CSE-2008 Assignment-9

Academic year: 2020-2021 **Semester:** WIN

Faculty Name: Mr. Sanket Mishra sir **Date:** 18 /11/2021

Student name: M.Taran **Reg. no.:** 19BCE7346

QUESTION:

Write a Java program using multithreading that helps to identify the summary statistics for a range of numbers. The program should accept a list of integers from the console and then there should be four threads present other than the main thread. The first thread should be named HEARTS , the second thread name is SPADES, third thread name is CLUBS with priority as main and fourth one is DIAMONDS . All the user defined threads have an uniform priority of 7 while main has priority of 8. Hearts will calculate the max of the list of numbers, Spades will calculate the standard deviation of the list of numbers, Clubs will compute the skewness of the list of numbers and Diamonds will calculate the median of the list of numbers. You need to make sure the results of the threads are stored in a global variable which is printed on the main thread. The main thread prints the results of child threads after their termination. You need to use wait() so the threads print in the given order. The program needs to use a synchronized method for utilizing multithreading. After completion of computation use notifyAll() to inform all threads about the completion. The order of execution should be Hearts, Spades, Clubs and then Diamonds.

CODE :

```
import java.util.*;
public class Main {

     public static void main(String []args){
        int n;
```

```
        int[] data;
        final int[] res = new int[2];
    final double[] resf = new double[2];
        Scanner s = new Scanner(System.in);

        System.out.print("Enter the number of elements :");

        n = s.nextInt();
        data = new int[n];

    System.out.print("Enter the elements :");
        for (int i = 0; i < n; i++)
            data[i] = s.nextInt();
        Thread HEARTS = new Thread(new Runnable() {
            @Override
            public void run() {
                 int max = data[0];
                for (int i = 1; i < n ; i++)
                    if (max < data[i])
                        max = data[i];
                res[0] = max;
            }

        });
        Thread SPADES = new Thread(new Runnable() {
            @Override
            public void run() {
                double standardDeviation = 0.0;
                double mean = 0.0;
                double res = 0.0;
                double sq = 0.0;

                for (int i = 0; i < n; i++) {
                    mean = mean + data[i];
                }
                mean = mean / (n);
                for (int i = 0; i < n; i++) {
                    standardDeviation= standardDeviation + Math.pow((data[i] -
mean), 2);
                }
                sq = standardDeviation / n;
                res = Math.sqrt(sq);

                resf[0] = res;
            }
        });
```

```
        Thread CLUBS = new Thread(new Runnable() {
            @Override
            public void run() {

                double skew;

                //for mean calculation

                double mean =0.0;
                for (int i = 0; i < n; i++) {
                    mean = mean + data[i];
                }
                mean = mean / (n);

               //for median calculation

               Arrays.sort(data);
               double median;
               if (n % 2 == 0) {
                   int sum = data[n / 2] + data[n / 2 - 1];
                   median = ((double) sum) / 2;

               }else {
                   median = (double) data[n / 2];
                   }


                skew = 3*(mean-median);
                skew =skew/resf[0];

                resf[1]=skew;


            }
        });

        Thread DIAMONDS = new Thread(new Runnable() {
            @Override
            public void run() {
                 Arrays.sort(data);
               int median;
               if (n % 2 == 0) {
                   int sum = data[n / 2] + data[n / 2 - 1];
                   median = ((int) sum) / 2;

               }else {
```

```
                    median = (int) data[n / 2];
                    }

                res[1]=median;
            }
        });

        HEARTS.start();
        SPADES.start();
        CLUBS.start();
        DIAMONDS.start();
    try {
        HEARTS.join();
        SPADES.join();
        CLUBS.join();
        DIAMONDS.join();
    }
    catch (Exception e) {
    }
        System.out.println("Max from Hearts: "+res[0]);
        System.out.println("Std Dev from Spades : "+resf[0]);
        System.out.println("Skew from clubs : "+resf[1]);
        System.out.println("Median from Diamonds :  "+res[1]);
     }
}
```

OUTPUT:

Result

compiled and executed in 46.439 sec(s)

```
Enter the number of elements :7
Enter the elements :90
81
78
95
79
72
85
Max from Hearts: 95
Std Dev from Spades : 7.199773239059516
Skew from clubs : 0.7738338953792331
Median from Diamonds :  81

Main ended..
```

Result

compiled and executed in 30.903 sec(s)

```
Enter the number of elements :5
Enter the elements :12
23
45
56
67
Max from Hearts: 67
Std Dev from Spades : 18.06211504780102
Skew from clubs : -0.4318431135754292
Median from Diamonds :  45

Main ended..
```