Faculty of Technology and Engineering

Chandubhai S Patel Institute of Technology

Department of Computer Science & Engineering

Date: 29 / 09 / 2022

Academic Year	:	2022-23	Semester	:	3
Course code	:	CE251	Course name	:	Java Programming

Part - 3

Practical - 1

Aim	Write a program to create thread which display “Hello World” message. A. by extending Thread class B. by using Runnable interface

Code

```java
// package Practicle_file;

/*
   Name: Drash Aswani
   ID  : 21ce006
   Pr  : Write a program to create thread which display “Hello
World” message.
         A. by extending Thread class
         B. by using Runnable interface.
*/

public class _006_6_1 {
    public static void main(String[] args) {
        Thread1 t1 = new Thread1();
        t1.start();
        Thread t = new Thread(new Thread2());
        t.start();
    }
}

class Thread1 extends Thread {
    public void run() {
        System.out.println("Hello world. " + "thread class is
been extended by this thread");
    }
}

class Thread2 implements Runnable {
    public void run() {
        System.out.println("Hello world. " + "runnable interface
is been imolemented by this thread");
    }
}
```

Practical-2

Aim	Generate 15 random numbers from 1 to 100 and store it in an int array. Write a program to display the numbers stored at odd indexes by thread1 and display numbers stored at even indexes by thread2.

Code

```java
/*
   Name: Darsh Aswani
   ID  : 21ce006
   Pr  : Generate 15 random numbers from 1 to 100 and store it in an
int array. Write a program to
         display the numbers stored at odd indexes by thread1 and
display numbers stored at even
         indexes by thread2.
*/

import java.util.Random;

public class _006_6_2 {
    public static void main(String[] args) {
        int[] array = new int[15];
        //random elements in array
        for (int i = 0; i < 15; i++) {
            array[i] = new Random().nextInt(100);
        }

        System.out.println("The array is : ");
        for (int i = 0; i < 15; i++) {
            System.out.print(array[i]+" ");
        }

        System.out.println("\n\nStarting thread...");

        //thread for printing odd indexes of array
        Thread thread1 = new Thread(){
            public void run(){
                for (int i = 1; i < 15; i=i+2) {
                    System.out.println("Odd : "+array[i]);
                }
            }
        };
```

```
        //thread for printing even indexes of array
        Thread thread2 = new Thread(){
            public void run(){
                for (int i = 0; i < 15; i=i+2) {
                    System.out.println("Even : "+array[i]);
                }
            }
        };

        thread1.start();
        thread2.start();
    }
}
```

Practical - 3

Aim	Write a program to increment the value of one variable by one and display it after one second using thread using sleep() method.

Code

```java
/*
   Name: Darsh Aswani
   ID  : 21ce006
   Pr  : Write a program to increment the value of one variable by one
and display it after one
         second using thread using sleep() method.
*/

public class _006_6_3 {
    public static void main(String[] args) {
        int var;
        for (var = 1; var <= 10; var++) {
            try {
                Thread.sleep(1000);
            } catch (InterruptedException e) {
                System.out.println(e.getMessage());
            }
            System.out.print(var + " ");
        }
    }
}
```

Practical - 4

Aim	Write a program to create three threads ‘FIRST’, ‘SECOND’, ‘THIRD’. Set the priority of the ‘FIRST’ thread to 3, the ‘SECOND’ thread to 5(default) and the ‘THIRD’ thread to 7.

Code

```java
/*
   Name: Darsh Aswani
   ID  : 21ce006
   Pr  : Write a program to create three threads ‘FIRST’, ‘SECOND’,
‘THIRD’. Set the
         priority of the ‘FIRST’ thread to 3, the ‘SECOND’ thread to
5(default) and the
         ‘THIRD’ thread to 7.
*/

public class _006_6_4 {
    public static void main(String[] args) {
        FIRST fobj = new FIRST();
        fobj.setName("FIRST");
        SECOND sobj = new SECOND();
        sobj.setName("SECOND");
        THIRD tobj = new THIRD();
        tobj.setName("THIRD");

        fobj.setPriority(3);
        sobj.setPriority(5);
        tobj.setPriority(7);

        fobj.start();
        sobj.start();
        tobj.start();
    }
}

class FIRST extends Thread {
    public void run() {
        System.out.println("Priority of thread " +
Thread.currentThread().getName() + " : "
                + Thread.currentThread().getPriority());
    }
}

class SECOND extends Thread {
    public void run() {
        System.out.println("Priority of thread " +
Thread.currentThread().getName() + " : "
                + Thread.currentThread().getPriority());
    }
}
```

```
class THIRD extends Thread {
    public void run() {
        System.out.println("Priority of thread " +
Thread.currentThread().getName() + " : "
                + Thread.currentThread().getPriority());
    }
}
```

Practical - 5

Aim	Write a program to solve producer-consumer problem using thread Synchronization.

Code

```java
/*
   Name: Darsh Aswani
   ID  : 21ce006
   Pr  : Write a program to solve producer-consumer problem using
thread Synchronization.
*/

/*
 * This program has 2 threads Producer(pt) and Consumer(ct).
 * Producer thread will add values to LinkedList (2 values at a time).
 * Consumer will be able to retrive these values only after Producer
has Produced some values.
 * Both Thread will execute synchronously.
 */

public class _006_6_5 {
    public static void main(String[] args) throws InterruptedException
{
        ProducerConsumer pc = new ProducerConsumer();

        // Creating a producer thread
        Thread pt = new Thread(new Runnable() {
            @Override
            public void run() {
                try {
                    pc.produce();
                } catch (InterruptedException e) {
                    e.printStackTrace();
                }
            }
        });

        // Creating consumer thread
        Thread ct = new Thread(new Runnable() {
            @Override
            public void run() {
                try {
                    pc.consume();
                } catch (InterruptedException e) {
                    e.printStackTrace();
                }
            }
        });
```

```
        // Start both threads
        pt.start();
        ct.start();

        // producer finishes before consumer
        pt.join();
        ct.join();
    }
}
// ProducerConsumer file
/*
   Name: Darsh Aswani
   ID  : 21ce006
   Pr  : Write a program to solve producer-consumer problem using
thread Synchronization.
*/

import java.util.LinkedList;

public class ProducerConsumer {
    LinkedList<Integer> list = new LinkedList<>();
    int capacity = 2; // assume the capacity is 2.

    synchronized void produce() throws InterruptedException {
        int value = 1;
        while (true) {
            // producer thread waits while list is full
            while (list.size() == capacity) {
                wait();
            }

            // producing a value...
            System.out.println("Value produced  --> " + value);
            list.add(value++);

            // notifies that values are produced and ready to consume
            notify();
            Thread.sleep(3000);
        }
    }

    synchronized void consume() throws InterruptedException {
        while (true) {
            // waits till the list is empty
```

```
            while (list.size() == 0) {
                wait();
            }

            // to retrieve the first job in the list
            int val = list.removeFirst();

            System.out.println("Value consumed -->" + val);

            // notifies producer thread
            notify();

            Thread.sleep(3000);
        }
    }
}
```