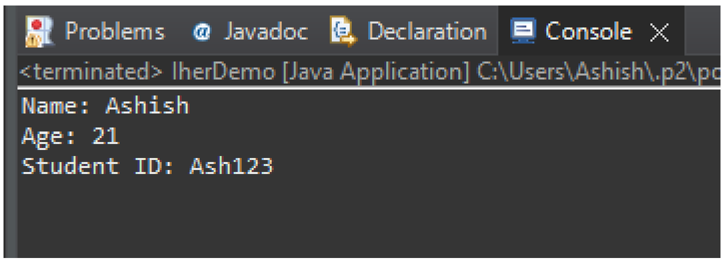
# Lab 3

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:

```java
package hellow;

class Persson {
  protected String name; // Makeing name protected for access in subclass
  protected int age;

  public Persson(String name, int age) {
   this.name = name;
   this.age = age;
  }

  public void display() {
   System.out.println("Name: " + name);
   System.out.println("Age: " + age);
  }
 }

 class Student extends Persson {
  private String studentID;

  public Student(String name, int age, String studentID) {
   super(name, age); //i'mCalling superclass constructor to initialize name and age
   this.studentID = studentID;
  }

  public void display() {
   super.display(); // i am here Calling superclass method to display name and age
   System.out.println("Student ID: " + studentID);
  }
 }

 public class IherDemo {
  public static void main(String[] args) {
   Student student = new Student("Ashish", 21, "Ash123");
   student.display();
  }
 }
```

Output :

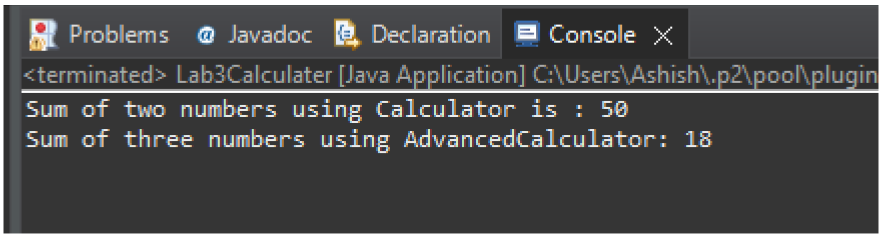


```
<terminated> IherDemo [Java Application] C:\Users\Ashish\.p2\pc
Name: Ashish
Age: 21
Student ID: Ash123
```

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 hellow;

class Calculator {
  public int add(int a, int b) {
    return a + b; // adding two number
  }
 }

 class AdvancedCalculator extends Calculator {
  public int add(int a, int b, int c) {
    //Overloading the add method to handle three integers.
    return a + b + c;
  }
 }

 public class Lab3Calculater  {
  public static void main (String[] args) {
    Calculator calculator  = new Calculator();
    AdvancedCalculator advancedCalculator = new AdvancedCalculator();

    int sumTwo = calculator.add(20, 30);
    int sumThree = advancedCalculator.add(1, 4, 13);

    System.out.println("Sum of two numbers using Calculator is : " + sumTwo);
    System.out.println("Sum of three numbers using AdvancedCalculator: " +
sumThree);
  }

 }
```

Output :

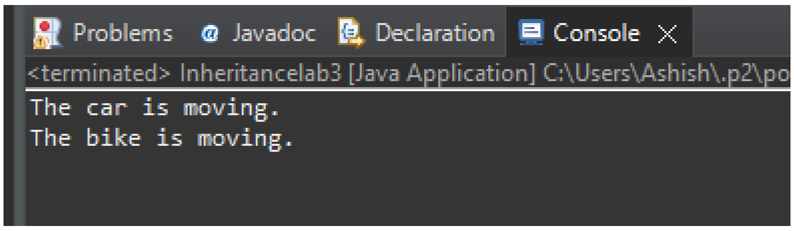


```
<terminated> Lab3Calculater [Java Application] C:\Users\Ashish\.p2\pool\plugin
Sum of two numbers using Calculator is : 50
Sum of three numbers using AdvancedCalculator: 18
```

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:

```
package hellow;

//first Creating a Superclass
class Vehicle {
 public void move() {
   System.out.println("Vehicle is moving");
 }
}

class Car extends Vehicle {  //Subclass Bike extends Vehicle
 public void move() {
   System.out.println("Car is moving");
 }
}
class Bike extends Vehicle { //Subclass Bike extends Vehicle
 public void move() {
   System.out.println("Bike is moving");
 }
}


public class Inheritancelab3 {
 public static void main(String[] args) {

   //Calling move mehod by makuing oblecvt of classes

   Vehicle car = new Car();
   Vehicle bike = new Bike();

   car.move();
   bike.move();
 }
}
```

Output :

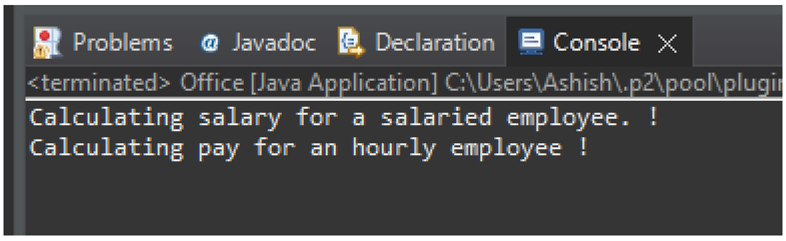


```
<terminated> Inheritancelab3 [Java Application] C:\Users\Ashish\.p2\po
The car is moving.
The bike is moving.
```

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.

Code:
```
package hellow;

//Abstract superclass Employee
abstract class Employees {
 public abstract void calculatePay(); // Abstract method far calculate and pay
}

class SalariedEmployee extends Employees {
 public void calculatePay() {
   System.out.println("Calculating salary for a salaried employee. !");
 }
}

//Subclass HourlyEmployee
class HourlyEmployee extends Employees {
 public void calculatePay() {
   System.out.println("Calculating pay for an hourly employee !");
 }
}

public class Office {
 public static void main(String[] args) {
   Employees salariedEmp = new SalariedEmployee();
   Employees hourlyEmp = new HourlyEmployee();

   salariedEmp.calculatePay(); //calling methods
   hourlyEmp.calculatePay();
 }
}
```

Output :



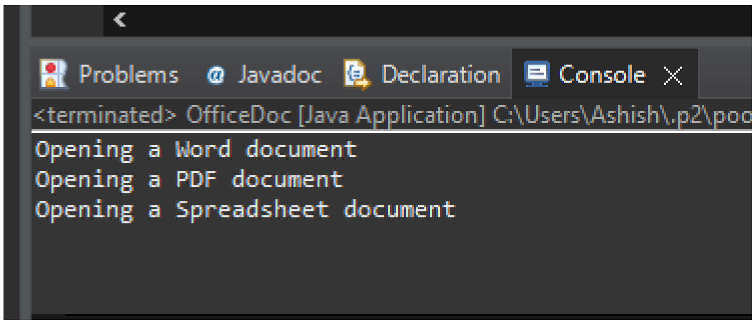
```
<terminated> Office [Java Application] C:\Users\Ashish\.p2\pool\plugi
Calculating salary for a salaried employee. !
Calculating pay for an hourly employee !
```

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 hellow;
class Document {
 // Method to open document (to be overridden by subclasses)
 public void open() {
   System.out.println("Opening a generic document");
```

```java
 }
}

//Sub claases
class WordDocument extends Document {
 public void open() {
   System.out.println("Opening a Word document");
 }
}

class PDFDocument extends Document {
 public void open() {
   System.out.println("Opening a PDF document");
 }
}

class SpreadsheetDocument extends Document {
 public void open() {
   System.out.println("Opening a Spreadsheet document");
 }
}

public class OfficeDoc {
 public static void main(String[] args) {
   Document doc1 = new WordDocument();
   Document doc2 = new PDFDocument();
   Document doc3 = new SpreadsheetDocument();

   //calling the method from classes
   doc1.open();
   doc2.open();
   doc3.open();
 }
```

Output :



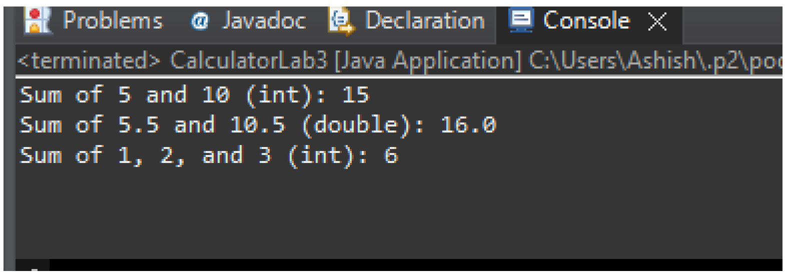
```
<terminated> OfficeDoc [Java Application] C:\Users\Ashish\.p2\poo
Opening a Word document
Opening a PDF document
Opening a Spreadsheet document
```

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 hellow;

// creating a Class with overloaded add methods
class Calculat {
  // Method to add two integers
  public int add(int a, int b) {
    return a + b;
  }

  // Method for add two doubles
  public double add(double a, double b) {
    return a + b;
  }

  // Method for add three integers
  public int add(int a, int b, int c) {
    return a + b + c;
  }
}

public class CalculatorLab3 {
  public static void main(String[] args) {
    Calculat calc = new Calculat();

    // Demonstrate adding two integers
    int sum1 = calc.add(5, 10);
    System.out.println("Sum of 5 and 10 (int): " + sum1);

    double sum2 = calc.add(5.5, 10.5);
    System.out.println("Sum of 5.5 and 10.5 (double): " + sum2);

    int sum3 = calc.add(1, 2, 3);
    System.out.println("Sum of 1, 2, and 3 (int): " + sum3);
  }
}
```

Output :

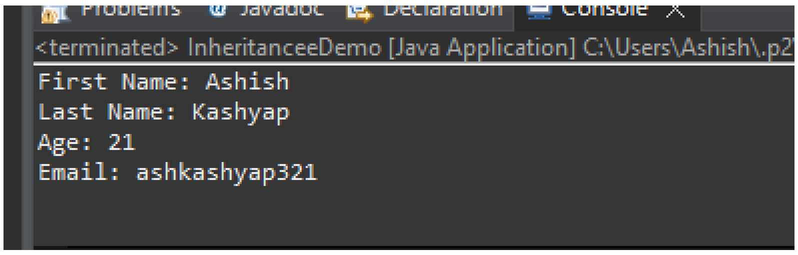


```
<terminated> CalculatorLab3 [Java Application] C:\Users\Ashish\.p2\poo
Sum of 5 and 10 (int): 15
Sum of 5.5 and 10.5 (double): 16.0
Sum of 1, 2, and 3 (int): 6
```

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:
```java
package hellow;

import java.io.Serializable;

class Person implements Serializable {
  private String firstName;
  private String lastName;
  private int age;
  private String email;

  // creatingg constructor
  public Person() {
  }

  // 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;
  }
}
```

```java
public class InheritanceeDemo {
  public static void main(String[] args) {
    // Create an instance of Person
    Person person = new Person();

    person.setFirstName("Ashish");
    person.setLastName("Kashyap");
    person.setAge(21);
    person.setEmail("ashkashyap321");


    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 :

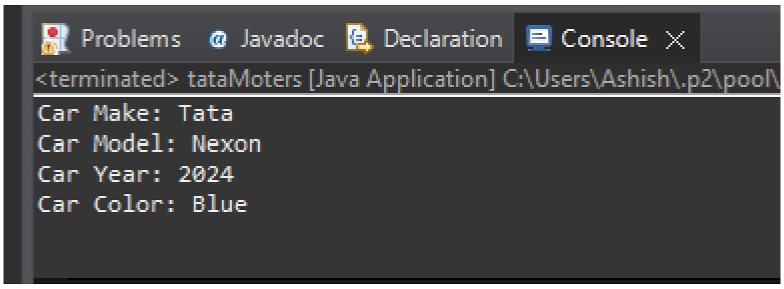


```
<terminated> InheritanceeDemo [Java Application] C:\Users\Ashish\.p2
First Name: Ashish
Last Name: Kashyap
Age: 21
Email: ashkashyap321
```

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:

```java
package hellow;
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;
  }
  // 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 :



```
<terminated> tataMoters [Java Application] C:\Users\Ashish\.p2\pool\
Car Make: Tata
Car Model: Nexon
Car Year: 2024
Car Color: Blue
```