06/02/2024

Lab Program 8

Q. Java program to work with threads.

Ans:

```
class MessageThread extends Thread {
    private final String message;
    private final long interval;
    MessageThread (String message, long interval) {
        this.message = message;
        this.interval = interval;
    }
    public void run () {
        try {
            while (true) {
                System.out.println (message);
                Thread.sleep (interval);
            }
        }
        catch (InterruptedException e) {
            System.out.println (Thread.currentThread ().getName()+
                                " interrupted.");
        }
    }
}

public class TwoThreadDemo {
    public static void main (String args[]) {
        MessageThread t1 = new MessageThread ("BMS College of
                Engineering", 10000);
        MessageThread t2 = new MessageThread ("CSE", 2000);
```

```
        t1.setName("Thread 1");
        t2.setName("Thread 2");
        t1.start;
        t2.start;
        try{
            Thread.sleep(20000);
        }
        catch(InterruptedException e){
            System.out.println("Main thread interrupted");
        }
        t1.interrupt();
        t2.interrupt();
        System.out.println("Main thread exiting");
    }
}
```

Output :

```
BMS College of Engineering
CSE → After 2 seconds
CSE → " "
CSE → " "
CSE → " "
CSE → " "
BMS College of Engineering → After 10 seconds
CSE → After 2 seconds
CSE → " "
CSE → " "
CSE → " "
CSE → " "
Main thread exiting → After 20 seconds
Thread 2 interrupted
Thread 1 interrupted.
```

06/02/2024

Lab Program 10

1. Java Program to demonstrate inter process communication.

Ans.

```
class Q {
    int n;
    boolean valueSet = false;
    synchronized int get(){
        while(!valueSet)
            try {
                System.out.println("\nConsumer Waiting\n");
                wait();
            }
            catch(InterruptedException e){
                System.out.println("InterruptedException caught");
            }
        System.out.println("Got: " + n);
        valueSet = false;
        System.out.println("\nIntimate Producer\n");
        notify();
        return n;
    }
    synchronized void put(int n){
        while(valueSet)
            try{
                System.out.println("\nProducer Waiting\n");
                wait();
            }
            catch(InterruptedException e){
                System.out.println("InterruptedException caught");
            }
        this.n = n;
        valueSet = true;
```

```
        System.out.println("Put: " + n);
        System.out.println("\nIntimate Consumer\n");
        notify();
    }
}

class Producer implements Runnable {
    Q q;
    Producer(Q q){
        this.q = q;
        new Thread(this, "Producer").start();
    }
    public void run(){
        int i = 0;
        while(i<5){
            q.put(i++);
        }
    }
}

class Consumer implements Runnable {
    Q q;
    Consumer(Q q){
        this.q = q;
        new Thread(this, "Consumer").start();
    }
    public void run(){
        int i = 0;
        while(i<5){
            int r = q.get();
            System.out.println("Consumed: " + r);
            i++;
        }
    }
}
```

```
class PCFixed {
    public static void main(String args[]) {
        Q q = new Q();
        new Producer(q);
        new Consumer(q);
        System.out.println("Press Control-C to stop");
    }
}
```

Output :

```
Put : 0
Intimate Consumer
Producer waiting
Press Control-C to stop
Got : 0
Intimate Producer
Consumed : 0
Put : 1
Intimate consumer
Producer waiting
Got : 1
Intimate Producer
consumed : 1
Put : 2
Intimate Consumer
Producer waiting
Got : 2
Intimate Producer
consumed : 2
Put : 3
Intimate Consumer
Producer waiting
```

Got : 3
Intimate Producer
Put : 4
Intimate Consumer
Consumed : 3
Got : 4
Intimate Producer
consumed : 4

6/2/2024