

1.

Valiveti manikanta bhuvanesh

Write a multithreaded program to compute sum of elements of NxN matrix. It should be done in two phases.

19BCD7088

Phase I:

Create N threads to compute 'N' Columns sum, where 'i' th thread computes a 'i' th column sum.

Phase II:

Create a thread to compute sum of 'N' Column's Sums. Finally main thread is going to print result.

Note: Main thread should wait till all threads completes their work.

You can use Math.random() to initialize NxN matrix, in the range 0 to 100; 5<=N<=8.

Implement this application in two versions, using both Thread class & Runnable interface.

```
import java.lang.*;
```

```
import java.util.*;
```

```
public class Sum extends Thread{
```

```
    public static int n=5;
```

```
    public static int sum;
```

```
    public static int[][]a=new int[n][n];
```

```
    public static int[]s=new int[n];
```

```
    static class ColSum extends Thread{
```

```
        int i;
```

```
        int sum=0;
```

```
        ColSum(int i){
```

```
            this.i=i;
```

```
        }
```

```
        public void run(){
```

```
            for(int q=0;q<n;q++){
```

```
                sum=sum+a[q][i];
```

```
            }
```

```
            s[i]=sum;
```

```

    }
}

static class Tsum extends Thread{
    public void run(){
        int sum1=0;
        for(int m=0;m<n;m++){
            sum1=sum1+s[m];
        }
        sum=sum1;
    }
}

public static void main(String[] args) {
    Random r= new Random();
    for(int i=0;i<n;i++) {
        for (int j=0;j<n;j++) {
            a[i][j]=r.nextInt(100);
            System.out.print(a[i][j] + " ");
        }
        System.out.println();
    }

    Sum.ColSum [] c=new Sum.ColSum[n];
    for(int k=0;k<n;k++){
        c[k]=new Sum.ColSum(k);
    }

    Sum.Tsum t2=new Sum.Tsum();

    for(int y=0;y<n;y++){
        c[y].start();
    }

    t2.start();

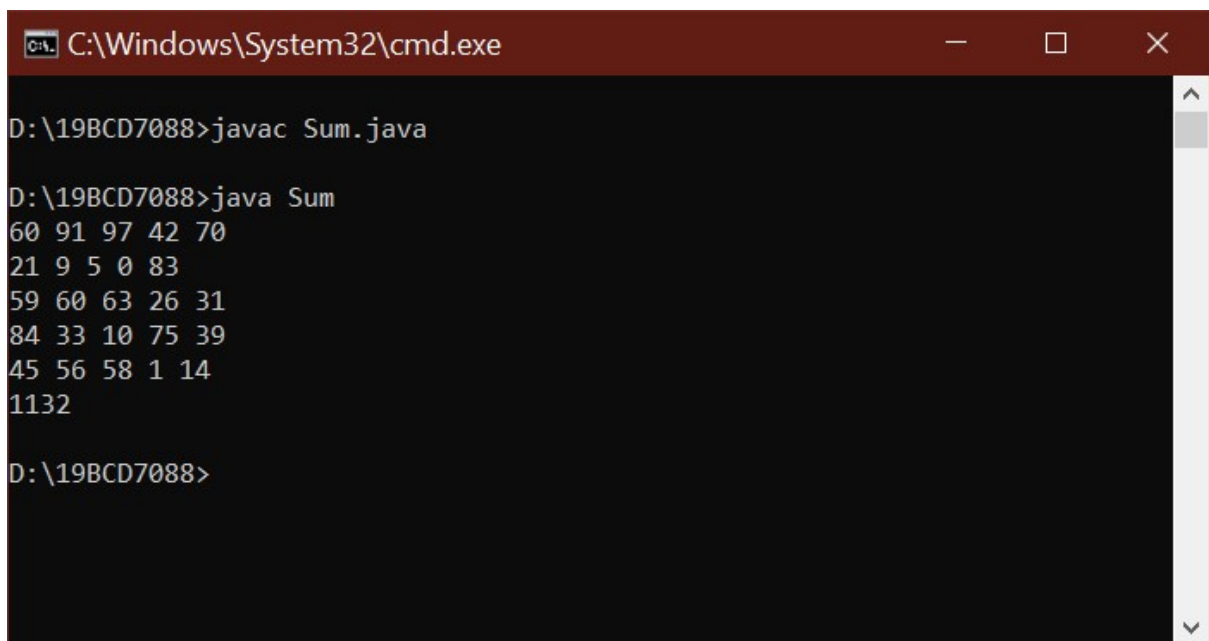
    Thread t = Thread.currentThread();

```

```

        try{
            t.sleep(1500);
        }
        catch(Exception e){
            System.out.println(e);
        }
        System.out.println(sum);
    }
}

```



The screenshot shows a Windows Command Prompt window with the title bar "C:\Windows\System32\cmd.exe". The command prompt is at the directory "D:\19BCD7088". The user has entered the command "javac Sum.java" to compile the program. Then, they entered "java Sum" to run the program. The output of the program is displayed as follows:

```

D:\19BCD7088>javac Sum.java

D:\19BCD7088>java Sum
60 91 97 42 70
21 9 5 0 83
59 60 63 26 31
84 33 10 75 39
45 56 58 1 14
1132

D:\19BCD7088>

```

```

import java.lang.*;
import java.util.*;

public abstract class Sum1 implements Runnable{
    public static int n=5;
    public static int sum;
    public static int[][]a=new int[n][n];
    public static int[]s=new int[n];

    static class ColSum1 implements Runnable{
        int i;
        int sum=0;
    }
}

```

```

ColSum1(int i){
    this.i=i;
}
public void run(){
    for(int q=0;q<n;q++){
        sum=sum+a[q][i];
    }
    s[i]=sum;
}
}
static class Tsum implements Runnable{
    public void run(){
        int sum1=0;
        for(int m=0;m<n;m++){
            sum1=sum1+s[m];
        }
        sum=sum1;
    }
}

public static void main(String[] args) {
    Random r= new Random();
    for(int i=0;i<n;i++) {
        for (int j=0;j<n;j++) {
            a[i][j]=r.nextInt(100);
            System.out.print(a[i][j] + " ");
        }
        System.out.println();
    }
    Thread [] t1=new Thread[n];
    for(int k=0;k<n;k++){
        t1[k]=new Thread(new Sum1.ColSum1(k));
    }
}

```

```

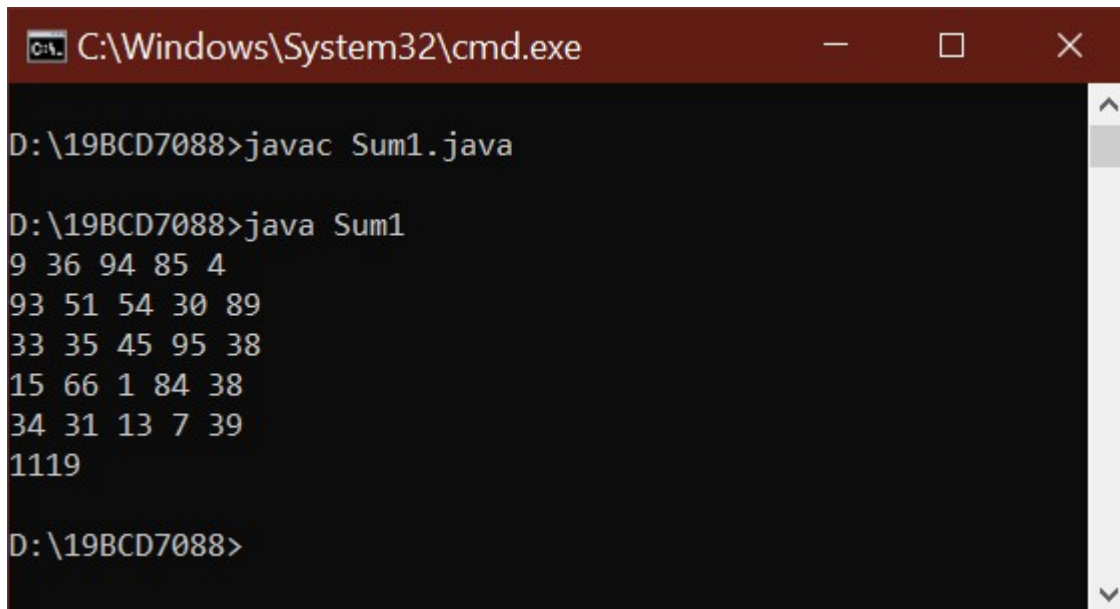
    }

    Thread t2=new Thread (new Sum1.Tsum());

    for(int y=0;y<n;y++){
        t1[y].start();
    }
    t2.start();
    Thread t = Thread.currentThread();

        try{
            t.sleep(1500);
        }
        catch(Exception e){
            System.out.println(e);
        }
    System.out.println(sum);
}
}

```



```

C:\Windows\System32\cmd.exe
D:\19BCD7088>javac Sum1.java
D:\19BCD7088>java Sum1
9 36 94 85 4
93 51 54 30 89
33 35 45 95 38
15 66 1 84 38
34 31 13 7 39
1119
D:\19BCD7088>

```

2.

a.

Create a CeaserCipher class to perform substitution and reverse substitution of characters of a message.

- mEncryption method - substitute a character with another character of alphabet.
- mDecryption method - similar to mEncryption method but it performs in reverse.

Each character of message is considered as numeric value with the following mapping: a-z to 0-25, respectively.

(a-0, b-1, c-2, ..., z-25)

The mEncryption method replaces each character of the message with another character by using the following formula: $(N(ch) + k) \% 26$, where $N(ch)$ means Numeric value of a character 'ch', k means key value $0 \leq k \leq 25$.

The mDecryption method substitutes each character with the following formula: $(N(ch) - k) \% 26$.

Inputs to each method is a message and a key and output is substituted message printed on console character by character.

(Ex: Input to mEncryption is: rama and 25 and output is: qzlj ;

Input to mDecryption is: qzlj and 25 and output is: rama)

Create a TestCeaserCipher class to test mEncryption & mDecryption methods.

```
import java.util.Scanner;
```

```
class TestCeaserCipher
```

```
{
```

```
    public static String a = "abcdefghijklmnopqrstuvwxyz";
```

```
    public static String mEncryption(String p, int key)
```

```
    {
```

```
        p = p.toLowerCase();
```

```
        String c = "";
```

```
        for (int i = 0; i < p.length(); i++)
```

```

{
    int cp = a.indexOf(p.charAt(i));
    int kv = (key + cp) % 26;
    char rv = a.charAt(kv);
    c += rv;
}
return c;
}

public static String mDecryption(String c, int key)
{
    c = c.toLowerCase();
    String p = "";
    for (int i = 0; i < c.length(); i++)
    {
        int cp = a.indexOf(c.charAt(i));
        int kv = (cp - key) % 26;
        if (kv < 0)
        {
            kv = a.length() + kv;
        }
        char rv = a.charAt(kv);
        p += rv;
    }
    return p;
}

public static void main(String[] args)
{
    Scanner sc=new Scanner(System.in);
    System.out.println("Enter the Message to encrypt");
    String m =sc.nextLine();
    System.out.println("Enter the Key for encryption");

```

```

        int key = sc.nextInt();

        String c=mEncryption(m, key);

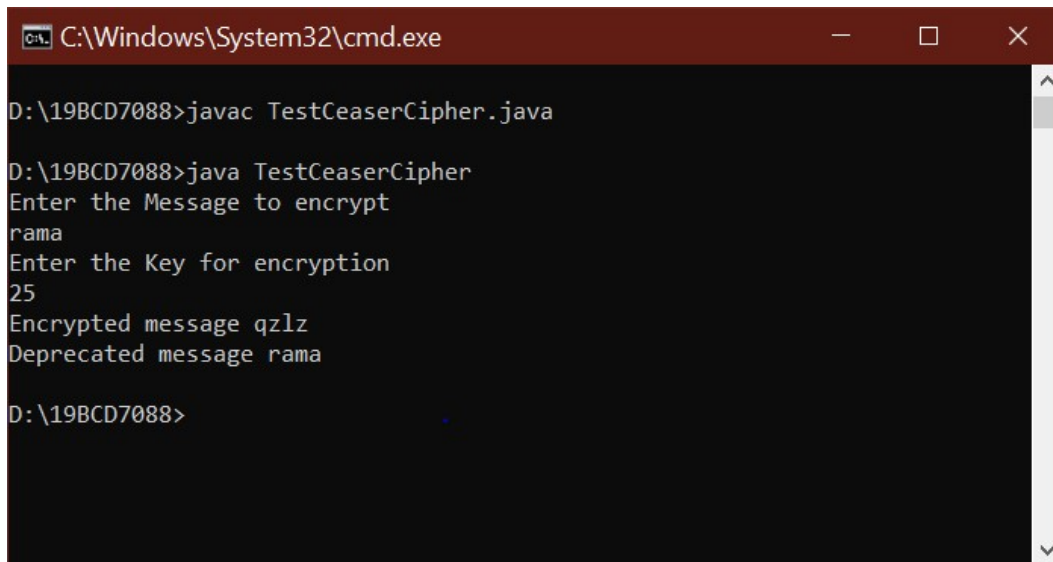
        System.out.println("Encrypted message " + c);

        System.out.println("Deprecated message " + mDecryption(c, key));

    }

}

```



```

C:\Windows\System32\cmd.exe

D:\19BCD7088>javac TestCeaserCipher.java

D:\19BCD7088>java TestCeaserCipher
Enter the Message to encrypt
rama
Enter the Key for encryption
25
Encrypted message qz1z
Deprecated message rama

D:\19BCD7088>

```

2.

b.

Jennifer comes with a message "gdhrzfnncanx". She wants to perform reverse substitution using mDecryption method but not aware of key 'k'. To help her, develop a multithreaded program to create separate thread for each possible key 'k' and print all reverse substitutions. Do necessary changes to CeaserCipher class and provide synchronization for threads if the output from threads are mixed. Name the file as BruteForceCeaserCipher.java.

```

public class BruteForceCeaserCipher extends Thread
{
    public static String m = "gdhrzfnncanx";
    public static String a = "abcdefghijklmnopqrstuvwxyz";
    static class Decrypt extends Thread
    {
        int key;

        Decrypt(int key)
        {

```



```

        this.key=key;
    }
    String l = m.toLowerCase();
String p = "";
public synchronized void decrypt(){
    for (int i = 0; i < l.length(); i++)
        {
            int cp = a.indexOf(l.charAt(i));
            int kv = (cp - key) % 26;
            if (kv < 0)
            {
                kv = a.length() + kv;
            }
            char r = a.charAt(kv);
            p += r;
        }
        System.out.println("Message for key "+key+ " is " + p);
    }

    public void run(){
        decrypt();
    }
}

public static void main(String[] args)
{
    BruteForceCeaserCipher.Decrypt [] c=new BruteForceCeaserCipher.Decrypt[26];
    for(int k=0;k<26;k++){
        c[k]=new BruteForceCeaserCipher.Decrypt(k);
    }
    for(int y=0;y<26;y++){
        c[y].start();
    }
}

```

}

}

```
C:\Windows\System32\cmd.exe

D:\19BCD7088>javac BruteForceCeaserCipher.java

D:\19BCD7088>java BruteForceCeaserCipher
Message for key 20 is mjnxflttigtd
Message for key 24 is ifjtbhppedcpz
Message for key 18 is olpzhnvvkivf
Message for key 22 is khlvdjrrgerb
Message for key 15 is rosckqyylnyi
Message for key 19 is nkoygmuujhue
Message for key 17 is pmqaiowljwg
Message for key 12 is urvfntbbqobl
Message for key 7 is zwaksyggvtgq
Message for key 0 is gdhrzfnncanx
Message for key 16 is qnrbjpxxmkxh
Message for key 8 is yvzjrxffusfp
Message for key 9 is xuyiqweetreo
Message for key 3 is daeowckkzxku
Message for key 25 is heisagoodboy
Message for key 6 is axbltzhhwuhr
Message for key 11 is vswgouccrpcm
Message for key 13 is tquemsapnak
Message for key 2 is ebfpxdllaylv
Message for key 21 is limweksshfsc
Message for key 23 is jgkuciqqfdqa
Message for key 5 is bycmuaiixvis
Message for key 10 is wtxhpvddsqdn
Message for key 1 is fcgqyemmbzwm
Message for key 14 is sptdlrzzomzj
Message for key 4 is czdnvbjjywj

D:\19BCD7088>
```