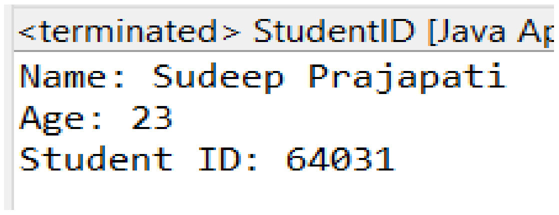
1.Create a superclass Person with attributes name and age, and a method display(). Create a subclass Student that adds an attribute studentID. Write a program to create a Student object and display all its attributes.

Code:

```
package Hellow;  //This is my package

//Superclass Person
class Person {
	// Attributes
	protected String name;
	protected int age;

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

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

//Subclass Student inheriting from Person
class Student extends Person {
	private int studentID;

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

	// Method to display student information, overriding display() from Person
	@Override
	public void display() {
		super.display(); // Call superclass display method
		System.out.println("Student ID:  " + studentID);
	}
}

//StudentID class to demonstrate usage
public class StudentID {
	public static void main(String[] args) {
		Student student = new Student("Sudeep Prajapati", 23, 64031);
		// Displaying all attributes of the Student
		student.display();
	}
}
```

Output:

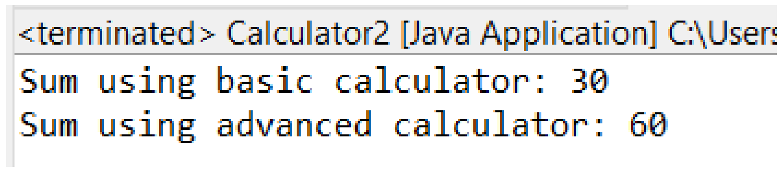


```
<terminated> StudentID [Java Ap
Name:  Sudeep Prajapati
Age:  23
Student ID:  64031
```

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:
```java
package EDemo;

//Superclass Calculator
class Calculator {
	// Method to add two integers
	public int add(int a, int b) {
		return a + b;
	}
}

//Subclass AdvancedCalculator inheriting 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;
	}
}

//Main class to demonstrate usage
public class Calculator2 {
	public static void main(String[] args) {
		Calculator basicCalc = new Calculator();
		AdvancedCalculator advancedCalc = new AdvancedCalculator();

		// Using the add methods
		int sum1 = basicCalc.add(10, 20); // Uses Calculator's add method
		int sum2 = advancedCalc.add(10, 20, 30); // Uses AdvancedCalculator's add method
		System.out.println("Sum using basic calculator: " + sum1);
		System.out.println("Sum using advanced calculator: " + sum2);
	}
}
```

Output:



```
<terminated> Calculator2 [Java Application] C:\Users
Sum using basic calculator: 30
Sum using advanced calculator: 60
```

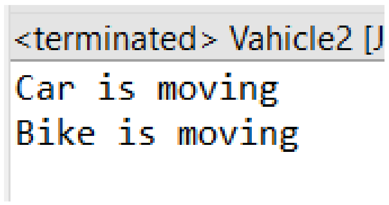
3. Create a superclass Vehicle with a method move(). Create subclasses Car and Bike that inherit from Vehicle. Write a program to create objects of Car and Bike and call the move() method on each.

Code:

```java
package 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 Vahicle2{
        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:



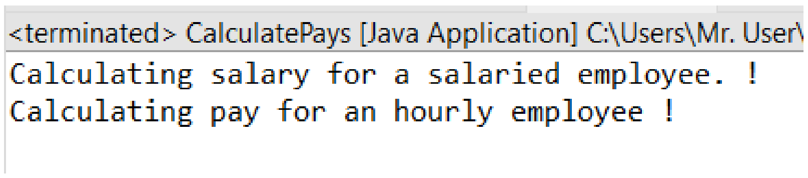
```
<terminated> Vahicle2 [J
Car is moving
Bike is moving
```

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 EDemo;
//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 CalculatePays {
        public static void main(String[] args) {
                Employees salariedEmp = new SalariedEmployee();
                Employees hourlyEmp = new HourlyEmployee();
                salariedEmp.calculatePay(); //calling methods
                hourlyEmp.calculatePay();
        }
}
```

Output:



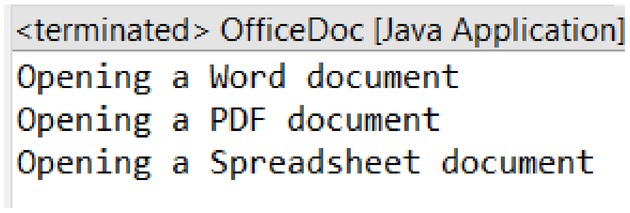
```
<terminated> CalculatePays [Java Application] C:\Users\Mr. User\
Calculating salary for a salaried employee. !
Calculating pay for an hourly employee !
```

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:

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



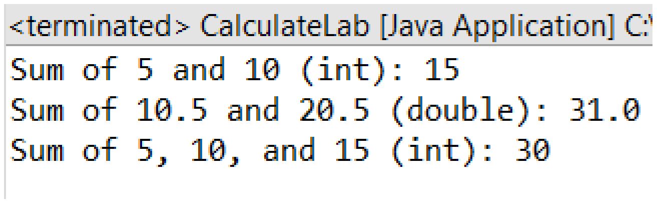
```
<terminated> OfficeDoc [Java Application]
Opening a Word document
Opening a PDF document
Opening a Spreadsheet document
```

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 CalculateLab {
	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(10.5, 20.5);
		System.out.println("Sum of 10.5 and 20.5 (double): " + sum2);
		int sum3 = calc.add(5,10,15);
		System.out.println("Sum of 5, 10, and 15 (int): " + sum3);
	}
}
```

Output:



```
<terminated> CalculateLab [Java Application] C:\
Sum of 5 and 10 (int): 15
Sum of 10.5 and 20.5 (double): 31.0
Sum of 5, 10, and 15 (int): 30
```

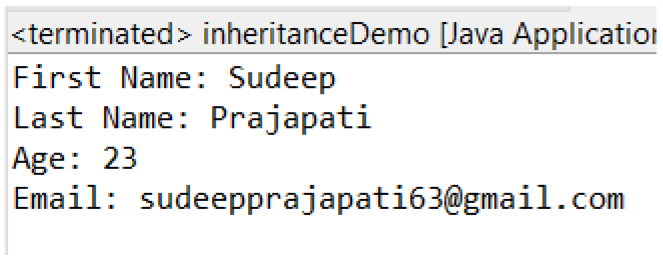
7. Create a JavaBean class Person with properties firstName, lastName, age, and email. Implement the required no-argument constructor, getter and setter methods for each property. Write a main class to create an instance of Person, set its properties, and print them out.

Code:

```
package EDemo;

import java.io.Serializable;
class Person implements Serializable {
	private String firstName;
	private String lastName;
```

```java
	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;
	}
}
public class inheritanceDemo {
	public static void main(String[] args) {
		// Create an instance of Person
		Person person = new Person();
		person.setFirstName("Sudeep");
		person.setLastName("Prajapati");
		person.setAge(23);
		person.setEmail("sudeepprajapati63@gmail.com");
		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:



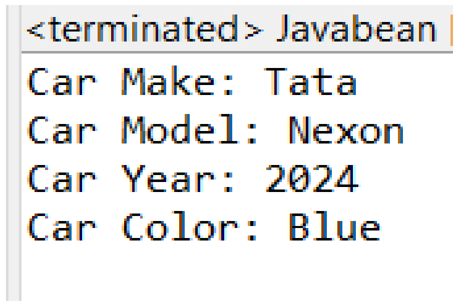
```
<terminated> inheritanceDemo [Java Applicatio
First Name: Sudeep
Last Name: Prajapati
Age: 23
Email: sudeepprajapati63@gmail.com
```

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.

Code:

```java
package EDemo;

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 Javabean { // 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> Javabean
Car Make: Tata
Car Model: Nexon
Car Year: 2024
Car Color: Blue
```