

**School of Engineering, Technology & Design**

**2019/20**

**MCOMD3NPC- Assessment 2**

**Distributed Sim…**

**Date: March 2020**

**General guidelines for submission**

* This is an individual submission and must be presented as your own work.
* The required date of submission is 5th May 14:00 via Turnitin on Blackboard.
* This assignment has been set by Vijay Sahota; Email: Vijay.sahota@canterbury.ac.uk
* This assignment has been moderated by Man Qi
* This assignment is worth 50% of the course marks

**General advice**

* You are required to back up your work regularly onto your N: drive and on removable storage devices. Always check the date-stamp on your files before submission.
* You **must** submit your work using the software versions we currently have on the University’s network.

**Learning Outcomes of this module:**

1. ~~Demonstrate a high level understanding of data classification.~~
2. ~~Demonstrate knowledge of existing tools to process data.~~
3. ~~Understand how relationships in data are made to further its understanding~~.
4. Implement an effective tool in a distributed environment.

**Scenario:**

You have been employed by a medical imaging company to investigate a possible solution for their technology to work over a very unreliable network of compute nodes.

You have been tasked to implement a simulation that will simulate a matrix operation (multiplication) over the stated unreliable network of nodes.

It is envisaged that an underlying messaging system will be used to communicate between calculations.

It is envisaged that you adjust/ alter the matrix multiplication operation to better suite a distributed environment.

Furthermore it is expected that an ability to select the instability of the network is available such that a tester can set the probability of failure of compute nodes to estimate the resilience of your solution.

**Note: It is expected that you produce your work in Eclipse, and as such will submit a zipped version of your project on Black board along with a separate word report in their respective buckets.**

**PLEASE NOTE:**

**All the code you produce must be your own work, and the use of Java built in models/ solutions such as synchronised are not to be used. The use of the Thread class and rand is permitted.**

**Each task should have their own classes/ implementations/ main thread – do not submit one single class that covers all tasks.**

**TASK 1: Simple implementation – gold standard:**

For this section you are to simply implement a matrix multiplication in Java for two 1000 x 1000 matrices of randomly generated numbers. The code must display in the command prompt both the original matrices and their multiplied result.

Note key snippets of code & screen shots should be included in your report.

*(~200 Words)*

[10 Marks]

**TASK 2: Thread-ification & Verification:**

For this section you are to simply implement a threaded version of matrix multiplication, where the results will be verified by the gold standard code.

Your code should simply print out two 1000 x 1000 matrices of randomly generated numbers, the results of your gold standard multiplication, the results of your threaded version, and the results of the subtraction of your gold & threaded results (which should be a matrix of all zeros).

Note key snippets of code & screen shots should be included in your report.

*(~400 Words)*

[20 Marks]

**TASK 3: Simulating the network:**

For this section you are to adopt a pattern such as thread pool, to accurately reflect steps (messages) that would normal be needed in a distributed environment along with brief justification.

Furthermore, due to speeds of modern computers you are to impose a delay of 200ms for each computational thread executed.

Marks will be awarded for suitable models chosen and the implementation of perceived thread safety/ measures taken.

Note key snippets of code & screen shots should be included in your report.

*(~700 Words)*

[30 Marks]

**TASK 4: Testing & Failure:**

For this section you are incorporate aspects of node failure, essentially the last instruction your executing thread will perform is a calculation to decide if the thread was to live or die, respectively not passing its result back to the prior requesting thread.

This instability of the network will be a probability set/ passed to the main execution thread as a parameter set once at the start of the simulation.

It is expected that you perform your own undocumented testing, but document any issues raised though testing and amendments made.

The marker will perform multiple tests and will reflect on the stability of your solution as well as aspects of scalability.

Note your main thread can be assumed to never to fail.

Note key snippets of code & screen shots should be included in your report.

*(~700 Words)*

[30 Marks]

**Quality** of report presentation, assumptions, proper referencing, etc.

[10 Marks]

**Mark Distribution:**

|  |  |  |
| --- | --- | --- |
| **Task** | **Detail** | **Marks** |
| **1** |  | **10** |
|  | Complete implementation & documented in the report. | **5-10** |
|  | Incomplete implementation & minimal documentation in the report. | **0-5** |
| **2** |  | **20** |
|  | Complete and correct threaded solution provided, & documented in the report. | **10-20** |
|  | Complete but with some errors threaded solution provided, & documented in the report. | **5-10** |
|  | Minimal – no threaded implementation | **0-5** |
| **3** |  | **30** |
|  | Complete, implementation with good consideration taken for thread safety and suitable model chosen to reflect nature of a distributed environment | **20-30** |
|  | Fair attempt to factor for thread safety and suitable model to reflect nature of a distributed environment | **10-20** |
|  | Minimal – none implementation | **0-10** |
| **4** |  | **30** |
|  | Good attempt made, network instability can be configured and the code can produce correct results for any tested instability probability. | **20-30** |
|  | Fair attempt made, network instability can be configured and the code can produce correct results but only for select/ partial tested instability probabilities. | **10-20** |
|  | Attempt is made, but poor implementation of network instability/ actual code is unstable | **0-10** |
|  |  |  |
|  | **Quality** of report presentation, assumptions, proper referencing, etc. | **0-10** |
|  | **MAX TOTAL** | **100** |