IT5512- WEB TECHNOLOGY LAB-SESSION-10

DATE: 01/11/2021

NAME: A.S. PRUTHIEV

REG NO.2019506067

IMPLEMENT MULTITHREADED PROGRAMS USING PRIORITY AND SYNCHRONIZE

1)AIM:

To write a Java multithreaded Program to print the pattern by prioritizing the threads

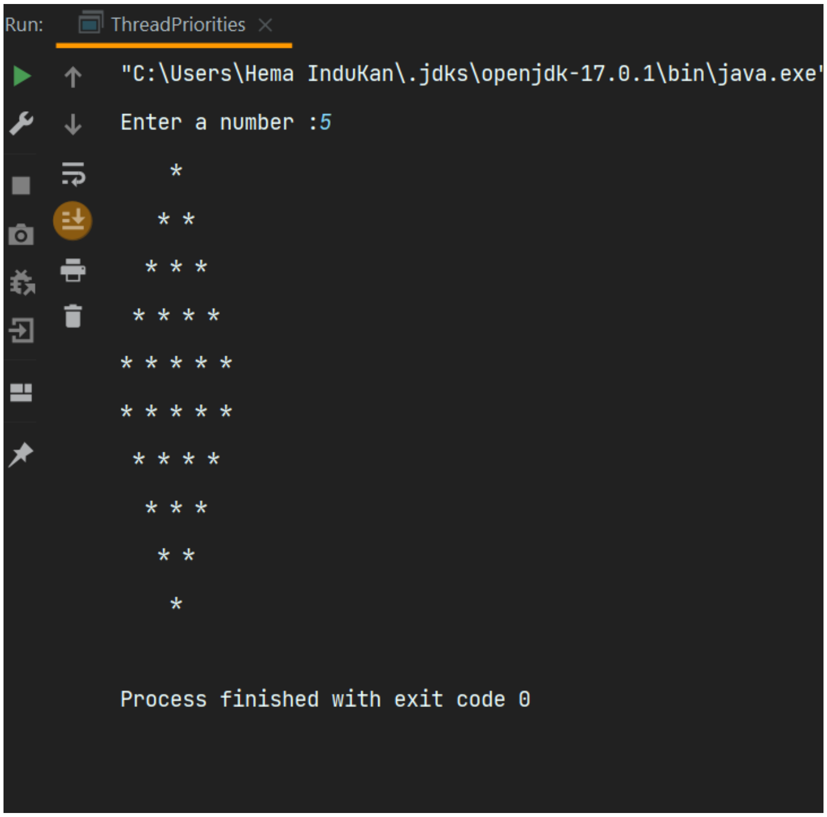
PROGRAM CODE:

```
package Java.Lab.lab9;


import java.util.Scanner;


public class ThreadPriorities{
   private static final Scanner input = new Scanner(System.in);
   public static void printTopPattern(int n){
      for(int i = n-1,k = 0 ; i >= 0 ;  i--,k++){
         for(int j = 0 ; j < i ; j++){
            System.out.print(" ");
         }
         for(int l = 0 ; l <= k ; l++){
            System.out.print("* ");
         }
         System.out.println();
      }
```

```
}
public static void printBottomPattern(int n){
   for(int i = n-1,k = 0 ; i >=0 ; i --,k ++){
      for(int j = n-1 ; j > i ; j--){
         System.out.print(" ");
      }
      for(int l = 0 ; l <= i ; l++){
         System.out.print("* ");
      }
      System.out.println();
   }
}
public static void main(String[] args) {
   int n;
   System.out.print("Enter a number :");
   n = input.nextInt();
   Thread t1 = new Thread(() -> {
     printTopPattern(n);
   });
   Thread t2 = new Thread(() -> {
     printBottomPattern(n);
   });
   t1.setPriority(10);
   t2.setPriority(9);
   t1.start();
   t2.start();
```

```
    }
}
```

OUTPUT :



```
"C:\Users\Hema InduKan\.jdks\openjdk-17.0.1\bin\java.exe'
Enter a number :5
    * 
   * * 
  * * * 
 * * * * 
* * * * * 
* * * * * 
 * * * * 
  * * * 
   * * 
    * 


Process finished with exit code 0
```

RESULT : Thus the program has been executed successfully.

2)AIM:

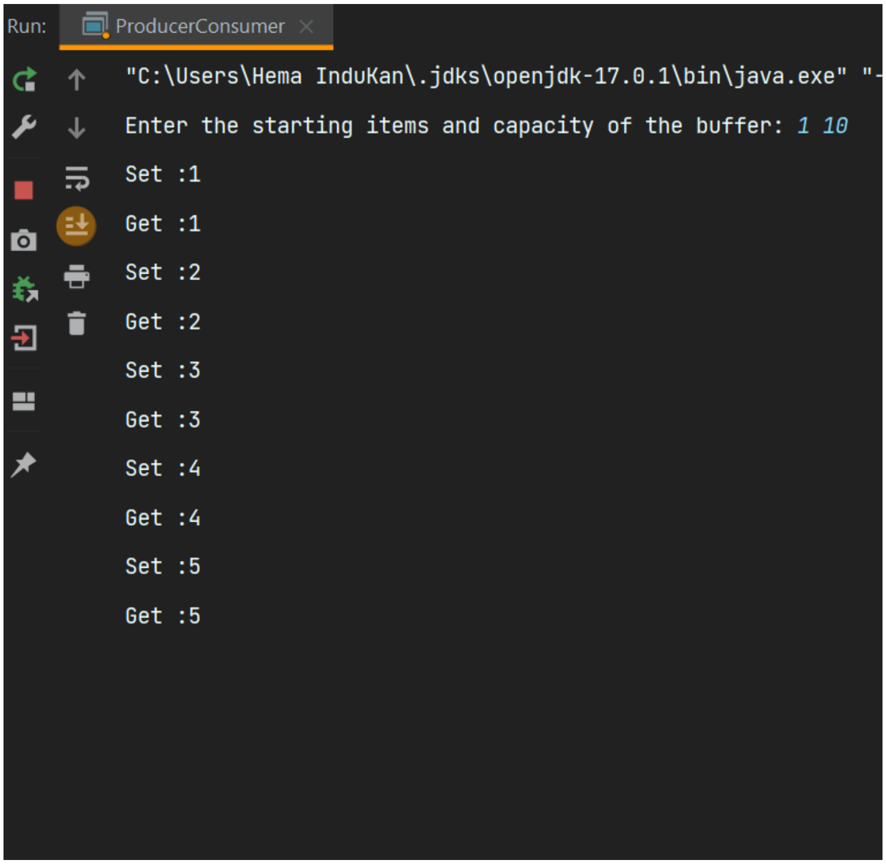
To write a Java multithreaded Program for producer and consumer problem using synchronized keyword

PROGRAM CODE:

```
package Java.Lab.lab9;

import java.util.LinkedList;
import java.util.Scanner;

class SetGetItem1 {
   private int capacity;
   private LinkedList<Integer> values = new LinkedList<>();

   SetGetItem1(int capacity) {
      this.capacity = capacity;
   }

   public int getCapacity() {
      return capacity;
   }

   public synchronized void setItem(int item) {
      while (values.size() == capacity) {
         try {
            wait();
         } catch (Exception e) {
            System.out.println(e.getMessage());
         }
      }
      values.add(item);
      notifyAll();
   }

   public synchronized int getItem() {
      while (values.isEmpty()) {
         try {
            wait();
         } catch (Exception e) {
            System.out.println(e.getMessage());
         }
      }
      notify();
      return values.pollFirst();
   }
}

class Producer1 implements Runnable {
   private SetGetItem1 setGetItem;
   private int item;

   public Producer1(int item, SetGetItem1 setGetItem) {
      this.item = item;
```

```
        this.setGetItem = setGetItem;
    }

    @Override
    public void run() {
        while (true) {
            System.out.println("Set :" + item);
            setGetItem.setItem(item++);
            try {
                Thread.sleep(1000);
            } catch (Exception e) {
                System.out.println(e.getMessage());
            }
        }
    }
}

class Consumer1 implements Runnable {
    private SetGetItem1 setGetItem;

    public Consumer1(SetGetItem1 setGetItem) {
        this.setGetItem = setGetItem;
    }

    @Override
    public void run() {
        while (true) {
            int Get = setGetItem.getItem();
            System.out.println("Get :" + Get);
            try {
                Thread.sleep(1000);
            } catch (Exception e) {
                System.out.println(e.getMessage());
            }
        }
    }
}

public class ProducerConsumer {
    private static final Scanner input = new Scanner(System.in);

    public static void main(String[] args) {
        System.out.print("Enter the starting items and capacity of the buffer: ");
        int number = input.nextInt();
        int cap = input.nextInt();
        SetGetItem1 setGetItem = new SetGetItem1(cap);
        new Thread(new Producer1(number, setGetItem)).start();
        new Thread(new Consumer1(setGetItem)).start();
    }
}
```

OUTPUT :



```
"C:\Users\Hema InduKan\.jdks\openjdk-17.0.1\bin\java.exe" "-
Enter the starting items and capacity of the buffer: 1 10
Set :1
Get :1
Set :2
Get :2
Set :3
Get :3
Set :4
Get :4
Set :5
Get :5
```

RESULT : Thus the program has been executed successfully.