calculator to add two integers
int result1 = advancedCalculator.add(5, 3);
     System.out.println("Result of adding two integers: " + result1);

     // Using the overloaded add method of AdvancedCalculator to add three integers
int result2 = advancedCalculator.add(5, 3, 2);
     System.out.println("Result of adding three integers: " + result2);
 }
}
```

Output :



```
<terminated> Main1 [Java Application] C:\Progra
Result of adding two integers: 8
Result of adding three integers: 10
```

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.

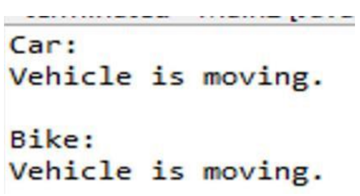
Program :

```
package demo;
//Superclass Vehicle with a method move()
class Vehicle {  // Method to move
public void move() {
     System.out.println("Vehicle is moving.");
 } }
package demo;
//Subclass Car that inherits from Vehicle class
Car extends Vehicle {
 // Additional attributes and methods specific to Car can be added here
}
package demo;
//Subclass Bike that inherits from Vehicle class
Bike extends Vehicle {
 // Additional attributes and methods specific to Bike can be added here
}
package demo;
//Main class to demonstrate the use of Vehicle, Car, and Bike classes
public class Main2{
 public static void main(String[] args) {
     // Creating objects of Car and Bike
     Car car = new Car();
     Bike bike = new Bike();

     // Calling the move method on Car object
     System.out.println("Car:");
car.move();

     // Calling the move method on Bike object
     System.out.println("\nBike:");
bike.move();
 }
}
```

Output :



```
Car:
Vehicle is moving.

Bike:
Vehicle is moving.
```

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 :

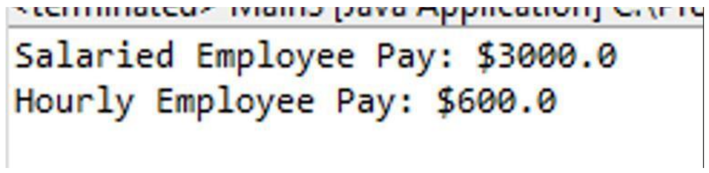
```
package demo;
//Superclass Employee with an abstract method calculatePay()
public abstract class Employee {  // Abstract method to
calculate pay  public abstract double calculatePay();
}
package demo;
//Subclass SalariedEmployee that inherits from Employee
class SalariedEmployee extends Employee {  private
double salary; // Monthly salary

 // Constructor
 public SalariedEmployee(double salary) {
this.salary = salary;
 }

 // Implementing the calculatePay method for SalariedEmployee
 @Override
 public double calculatePay() {
return salary;
 } }
package demo;
//Subclass HourlyEmployee that inherits from Employee
class HourlyEmployee extends Employee {  private
double hourlyRate; // Hourly rate  private int
hoursWorked;   // Hours worked
 // Constructor
 public HourlyEmployee(double hourlyRate, int hoursWorked) {
this.hourlyRate = hourlyRate;      this.hoursWorked =
hoursWorked;
 }
 // Implementing the calculatePay method for HourlyEmployee
 @Override
 public double calculatePay() {
return hourlyRate * hoursWorked;
 } }
package demo;
//Main class to demonstrate the use of SalariedEmployee and HourlyEmployee classes
public class Main3{
 public static void main(String[] args) {
     // Creating objects of both subclasses
     SalariedEmployee salariedEmployee = new SalariedEmployee(3000);
     HourlyEmployee hourlyEmployee = new HourlyEmployee(15, 40);

     // Calling the calculatePay method on objects
     System.out.println("Salaried Employee Pay: $" + 
salariedEmployee.calculatePay());
     System.out.println("Hourly Employee Pay: $" + hourlyEmployee.calculatePay());
 }
}
```

Output :



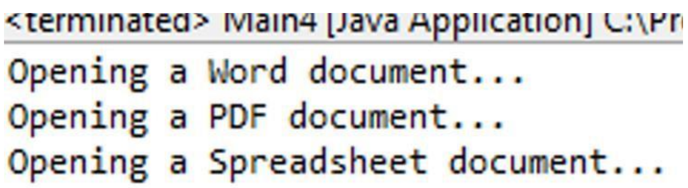
```
Salaried Employee Pay: $3000.0
Hourly Employee Pay: $600.0
```

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 demo;
//Superclass Document with a method open() class
Document {
 // Method to open a document
public void open() {
     System.out.println("Opening a generic document...");
 } }
package demo;
//Subclass WordDocument that extends Document class
WordDocument extends Document {
 // Implementing open method for Word documents
 @Override
 public void open() {
     System.out.println("Opening a Word document...");
 } }
package demo;
//Subclass PDFDocument that extends Document class
PDFDocument extends Document {
 // Implementing open method for PDF documents
 @Override
 public void open() {
     System.out.println("Opening a PDF document...");
 } }
package demo;
//Subclass SpreadsheetDocument that extends Document class
SpreadsheetDocument extends Document {
 // Implementing open method for Spreadsheet documents
 @Override
 public void open() {
     System.out.println("Opening a Spreadsheet document...");
 } }
package demo;
//Main class to demonstrate compile-time polymorphism
public class Main4{
 public static void main(String[] args) {
     // Creating instances of different types of documents
     Document doc1 = new WordDocument();
     Document doc2 = new PDFDocument();
     Document doc3 = new SpreadsheetDocument();
     // Demonstrating opening different types of documents using compile-time
polymorphism
     doc1.open(); // Opens a Word document
     doc2.open(); // Opens a PDF document
doc3.open(); // Opens a Spreadsheet document
 }
}
```

Output :



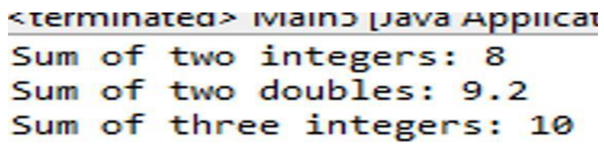
```
<terminated> Main4 [Java Application] C:\Pr
Opening a Word document...
Opening a PDF document...
Opening a Spreadsheet document...
```

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.

Program :

```
package demo;
//Calculator class with overloaded add() methods
public class Calculator1{  // 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;
 } }
package demo;
// Main class to demonstrate the usage of Calculator methods
public class Main5{
    public static void main(String[] args) {
        // Creating an instance of Calculator
        Calculator1 calculator = new Calculator1();
        // Adding two integers
        int sumInt = calculator.add(5, 3);
        System.out.println("Sum of two integers: " + sumInt);
        // Adding two doubles
        double sumDouble = calculator.add(5.5, 3.7);
        System.out.println("Sum of two doubles: " + sumDouble);
        // Adding three integers
        int sumThreeInt = calculator.add(5, 3, 2);
        System.out.println("Sum of three integers: " + sumThreeInt);
    }
}
```

Output :



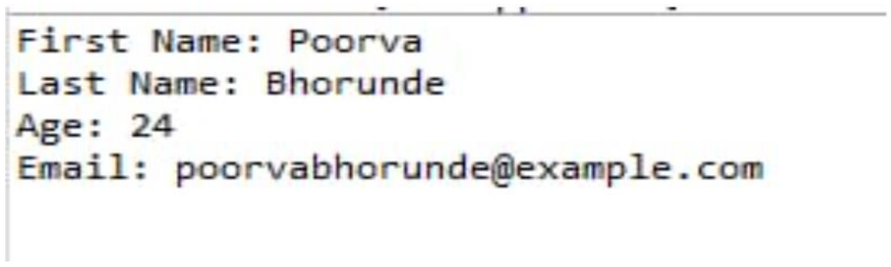
```
<terminated> Main5 [Java Applicat
Sum of two integers: 8
Sum of two doubles: 9.2
Sum of three integers: 10
```

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 demo;
    class Person {
    // Properties
    private String firstName;
    private String lastName;
    private int age;
    private String email;
    // no-argument constructor
    public Person() {
    }
    // Getter and setter methods for firstName
    public String getFirstName() {
        return firstName;
    }
    public void setFirstName(String firstName) {
        this.firstName = firstName;
    }
    // Getter and setter methods for lastName
    public String getLastName() {
        return lastName;
    }
    public void setLastName(String lastName) {
        this.lastName = lastName;
    }
    // Getter and setter methods for age
    public int getAge() {
        return age;
    }
    public void setAge(int age) {
        this.age = age;
    }
    // Getter and setter methods for email
    public String getEmail() {
        return email;
    }
    public void setEmail(String email) {
        this.email = email;
    }
}
// Main class
public class Main7 {

    public static void main(String[] args) {
        Person person = new Person();
        // Setting properties of the person
        person.setFirstName("Poorva");
        person.setLastName("Bhorunde");
        person.setAge(24);
        person.setEmail("poorvabhorunde@gmail.com");
        // Printing details of the person
        System.out.println("First Name is: " + person.getFirstName());
        System.out.println("Last Name is: " + person.getLastName());
        System.out.println("Age is: " + person.getAge());
        System.out.println("Email is: " + person.getEmail());
    }
}
```

Output :



```
First Name: Poorva
Last Name: Bhorunde
Age: 24
Email: poorvabhorunde@example.com
```

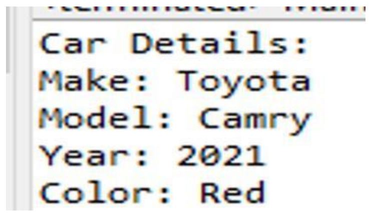
8. Create a JavaBean class Car with properties make, model, year, and color. Implement the required no-argument constructor, getter and setter methods for each property. Write a main class to create an instance of Car, set its properties, and print the car details.

Program :

```
package demo;
//Car JavaBean class with properties make, model, year, and color
public class Car1{  private String make;  private String model;
private int year;  private String color;  // No-argument
constructor
 public Car1() {
 }
 // Getter and setter methods for make
public String getMake() {      return
make;
 }
 public void setMake(String make) {
this.make = make;
 }
 // Getter and setter methods for model
public String getModel() {      return
model;
 }
 public void setModel(String model) {
this.model = model;
 }
 // Getter and setter methods for year
public int getYear() {      return
year;
 }
 public void setYear(int year) {
this.year = year;
 }
 // Getter and setter methods for color
public String getColor() {      return
color;
 }
 public void setColor(String color) {
this.color = color;
 } }
package demo;
//Main class to demonstrate the usage of Car class public
class Main7{
 public static void main(String[] args) {
     // Creating an instance of Car
     Car1 car = new Car1();

     // Setting properties
car.setMake("Toyota");
car.setModel("Camry");
car.setYear(2021);
car.setColor("Red");       //
Printing out the 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: Toyota
Model: Camry
Year: 2021
Color: Red
```