Kartik Verma

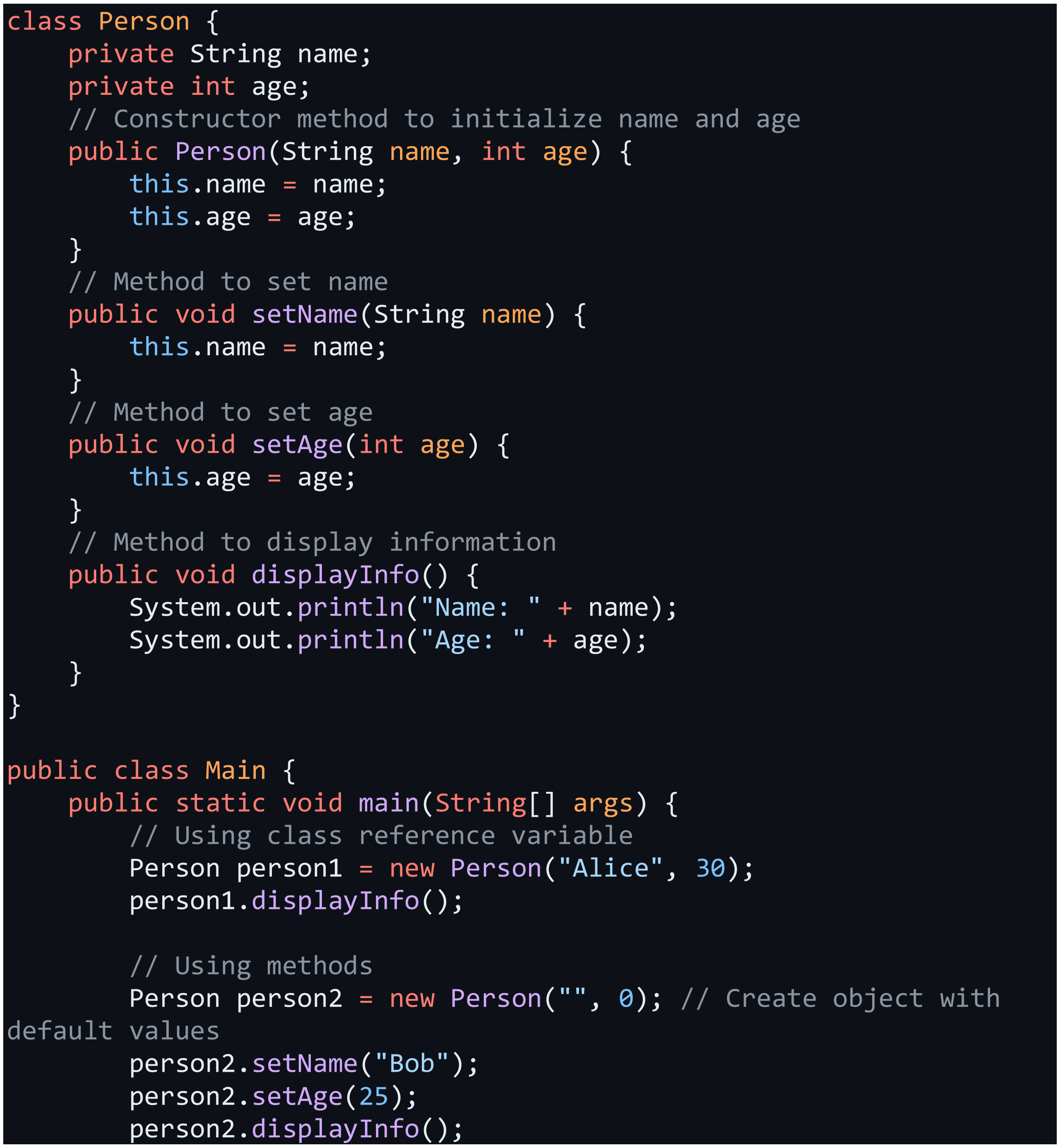
CSE-3

Java Lab File

Lab 3:-

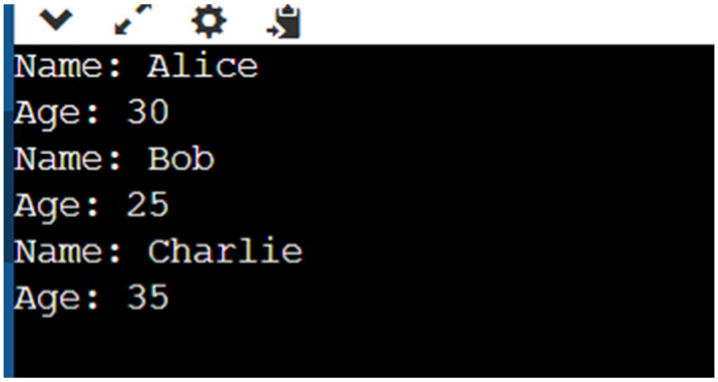
Program 7: Write and implement java program to create class and object initialization by the 3 methods i.e. using class reference variable, using methods, using constructors.

Code:-



```
class Person {
    private String name;
    private int age;
    // Constructor method to initialize name and age
    public Person(String name, int age) {
        this.name = name;
        this.age = age;
    }
    // Method to set name
    public void setName(String name) {
        this.name = name;
    }
    // Method to set age
    public void setAge(int age) {
        this.age = age;
    }
    // Method to display information
    public void displayInfo() {
        System.out.println("Name: " + name);
        System.out.println("Age: " + age);
    }
}

public class Main {
    public static void main(String[] args) {
        // Using class reference variable
        Person person1 = new Person("Alice", 30);
        person1.displayInfo();

        // Using methods
        Person person2 = new Person("", 0); // Create object with
default values
        person2.setName("Bob");
        person2.setAge(25);
        person2.displayInfo();
        // Using constructor
        Person person3 = new Person("Charlie", 35);
        person3.displayInfo();
    }
}
```

Output



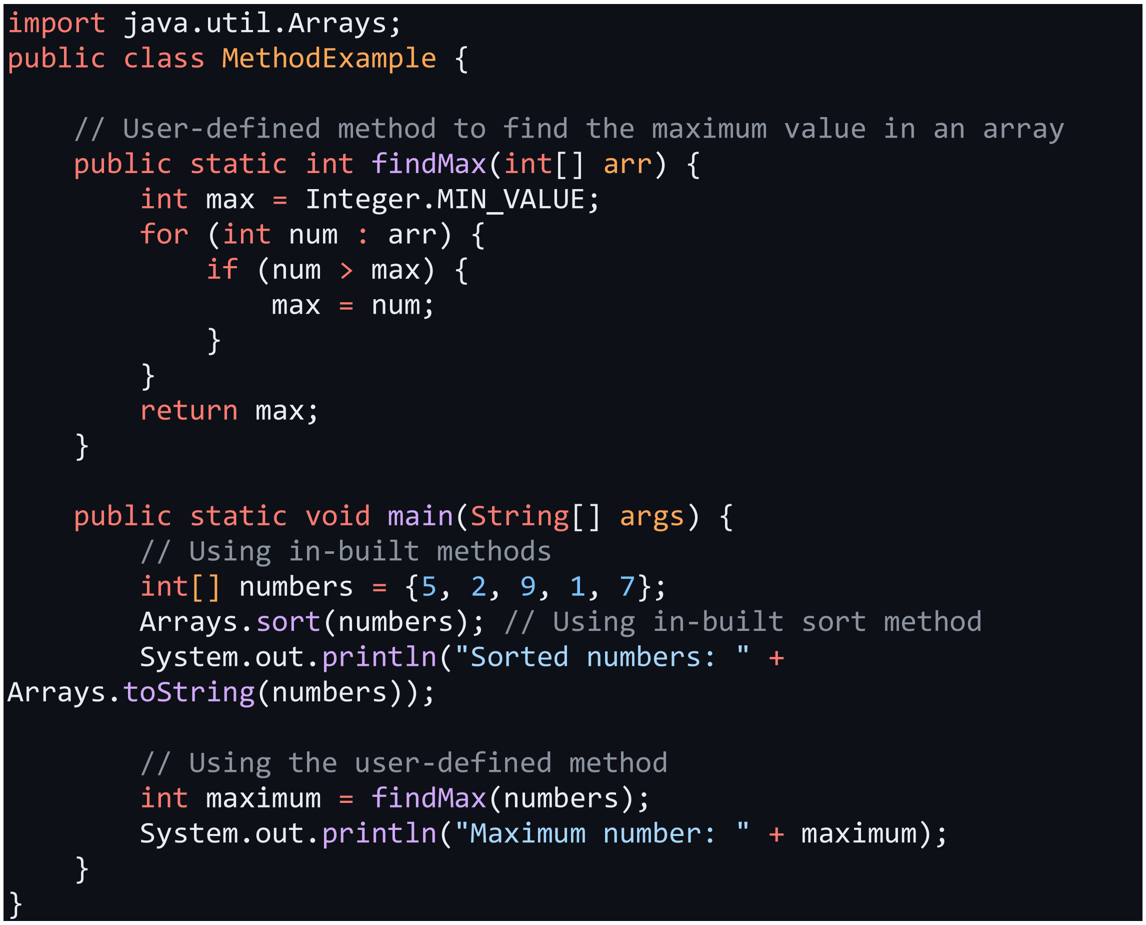
```
Name: Alice
Age: 30
Name: Bob
Age: 25
Name: Charlie
Age: 35
```

Java Lab File

Lab 3:-

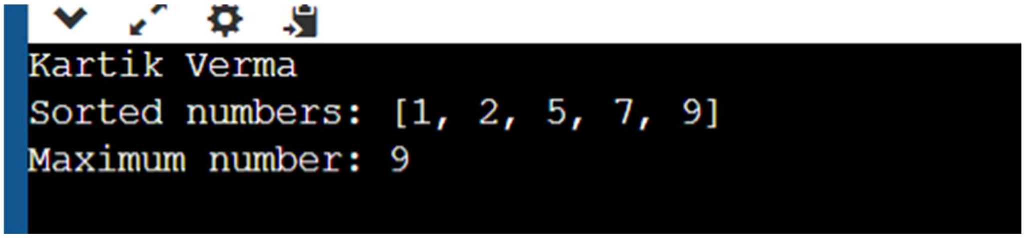
Program 8: Write and implement java program to in-built methods and user-defined methods.

Code:-



```
import java.util.Arrays;
public class MethodExample {

    // User-defined method to find the maximum value in an array
    public static int findMax(int[] arr) {
        int max = Integer.MIN_VALUE;
        for (int num : arr) {
            if (num > max) {
                max = num;
            }
        }
        return max;
    }

    public static void main(String[] args) {
        // Using in-built methods
        int[] numbers = {5, 2, 9, 1, 7};
        Arrays.sort(numbers); // Using in-built sort method
        System.out.println("Sorted numbers: " +
Arrays.toString(numbers));

        // Using the user-defined method
        int maximum = findMax(numbers);
        System.out.println("Maximum number: " + maximum);
    }
}
```

Output



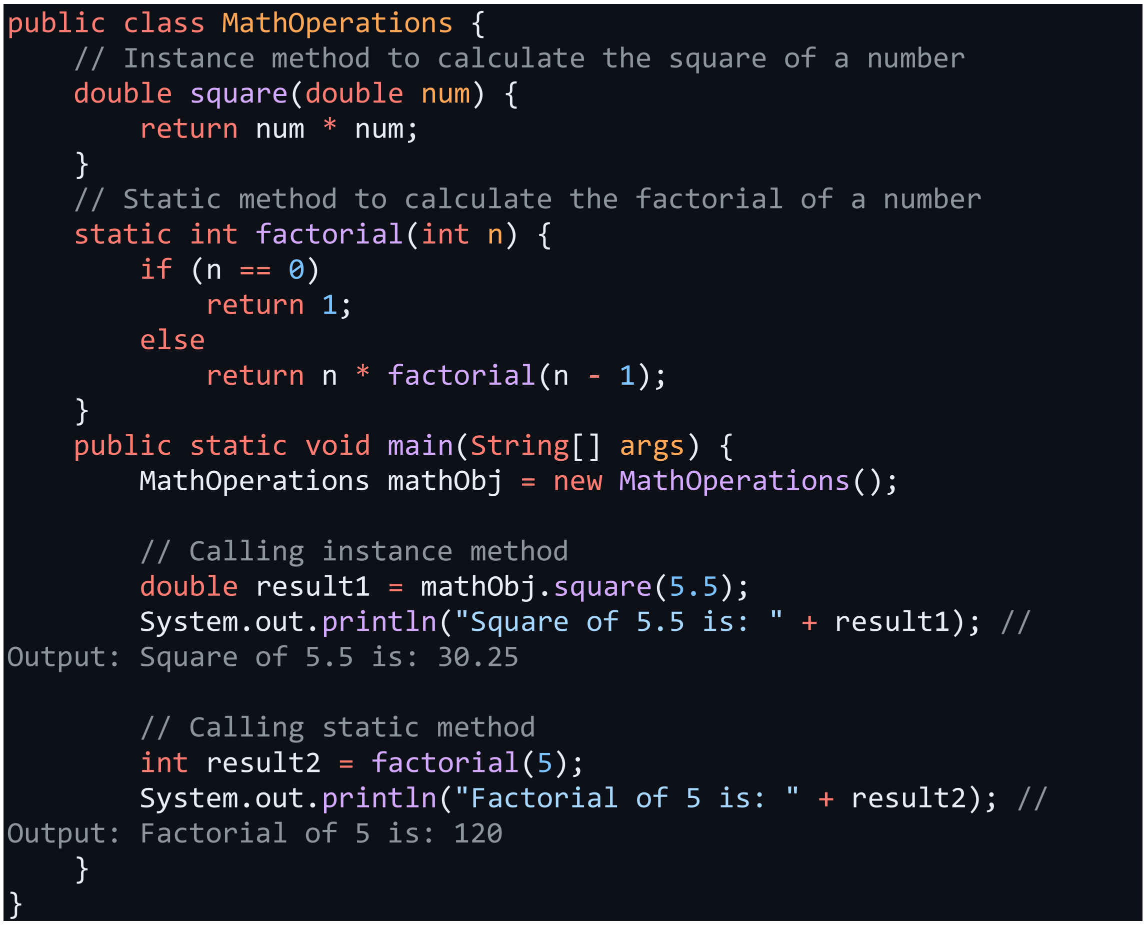
```
Kartik Verma
Sorted numbers: [1, 2, 5, 7, 9]
Maximum number: 9
```

Java Lab File

Lab 4:-

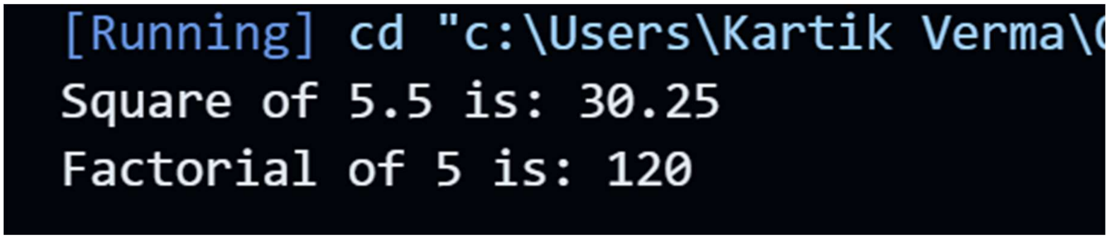
Program 9: Write and implement java program to instance methods and static methods.

Code:-



```
public class MathOperations {
    // Instance method to calculate the square of a number
    double square(double num) {
        return num * num;
    }
    // Static method to calculate the factorial of a number
    static int factorial(int n) {
        if (n == 0)
            return 1;
        else
            return n * factorial(n - 1);
    }
    public static void main(String[] args) {
        MathOperations mathObj = new MathOperations();

        // Calling instance method
        double result1 = mathObj.square(5.5);
        System.out.println("Square of 5.5 is: " + result1); //
Output: Square of 5.5 is: 30.25

        // Calling static method
        int result2 = factorial(5);
        System.out.println("Factorial of 5 is: " + result2); //
Output: Factorial of 5 is: 120
    }
}
```

Output



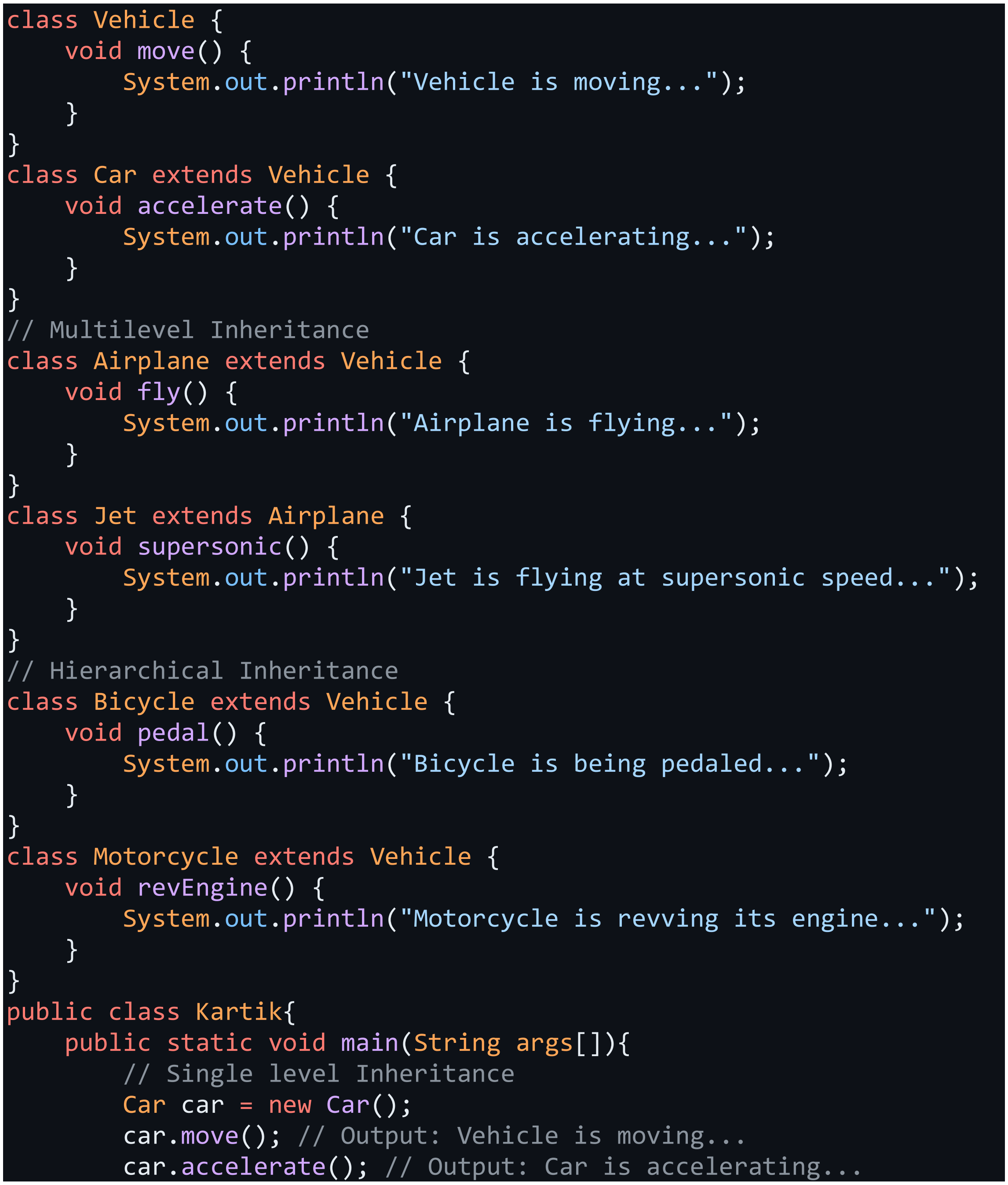
```
[Running] cd "c:\Users\Kartik Verma\
Square of 5.5 is: 30.25
Factorial of 5 is: 120
```

Java Lab File

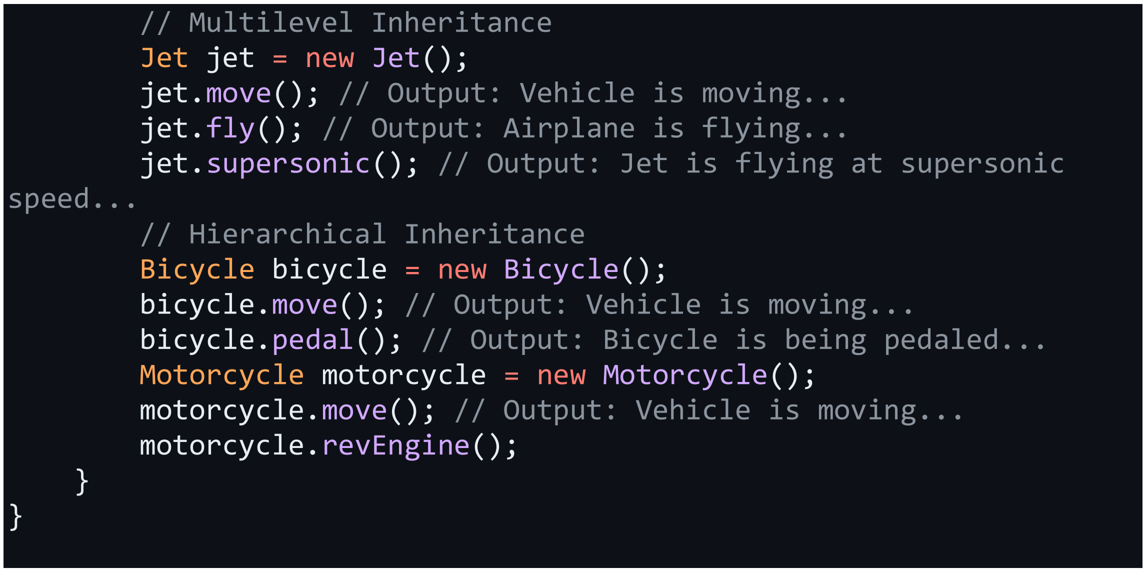
Lab 4:-

Program 10: Write a java program to implement Inhertiance: Single, Multilevel, and Hierarchical.

Code:-

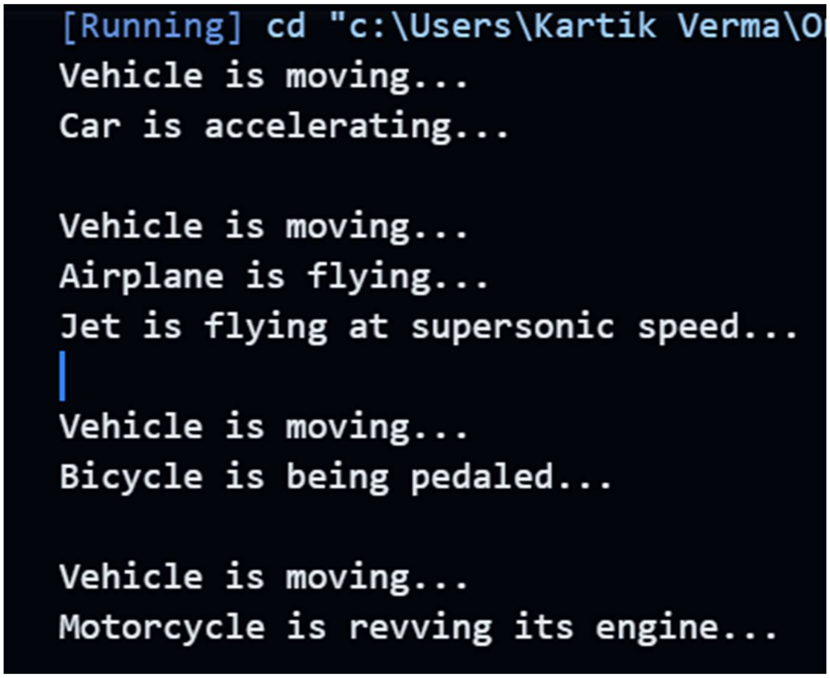


```java
class Vehicle {
    void move() {
        System.out.println("Vehicle is moving...");
    }
}
class Car extends Vehicle {
    void accelerate() {
        System.out.println("Car is accelerating...");
    }
}
// Multilevel Inheritance
class Airplane extends Vehicle {
    void fly() {
        System.out.println("Airplane is flying...");
    }
}
class Jet extends Airplane {
    void supersonic() {
        System.out.println("Jet is flying at supersonic speed...");
    }
}
// Hierarchical Inheritance
class Bicycle extends Vehicle {
    void pedal() {
        System.out.println("Bicycle is being pedaled...");
    }
}
class Motorcycle extends Vehicle {
    void revEngine() {
        System.out.println("Motorcycle is revving its engine...");
    }
}
public class Kartik{
    public static void main(String args[]){
        // Single level Inheritance
        Car car = new Car();
        car.move(); // Output: Vehicle is moving...
        car.accelerate(); // Output: Car is accelerating...
```



```
        // Multilevel Inheritance
        Jet jet = new Jet();
        jet.move(); // Output: Vehicle is moving...
        jet.fly(); // Output: Airplane is flying...
        jet.supersonic(); // Output: Jet is flying at supersonic
speed...
        // Hierarchical Inheritance
        Bicycle bicycle = new Bicycle();
        bicycle.move(); // Output: Vehicle is moving...
        bicycle.pedal(); // Output: Bicycle is being pedaled...
        Motorcycle motorcycle = new Motorcycle();
        motorcycle.move(); // Output: Vehicle is moving...
        motorcycle.revEngine();
    }
}
```

Output



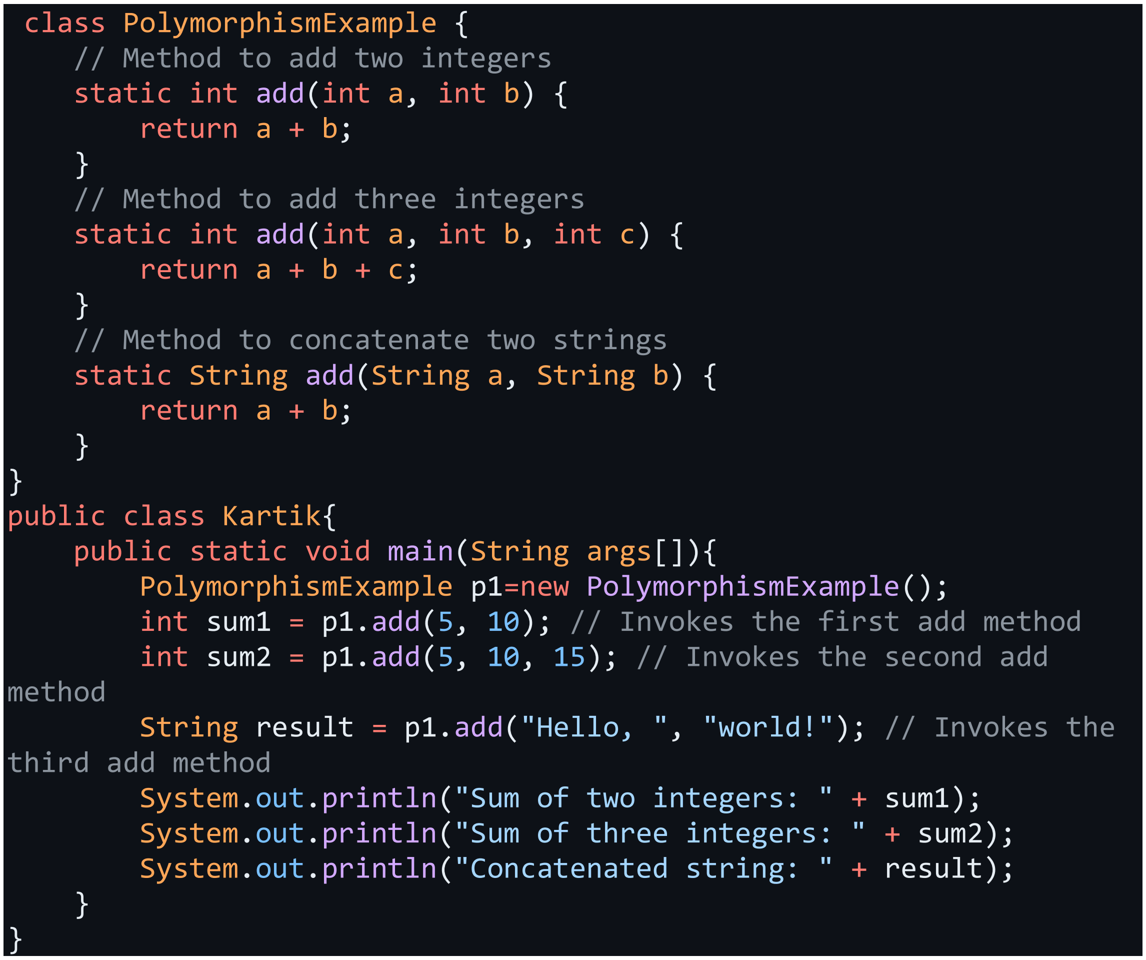
```
[Running] cd "c:\Users\Kartik Verma\O
Vehicle is moving...
Car is accelerating...

Vehicle is moving...
Airplane is flying...
Jet is flying at supersonic speed...

Vehicle is moving...
Bicycle is being pedaled...

Vehicle is moving...
Motorcycle is revving its engine...
```

Java Lab File

Lab 4:-

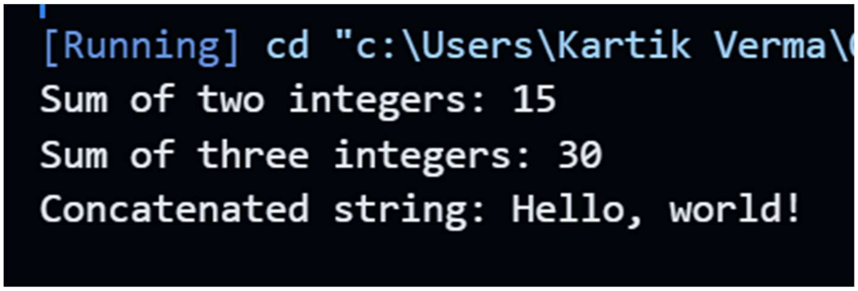
Program 11: Write a java program to implement compile time polymorphism using method overloading.

Code:-



```
 class PolymorphismExample {
    // Method to add two integers
    static int add(int a, int b) {
        return a + b;
    }
    // Method to add three integers
    static int add(int a, int b, int c) {
        return a + b + c;
    }
    // Method to concatenate two strings
    static String add(String a, String b) {
        return a + b;
    }
}
public class Kartik{
    public static void main(String args[]){
        PolymorphismExample p1=new PolymorphismExample();
        int sum1 = p1.add(5, 10); // Invokes the first add method
        int sum2 = p1.add(5, 10, 15); // Invokes the second add
method
        String result = p1.add("Hello, ", "world!"); // Invokes the
third add method
        System.out.println("Sum of two integers: " + sum1);
        System.out.println("Sum of three integers: " + sum2);
        System.out.println("Concatenated string: " + result);
    }
}
```

Output



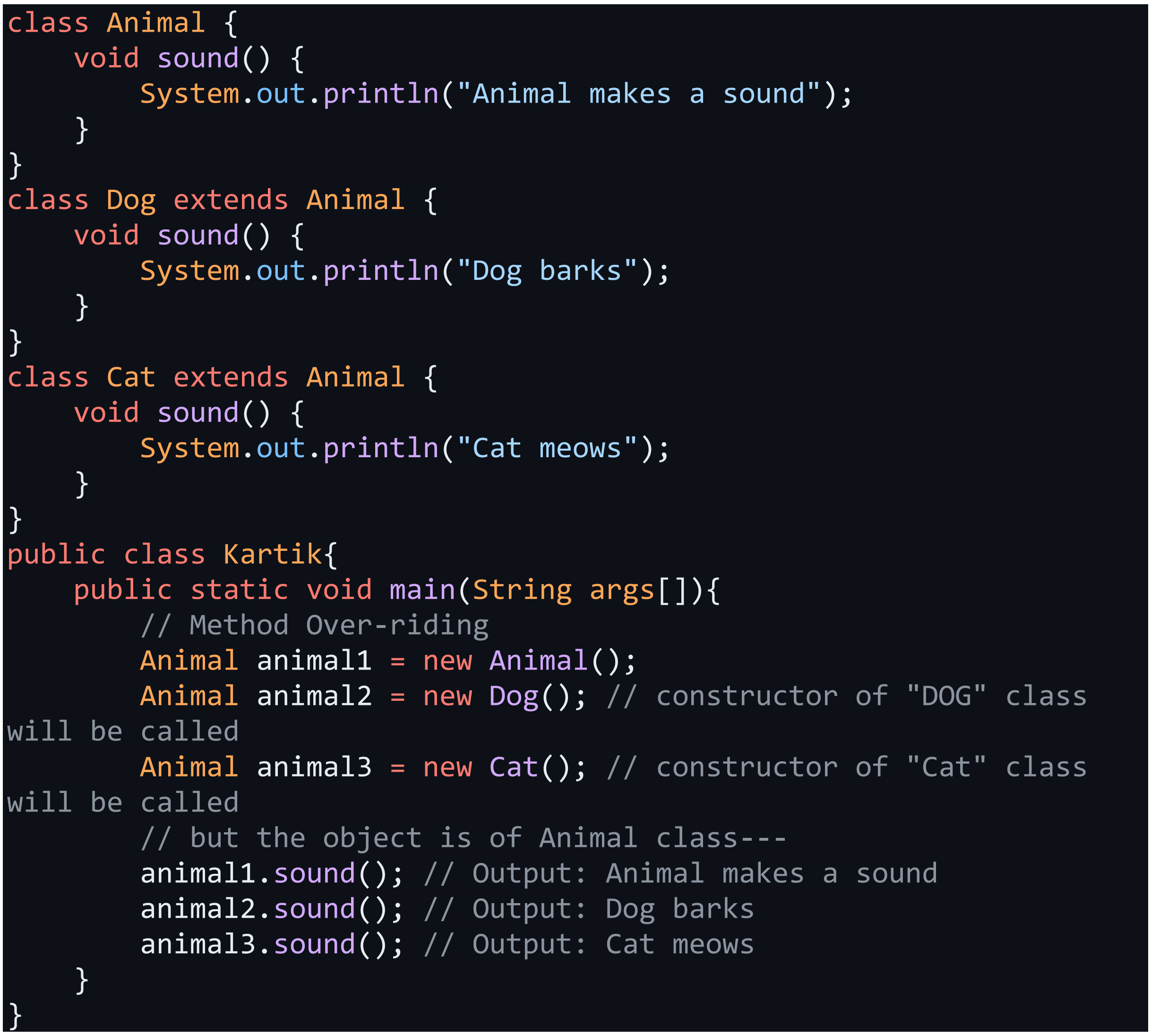
```
[Running] cd "c:\Users\Kartik Verma\
Sum of two integers: 15
Sum of three integers: 30
Concatenated string: Hello, world!
```

Java Lab File

Lab 4:-

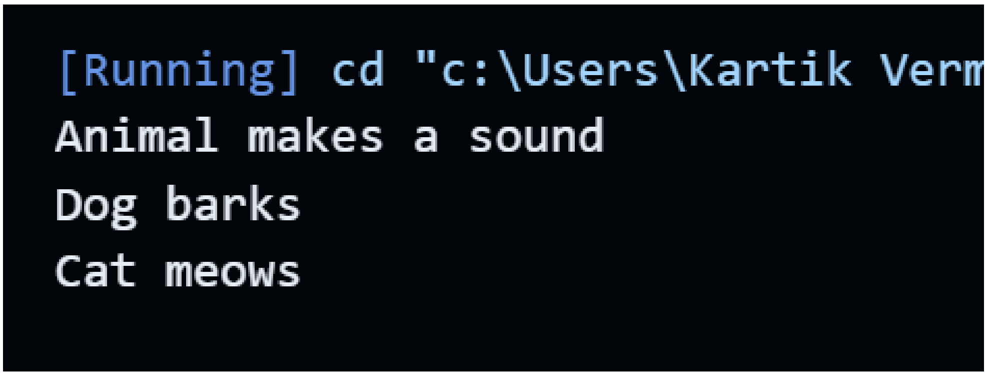
Program 12: Write a java program to implement run time polymorphism using method overriding.

Code:-



```
class Animal {
    void sound() {
        System.out.println("Animal makes a sound");
    }
}
class Dog extends Animal {
    void sound() {
        System.out.println("Dog barks");
    }
}
class Cat extends Animal {
    void sound() {
        System.out.println("Cat meows");
    }
}
public class Kartik{
    public static void main(String args[]){
        // Method Over-riding
        Animal animal1 = new Animal();
        Animal animal2 = new Dog(); // constructor of "DOG" class
will be called
        Animal animal3 = new Cat(); // constructor of "Cat" class
will be called
        // but the object is of Animal class---
        animal1.sound(); // Output: Animal makes a sound
        animal2.sound(); // Output: Dog barks
        animal3.sound(); // Output: Cat meows
    }
}
```

Output



```
[Running] cd "c:\Users\Kartik Verm
Animal makes a sound
Dog barks
Cat meows
```