CSE321: Operating Systems Lab
Assignment 2: Java Threads

Deadline: September 11, 2020.

Task 1 [Marks: 10]

Write a java program that creates two threads. The first thread prints from 1 to 10. The second thread prints from 11 to 20. Then the first thread again prints from 21 to 30.

Task 2 [Marks: 10]

Find the integer in the range 1 to 100000 that has the largest number of divisors. Use 10 threads to solve the problem. By using threads, your program will take less time to do the computation when it is run on a multiprocessor computer.
At the end of the program, output the integer that has the largest number of divisors, and the number of divisors that it has. For this task, you should simply divide up the problem into 10 parts and assign one thread to each part.

Hint: Thread-0 to find the integer in the range 1 to 10000 that has the largest number of divisors.
Thread-1 to find the integer in the range 10001 to 20000 that has the largest number of divisors.
Thread-2 to find the integer in the range 20001 to 30000 that has the largest number of divisors.
.
.
.
Thread-9 to find the integer in the range 90001 to 100000 that has the largest number of divisors.

Find the integer number having the largest number of divisors from the results from Thread-0 to Thread-9.

Task 3 [Marks: 10]

Write a java program that takes an array of integers and sort it using multiple threads. You are allowed to use any divide and conquer sorting algorithm.

Bonus part: Run your java program in both single threaded (using `Thread.run()` method) and multithreaded (using `Thread.start()` method) mode and show the performance difference in elapsed time. Use the `System.nanotime()` method to calculate the execution time. Your program should run faster in multithreaded mode if you have a multicore CPU.