

REGISTER NUMBER:18BIT0048

NAME: R.Senthil Kumar

Prof:- Deepikaa S

LAB SLOT: L5+L6

CSE1007:JAVA fundamentals (LAB)
DIGITAL ASSIGNMENT 5: JAVA core questions

QUESTION 1: The rainfall dataset was collected from various states of India over a period of time. Give the details of 10 states whose average are stored in a Map. Sort the states with respect to average rainfall using any sorting algorithm. Also search the state whose rainfall is the second most getting much rain.

CODE:

```
import java.util.*;
import java.io.*;
class q1
{
    public static void main(String args[])
    {
        Scanner inn=new Scanner(System.in);
        Map<String, Integer> data=new HashMap <String, Integer>();
        int i=0,j=0;
        System.out.println("Enter data for 10 states: ");
        Integer a;
        String s;
        for(i=0;i<10;i++)
        {
            System.out.println("State name: ");
            s=inn.nextLine();
            if(i!=0)s=inn.nextLine();//this is to ignore the enter key
            pressing

            System.out.println("Average rainfall (in cm): ");
            a=inn.nextInt();

            data.put(s,a);
        }

        System.out.println("Displaying HashMap...");
        Set <String>keys=data.keySet();
        Iterator <String> itr=keys.iterator();
        while(itr.hasNext())
        {
            s=itr.next();
            System.out.println("Key: "+s+" and value: "+data.get(s));
        }
    }
}
```

```

    }

    //Sorting algorithm : - Sorting technique implemented in java 8 for
    linkedhashmaps.
    System.out.println("Sorting data with respect to avg rainfall...");
    LinkedHashMap<String, Integer> reverseSortedMap = new
    LinkedHashMap<>();

    data.entrySet().stream().sorted(Map.Entry.comparingByValue(Comparator.reverseOr
    der()))
        .forEachOrdered(x -> reverseSortedMap.put(x.getKey(),
    x.getValue()));

    System.out.println("Displaying Sorted Map...");
    Set <String>keyss=reverseSortedMap.keySet();
    Iterator <String> itr=keyss.iterator();
    int counter=0;//The counter is a variable that counts the
    position, when counter is 1, it means that the
    //state with second most rainfall is being
    printed, and this data can be saved and printed later.
    String rec="";
    while(itr.hasNext())
    {

        s=itr.next();
        if(counter==1)rec=s;
        System.out.println("Key: "+s+" and value:
    "+reverseSortedMap.get(s));
        counter++;
    }
    System.out.println("\nThe state with second most average
    rainfall is "+rec+" with avg rainfall of "+data.get(rec)+"cm");

    }
}

```

OUTPUT:

```

Enter data for 10 states:
State name:
Tamil Nadu
Average rainfall (in cm):
17
State name:
Rajasthan

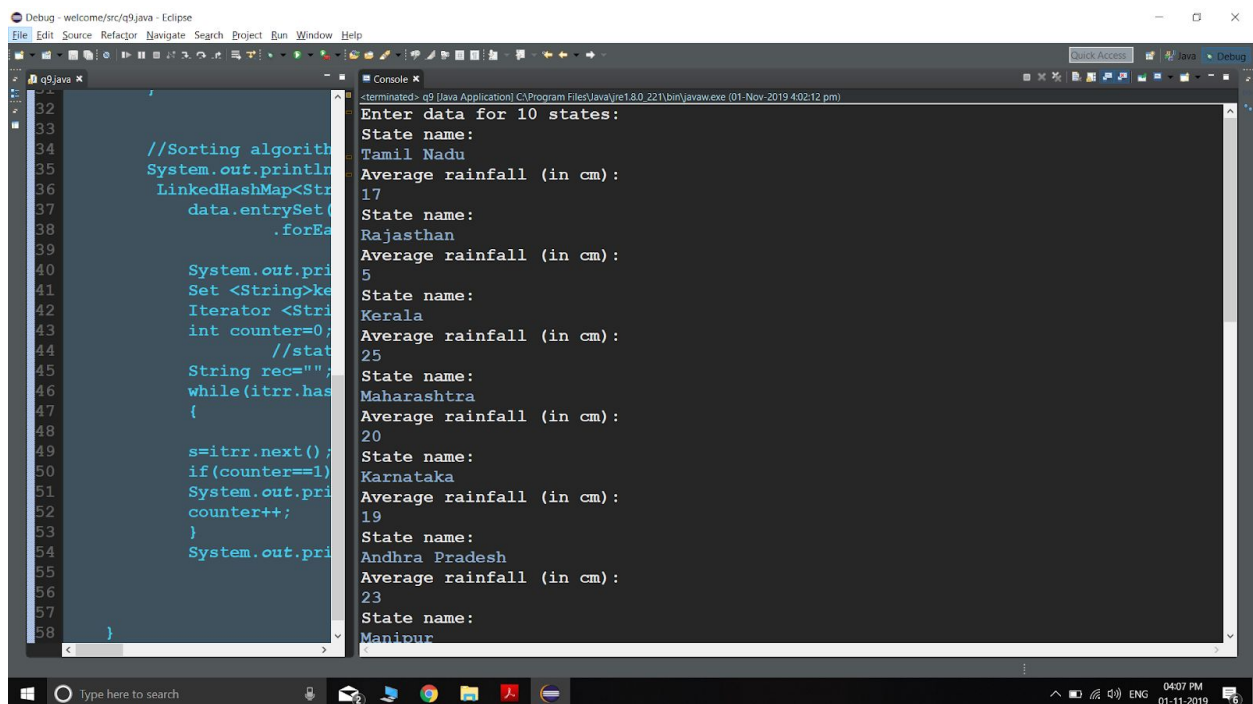
```

```
Average rainfall (in cm):
5
State name:
Kerala
Average rainfall (in cm):
25
State name:
Maharashtra
Average rainfall (in cm):
20
State name:
Karnataka
Average rainfall (in cm):
19
State name:
Andhra Pradesh
Average rainfall (in cm):
23
State name:
Manipur
Average rainfall (in cm):
30
State name:
Sikkim
Average rainfall (in cm):
13
State name:
Punjab
Average rainfall (in cm):
6
State name:
Bihar
Average rainfall (in cm):
4
Displaying HashMap...
Key: Bihar and value:      4
Key: Karnataka and value:  19
Key: Andhra Pradesh and value: 23
Key: Sikkim and value:     13
Key: Tamil Nadu and value: 17
Key: Punjab and value:     6
Key: Manipur and value:    30
Key: Maharashtra and value: 20
Key: Rajasthan and value:  5
Key: Kerala and value:     25
Sorting data with respect to avg rainfall...
Displaying Sorted Map...
Key: Manipur and value:    30
```

Key: Kerala and value: 25
Key: Andhra Pradesh and value: 23
Key: Maharashtra and value: 20
Key: Karnataka and value: 19
Key: Tamil Nadu and value: 17
Key: Sikkim and value: 13
Key: Punjab and value: 6
Key: Rajasthan and value: 5
Key: Bihar and value: 4

The state with second most average rainfall is Kerala with avg rainfall of 25cm

OUTPUT SCREENSHOT:



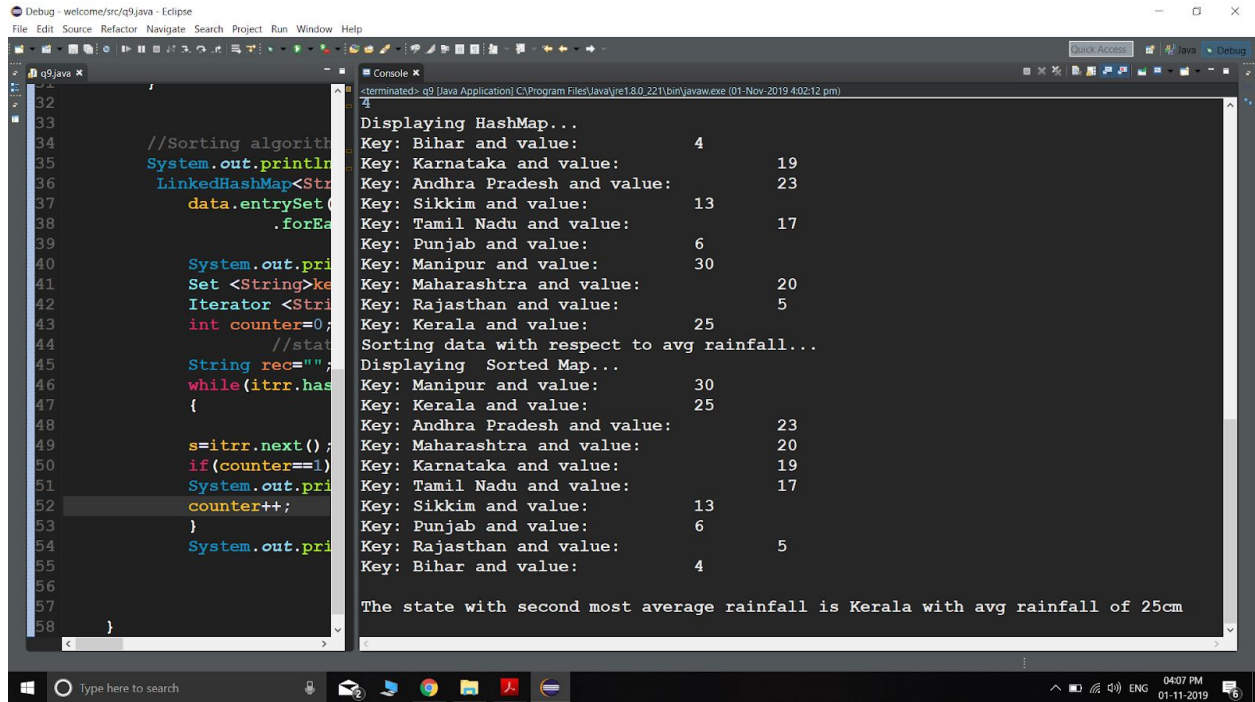
The screenshot shows the Eclipse IDE interface. The left pane displays a Java file named 'q9.java' with the following code:

```
32  
33  
34 //Sorting algorithm  
35 System.out.println  
36 LinkedHashMap<Str  
37 data.entrySet()  
38 .forEach  
39  
40 System.out.print  
41 Set <String>ke  
42 Iterator <Stri  
43 int counter=0;  
44 //stat  
45 String rec="";  
46 while(itrr.has  
47 {  
48  
49 s=itrr.next();  
50 if(counter==1)  
51 System.out.print  
52 counter++;  
53 }  
54 System.out.print  
55  
56  
57  
58 }
```

The right pane shows the console output for the program:

```
<terminated> q9 [Java Application] C:\Program Files\Java\jre1.8.0_221\bin\javaw.exe (01-Nov-2019 4:02:12 pm)  
Enter data for 10 states:  
State name:  
Tamil Nadu  
Average rainfall (in cm):  
17  
State name:  
Rajasthan  
Average rainfall (in cm):  
5  
State name:  
Kerala  
Average rainfall (in cm):  
25  
State name:  
Maharashtra  
Average rainfall (in cm):  
20  
State name:  
Karnataka  
Average rainfall (in cm):  
19  
State name:  
Andhra Pradesh  
Average rainfall (in cm):  
23  
State name:  
Manipur
```

The Windows taskbar at the bottom shows the system clock as 04:07 PM on 01-11-2019.



```
32
33
34 //Sorting algorithm
35 System.out.println("Unsorted Map...");
36 LinkedHashMap<String, Integer> sortedMap = new LinkedHashMap<>();
37 data.entrySet().forEach((entry) -> {
38     sortedMap.put(entry.getKey(), entry.getValue());
39 });
40 System.out.println("Sorted Map...");
41 Set<String> keys = sortedMap.keySet();
42 Iterator<String> itr = keys.iterator();
43 int counter=0;
44 //stat
45 String rec="";
46 while(itr.hasNext()) {
47     String key = itr.next();
48     int value = sortedMap.get(key);
49     if(counter==1) {
50         rec+=key+" and value: "+value+"\n";
51     }
52     counter++;
53 }
54 System.out.println(rec);
55
56
57
58 }
```

```
<terminated>- q9 [Java Application] C:\Program Files\Java\jre1.8.0_221\bin\javaw.exe (01-Nov-2019 4:02:12 pm)
4
Displaying HashMap...
Key: Bihar and value: 4
Key: Karnataka and value: 19
Key: Andhra Pradesh and value: 23
Key: Sikkim and value: 13
Key: Tamil Nadu and value: 17
Key: Punjab and value: 6
Key: Manipur and value: 30
Key: Maharashtra and value: 20
Key: Rajasthan and value: 5
Key: Kerala and value: 25
Sorting data with respect to avg rainfall...
Displaying Sorted Map...
Key: Manipur and value: 30
Key: Kerala and value: 25
Key: Andhra Pradesh and value: 23
Key: Maharashtra and value: 20
Key: Karnataka and value: 19
Key: Tamil Nadu and value: 17
Key: Sikkim and value: 13
Key: Punjab and value: 6
Key: Rajasthan and value: 5
Key: Bihar and value: 4
The state with second most average rainfall is Kerala with avg rainfall of 25cm
```

QUESTION 2: Implement a simple counter in Java which increments its value each second. Use sleep (1000) to delay each print. The counter stops whenever the user inserts the value 0 from console. Note that the program cannot make any assumption on when the user will press the key.

JAVA CODE:

```
import java.io.*;

class globally
{
    public static boolean flag;
    public static int count;
}

class counter
{
    boolean flag;
    int count;
    counter(int i)
    {
        count=i;
        globally.count=i;
        flag=false;

        globally.flag=false;
    }
}
```

```

void timer()
{
    while(!globally.flag)
    {
        try
        {
            java.lang.Thread.sleep(1000);
            count++;
            globally.count++;
            if(!globally.flag)
                System.out.println("time: "+this.count);
        }
        catch(InterruptedException e){}
    }
}

void checker()
{
    while(true)
    {
        System.out.println("Press zero to stop...");
        try
        {char ch = (char)System.in.read();

            System.out.println("You pushed : " + ch );
            if(ch=='0')
            {
                globally.flag=true;
                System.out.println("Counter time: "+globally.count+"seconds");
                break;
            }
        }

        catch(IOException e){}
    }
}

}

class MyThread implements Runnable
{
    Thread t;
    counter obj;
    MyThread(String s)
    {
        obj=new counter(0);
        t = new Thread(this, s);
        t.start();
    }
}

```

```

    }
    public void run()
    {
        if(t.getName()=="counter")
            obj.timer();
        else if(t.getName()=="checker")
            obj.checker();

        /*
        THERE ARE 2 THREADS: ONE FOR TIME-CLOCK, ONE FOR KEYBOARD HIT DETECTION. Once a
        keyboard hit is detected and verified that the pressed key is zero, then a
        "notify" is sent to the time-clock thread using inter thread communication
        which hence stops the clock and time is displayed.
        */

    }
}

class q2
{
    public static void main(String args[])
    {
        MyThread clock=new MyThread("counter");

        MyThread keypresser=new MyThread("checker");
        try
        {clock.t.join();
        keypresser.t.join();
        }
        catch (InterruptedException e){}
    }
}

```

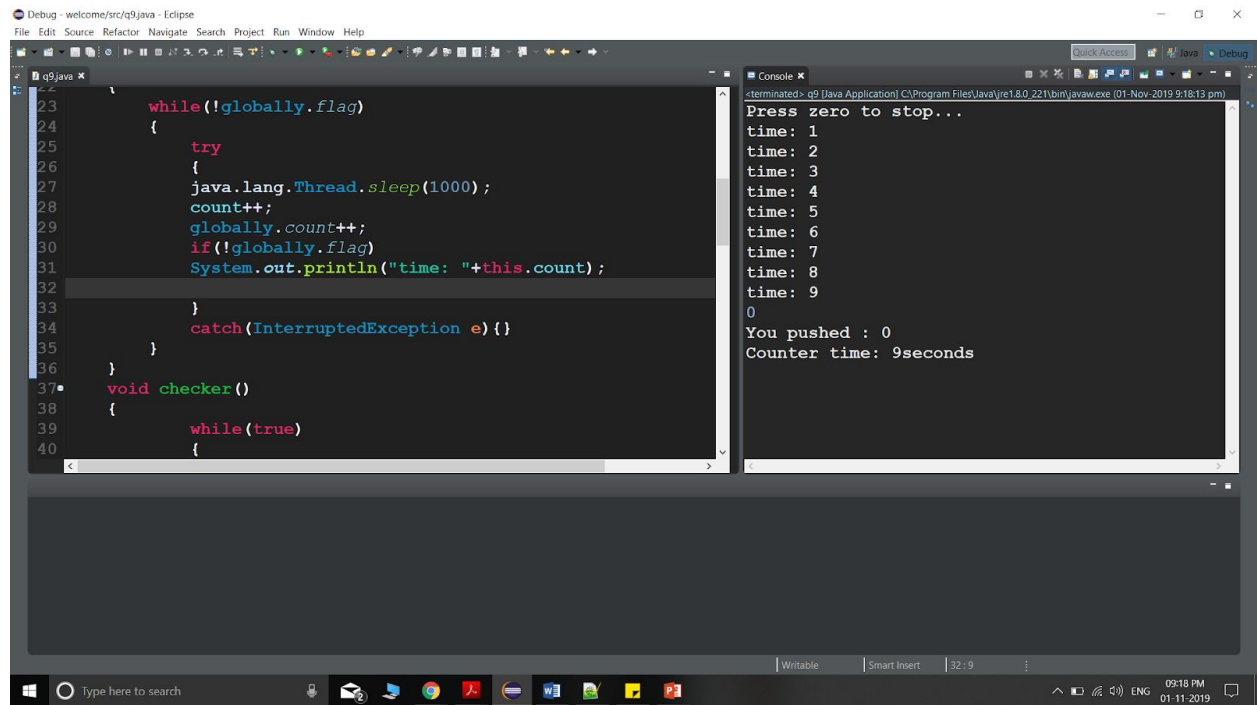
OUTPUT:

```

Press zero to stop...
time: 1
time: 2
time: 3
time: 4
time: 5
time: 6
time: 7
time: 8
time: 9
0
You pushed : 0
Counter time: 9seconds

```

OUTPUT SCREENSHOT:



The screenshot displays the Eclipse IDE interface. The main editor window shows a Java file named `q9.java` with the following code:

```
23     while(!globally.flag)
24     {
25         try
26         {
27             java.lang.Thread.sleep(1000);
28             count++;
29             globally.count++;
30             if(!globally.flag)
31                 System.out.println("time: "+this.count);
32         }
33         catch (InterruptedException e) {}
34     }
35 }
36
37 void checker()
38 {
39     while(true)
40     {
```

The console window on the right shows the output of the program:

```
<terminated> q9 [Java Application] C:\Program Files\Java\jre1.8.0_221\bin\javaw.exe (01-Nov-2019 9:18:13 pm)
Press zero to stop...
time: 1
time: 2
time: 3
time: 4
time: 5
time: 6
time: 7
time: 8
time: 9
0
You pushed : 0
Counter time: 9seconds
```

The Windows taskbar at the bottom shows the system clock as 09:18 PM on 01-11-2019.