Institute of Computer Technology

B. Tech Computer Science and Engineering

Subject: OOP (2CSE303)

**Assignment-3**

**1. Write an application that executes two threads. One thread displays “Good Morning” every 1000 milliseconds & another thread displays “Good Evening” every 3000 milliseconds along with their respective thread id followed by a message “Good Bye..”. Create the threads by implementing the Runnable interface.**

***SOLUTION***

package assignment3;

/\*\*

\*

\* @author YashPrajapati

\*/

class Morning implements Runnable

{

public void run()

{

for(int i=0;i<5;i++)

{

try

{

Thread.sleep(1000);

}

catch(Exception e)

{

System.out.println(e);

}

System.out.println("Good Morning. This is thread "+Thread.currentThread().getId());

}

}

}

class Evening implements Runnable

{

public void run()

{

for(int i=0;i<5;i++)

{

System.out.println("Good Evening. This is thread "+Thread.currentThread().getId());

try

{

Thread.sleep(3000);

}

catch(Exception e)

{

System.out.println(e);

}

}

System.out.println("Good Bye...");

}

}

public class A3Q1 {

public static void main(String[] args) {

Morning T1=new Morning();

Evening T2=new Evening();

Thread td=new Thread(T1);

Thread td1=new Thread(T2);

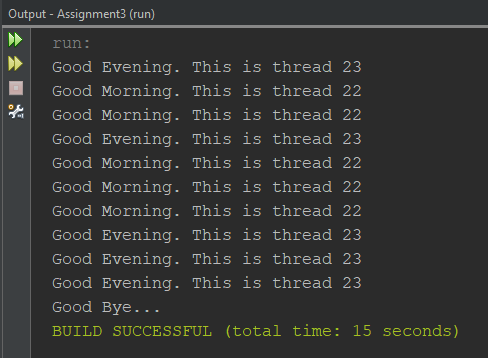
td.start();

td1.start();

}

}

***OUTPUT***

******

**2. Implement a Java based program to create two threads, one thread will print odd numbers and second thread will print even numbers between 1 to 100 numbers.**

***SOLUTION***

package assignment3;

/\*\*

\*

\* @author YashPrajapati

\*/

class odd extends Thread {

public void run() {

for (int i = 1; i < 100; i = i + 2) {

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

}

}

}

class even extends Thread {

public void run() {

for (int i = 2; i <= 100; i = i + 2) {

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

}

}

}

public class A3Q2 {

public static void main(String[] args) {

odd T1 = new odd();

even T2 = new even();

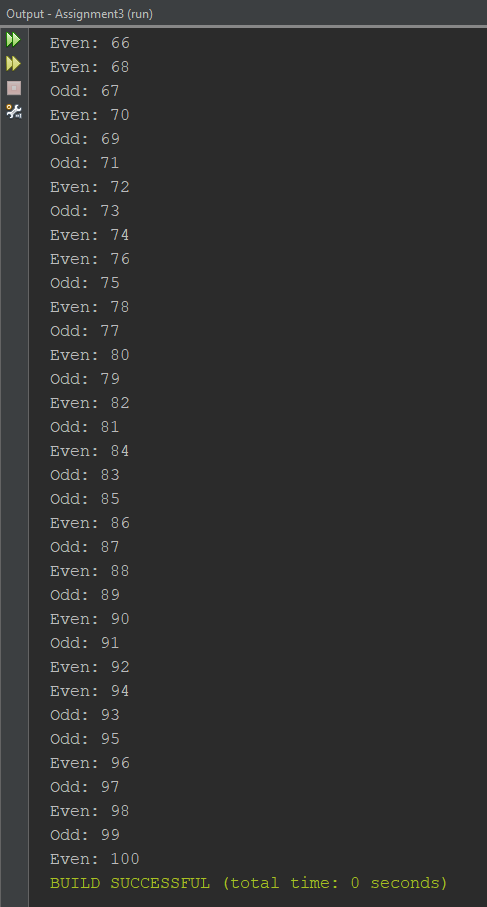
T1.start();

T2.start();

}

}

***OUTPUT***

******

**3. Write a Java program to spawn three threads with following actions: a. Set the second thread to maximum priority while the first and third thread should have normal priority b. Execution of only threads with priority less than 8 should take place. c. Also, print the alive status of all the three threads.**

***SOLUTION***

package assignment3;

/\*\*

\*

\* @author YashPrajapati

\*/

class MultithreadingDemo extends Thread{

@Override

public void run() {

if (Thread.currentThread().getPriority()<8) {

System.out.println("Thread Running \nID is "+Thread.currentThread().getId());

System.out.println("Priority for "+Thread.currentThread().getName()+" is: "+Thread.currentThread().getPriority());

for (int i = 0; i < 3; i++) {

System.out.println("Thread is currently active. ");

}

}

}

}

public class A3Q3 {

public static void main(String[] args) {

MultithreadingDemo t1=new MultithreadingDemo();

MultithreadingDemo t2=new MultithreadingDemo();

MultithreadingDemo t3=new MultithreadingDemo();

t1.start();

t2.start();

t3.start();

t1.setPriority(Thread.MIN\_PRIORITY);

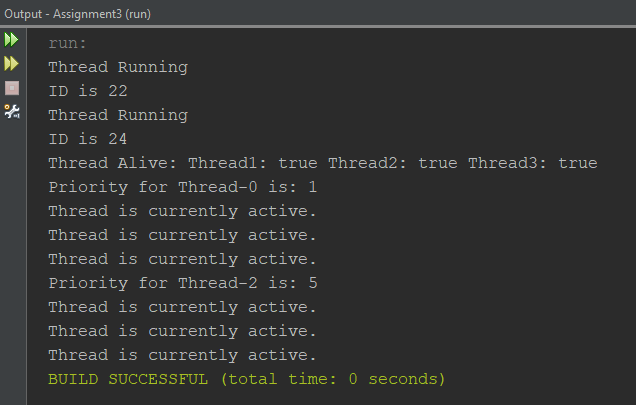
t2.setPriority(Thread.MAX\_PRIORITY);

System.out.println("Thread Alive: Thread1: "+t1.isAlive()+" Thread2: "+t2.isAlive()+" Thread3: "+t3.isAlive());

}

}

***OUTPUT***

******