Lab 5

Experiment 5

Q1. Create a class with a static String field that is initialized at the point of definition, and another one that is initialized by the static block. Add a static method that prints both fields and demonstrates that they are both initialized before they are used.

Create a main() that uses varargs instead of the ordinary main() syntax. Print all the elements in the resulting args array. Test it with various numbers of command-line arguments.

public class Stri {

    static String string\_arg1="Initialized at the time of declaration";

    static String string\_arg2;

    static{

        string\_arg2="Initialized in the static block";

    }

    static void print()

    {

        System.out.println("String srguement 1 : "+string\_arg1);

        System.out.println("String srguement 2 : "+string\_arg2);

    }

    public static void main(String ...args) {

        Stri.print();

        System.out.println("Command line arguements ");

        for(String arg:args)

        {

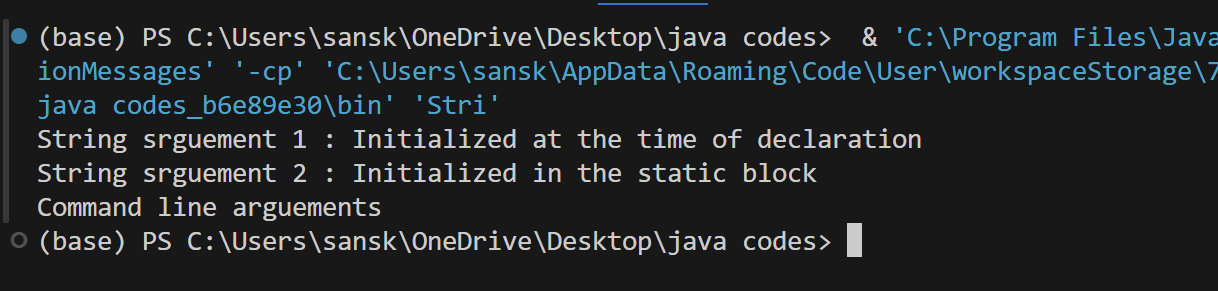
            System.out.println(arg);

        }

    }

}

**Output :**

****

Experiment 6

Q1. Design and implement a class named MyDate. The class contains:

The data fields year, month, and day that represent a date. (Note: month is 0-based, i.e., 0 is for January.)

A no-arg constructor that creates a MyDate object for the current date.

A constructor that constructs a MyDate object with a specified elapsed time since midnight, January 1, 1970, in milliseconds.

import java.util.Calendar;

import java.util.GregorianCalendar;

public class MyDate {

    private int year;

    private int month;

    private int day;

    public MyDate() {

        Calendar calendar = new GregorianCalendar();

        this.year = calendar.get(Calendar.YEAR);

        this.month = calendar.get(Calendar.MONTH);

        this.day = calendar.get(Calendar.DAY\_OF\_MONTH);

    }

    public MyDate(long elapsedTime) {

        Calendar calendar = new GregorianCalendar();

        calendar.setTimeInMillis(System.currentTimeMillis() + elapsedTime);

        this.year = calendar.get(Calendar.YEAR);

        this.month = calendar.get(Calendar.MONTH);

        this.day = calendar.get(Calendar.DAY\_OF\_MONTH);

    }

    public int getYear() {

        return year;

    }

    public int getMonth() {

        return month + 1;

    }

    public int getDay() {

        return day;

    }

    public static void main(String[] args) {

        MyDate currentDate = new MyDate();

        long elapsedTime = 1000L \* 24 \* 60 \* 60 \* 1000;

        MyDate futureDate = new MyDate(elapsedTime);

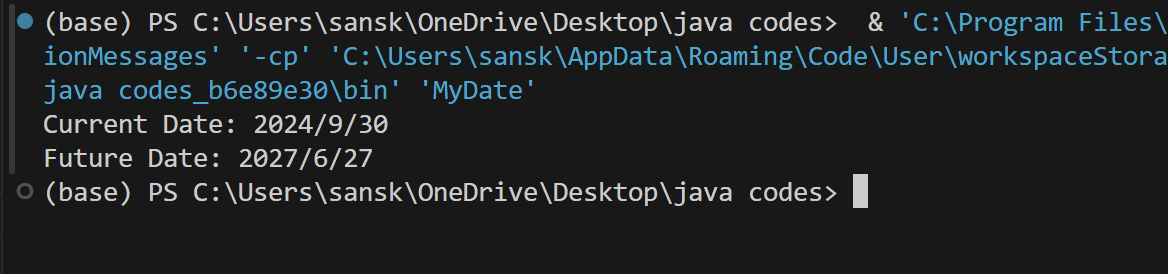
        System.out.println("Current Date: " + currentDate.getYear() + "/" + currentDate.getMonth() + "/" + currentDate.getDay());

        System.out.println("Future Date: " + futureDate.getYear() + "/" + futureDate.getMonth() + "/" + futureDate.getDay());

    }

}

**Output :**

****