**ASSGINMENT-10**

20BCSE50\_Kumar Jijnasu\_CSE-C1-08

1..

*OddThread.java*

public class OddThread extends Thread {

    int m,n;

    public OddThread(int m,int n){

        this.m=m;

        this.n=n;

    }

    public void run(){

        for(int i=m;i<n;i++){

            if(i%2==0){

                System.out.println("child : "+i);

            }

        }

    }

}

*EvenThread.java*

import java.util.Scanner;

public class EvenThread{

    public static void main(String[] args) {

        Scanner sc = new Scanner(System.in);

        System.out.print("Enter two no.s: ");

        int m=sc.nextInt(), n=sc.nextInt();

        OddThread thread1 = new OddThread(m,n);

        thread1.start();

        for(int i=m;i<n;i++){

            if(i%2!=0){

                System.out.println("parent : "+i);

            }

        }

    }

}

*2..*

*SqThread.java*

public class SqThread extends Thread {

    int n;

    public SqThread(int n)

    {

        this.n=n;

    }

    public void run()

    {

        System.out.println("square : "+(n\*n));

    }

}

*SumOfSqThread.java*

public class SumOfSqThread extends Thread{

    int n;

    public SumOfSqThread(int n) {

        this.n = n;

    }

    public void run()

    {

        int res=0;

        n \*= n;

        while(n>0)

        {

            res += n%10;

            n /= 10;

        }

        System.out.println("Sum of digits of square of number : "+res);

    }

}

*Q2Test.java*

public class Q2Test {

    public static void main(String args[]) {

        int n=5;

        SqThread th=new SqThread(n);

        SumOfSqThread sth=new SumOfSqThread(n);

        th.start();

        sth.start();

    }

}

*3..*

*OddThread.java*

public class OddThread extends Thread{

    int a[],b[];

    public OddThread(int a[], int b[])

    {

        this.a = a;

        this.b = b;

    }

    public void run()

    {

        for(int i=1;i<a.length;i+=2)

            a[i] \*= b[i];

    }

}

*EvenThread.java*

public class EvenThread extends Thread{

    int a[],b[];

    public EvenThread(int a[], int b[])

    {

        this.a = a;

        this.b = b;

    }

    public void run()

    {

        for(int i=0;i<a.length;i+=2)

            a[i] \*= b[i];

    }

}

*Q3Test.java*

public class Q3Test {

    public static void main(String[] args) {

        int a[]={1,2,3,4,5},b[]={5,4,3,2,1};

        OddThread ot = new OddThread(a,b);

        EvenThread et = new EvenThread(a,b);

        ot.start();

        et.start();

        try

        {

            Thread.sleep(100);

        }

        catch(InterruptedException e)

        {

            System.out.println(e);

        }

        for (int v : a)

            System.out.print(v+" ");

    }

}

*4..*

import java.util.Scanner;

public class SyncSumArr

{

    public static int sum=0;

    public void findsum(int a[],int i, int j)

    {

        synchronized(a)

        {

            System.out.print("Thread: Elements from "+i+" to "+(j-1));

            for(;i<j;i++)

                sum += a[i];

            System.out.println(" => Sum : "+sum);

        }

    }

}

class MyThread extends Thread

{

    int a[],i,j;

    SyncSumArr obj;

    MyThread(int a[],int i, int j,SyncSumArr obj)

    {

        this.a = a;

        this.i = i;

        this.j = j;

        this.obj = obj;

    }

    public void run()

    {

        obj.findsum(a,i,j);

    }

}

class SyncDriver

{

    public static void main(String[] args) {

        int a[] = {1,2,3,4,5,6,7,8,9,10},l = 10,th = 5,pth = 2;

        Scanner sc = new Scanner(System.in);

*// INPUTS:*

        System.out.print("Enter the length of array: ");

        l = sc.nextInt();

        a = new int[l];

        System.out.print("Enter the array: ");

        for(int i=0;i<l;i++)

            a[i] = sc.nextInt();

        System.out.print("Enter the no. of threads: ");

        th = sc.nextInt();

        pth = l/th;

        System.out.print("The array : ");

        for(int i=0;i<l;i++)

            System.out.print(a[i]+" ");

        System.out.println("\n");

        MyThread threads[] = new MyThread[th];

        SyncSumArr obj = new SyncSumArr();

        for(int i=0;i<th;i++)

        {

            threads[i] = new MyThread(a,i\*pth,(i+1)\*pth,obj);

            threads[i].start();

        }

        try

        {

            for(int i=0;i<th;i++)

                threads[i].join();

        }

        catch (Exception e)

        {

                System.out.println("Thread Joining error...");

        }

        System.out.println("RESULT : "+SyncSumArr.sum);

        sc.close();

    }

}

*5..*

import java.util.Scanner;

public class SyncMinArr

{

    public static int minimum=0;

    public int min=0;

    public int min(int x,int y)

    {

*// synchronized(this)*

*// {*

            if(x<y)

                return x;

            return y;

*// }*

    }

    public void findmin(int a[],int i, int j)

    {

        synchronized(a)

        {

            System.out.print("Thread: Elements from "+i+" to "+(j-1));

            min = a[i];

            for(;i<j;i++)

                min = min(a[i],min);

            minimum = min(min,minimum);

            System.out.println(" => Min : "+min);

        }

    }

}

class MyThread extends Thread

{

    int a[],i,j;

    SyncMinArr obj;

    MyThread(int a[],int i, int j,SyncMinArr obj)

    {

        this.a = a;

        this.i = i;

        this.j = j;

        this.obj = obj;

    }

    public void run()

    {

        obj.findmin(a,i,j);

    }

}

class SyncMinDriver

{

    public static void main(String[] args) {

        int a[] = {15,25,35,4,5,6,7,8,9,10},l = 10,th = 5,pth = 2;

        Scanner sc = new Scanner(System.in);

*// INPUTS:*

        System.out.print("Enter the length of array: ");

        l = sc.nextInt();

        a = new int[l];

        System.out.print("Enter the array: ");

        for(int i=0;i<l;i++)

            a[i] = sc.nextInt();

        System.out.print("Enter the no. of threads: ");

        th = sc.nextInt();

        pth = l/th;

        System.out.print("The array : ");

        for(int i=0;i<l;i++)

            System.out.print(a[i]+" ");

        System.out.println("\n");

        SyncMinArr.minimum = a[0];

        MyThread threads[] = new MyThread[th];

        SyncMinArr obj = new SyncMinArr();

        for(int i=0;i<th;i++)

        {

            threads[i] = new MyThread(a,i\*pth,(i+1)\*pth,obj);

            threads[i].start();

        }

        try

        {

            for(int i=0;i<th;i++)

                threads[i].join();

        }

        catch (Exception e)

        {

                System.out.println("Thread Joining error...");

        }

        System.out.println("RESULT : "+SyncMinArr.minimum);

        sc.close();

    }

}

*6..*

*//SIR… THIS PROGRAM IS NOT WORKING.. I HAD ALSO SENT YOU THIS PROGRAM IN WHATSAPP AFTER YOU TOLD IN LAB*

class Producer extends Thread

{

    Consumer c;

    int n;

    Producer(){}

    public Producer(Consumer c,int n) {

        this.c = c;

        this.n = n;

    }

    synchronized public void generate()

    {

        long x;

        for(int i=0;i<n;i++)

        {

            x = Math.round((Math.random()\*100));

            System.out.println("Number Generated by Producer: "+x);

            c.num = x;

            c.update();

            try

            {

                wait();

            }

            catch(Exception e)

            {

                System.out.println("Waiting Error: "+e);

            }

        }

        System.out.println("The even numbers received by producer : "+c.evens);

    }

    public void run()

    {

        generate();

    }

}

class Consumer extends Thread

{

    public int evens=0;

    long num;

    public Consumer() {}

    synchronized public void update()

    {

        if(num%2==0)

            evens++;

        notify();

    }

    public void run()

    {

        update();

    }

}

public class WaitDemo {

    public static void main(String[] args) {

        Producer p= new Producer(new Consumer(),10);

        p.start();

    }

}