**CSCI465 Operating System**

**Mohamad Al Ali | 31630038**

**Notes:**

* **The assignment should be send by email on Friday April 12 at most**
* **You should send it as a single word file**
* **A -10 per day penalty will be applied to late projects (weekends count as two days)**
* **A ZERO will be applied for late assignments on the next day**
* **Question 1 answer should be a single paragraph only for each sub question**

**Question 1:**

1. What is the difference between threads and process?

A process is an executing instance of an application and A thread is a path of execution within a process. ... Another difference between a thread and a process is that threads within the same process share the same address space, whereas different processes do not.

1. Discusses the need for synchronization in multithreaded programs.

Threads share the same memory space, i.e., they can share resources. However, there are critical situations where it is desirable that only one thread at a time has access to a shared resource. For this Java provides synchronization to control access to shared resources. For the consistency of data Synchronization is used. Without synchronization it is possible for one thread to modify a shared object while another thread is in the process of using or updating that object's value. This often leads to an error.

1. What is the effect of calling the start method on a Thread object?

A class implementing Runnable is nothing special, it just has a run method. Thread#start is a natively implemented method that creates a separate thread and calls Thread 's run method, executing the code in the new thread.

1. Which is faster, creating processes or Threads? Justify your answer.

Inter-thread communication (sharing data etc.) is significantly simpler to program than inter-process communication. Context switches between threads are faster than between processes. That is, it's quicker for the OS to stop one thread and start running another than do the same with two processes.

**Question 2:**

Write a java program TestThreadMany.java that takes a positive integer n and creates exactly n threads that print out their own name.

class MyThread extends Thread {

public MyThread (String s) {

super(s);

}

public void run() {

System.out.println("Run: "+ getName());

}

}

class TestThread {

public static void main (String arg[]) {

Scanner input = new Scanner(System.in);

System.out.println("Please input the number of Threads you want to create: ");

int n = input.nextInt();

for (int x=0; x<n; x++)

{

MyThread temp= new MyThread("Thread Number" + x);

temp.start();

System.out.println("Started Thread:" + x);

}

}

}

**Question 3:**

Write a program that runs two threads, each thread randomizes a number between 1 and 50. The main runs a third thread that calculates and prints the maximum of the randomized numbers.

package randomize;

import java.util.Random;

import java.lang.Runnable;

class RandomizeThread implements Runnable {

Random r = new Random();

int number;

public void run() {

number = r.nextInt(50) + 1;

}

int getNumber()

{

return number;

}

}

package randomize;

import java.lang.Thread;

public class Randomize {

public static void main(String[] args) throws InterruptedException {

RandomizeThread rt1 = new RandomizeThread();

RandomizeThread rt2 = new RandomizeThread();

RandomizeThread rt3 = new RandomizeThread();

Thread t1=new Thread(rt1);

Thread t2=new Thread(rt2);

Thread t3=new Thread(rt3);

t1.start();

t2.start();

t3.start();

t1.join();

t2.join();

t3.join();

System.out.println("Random number 1: " + rt1.getNumber());

System.out.println("Random number 2: " + rt2.getNumber());

System.out.println("Random number 3: " + rt3.getNumber());

int total = 0;

total += rt1.getNumber();

total += rt2.getNumber();

total += rt3.getNumber();

System.out.println("Total: " + total);

}

}