**IS303 (AA) – Concurrent Programming Project (v0.9)**

1. Requirements:

Come up with an idea of a Java application that makes use of multi-threading, and implement it. The only strict requirement is that in your scenario, it makes sense to write a multi-threaded application rather than a single-threaded one. This is an “open” project, and ideas or suggestions are deliberately excluded here so as not to limit your creativity.

This project is an opportunity for students to explore and try concepts related to concurrent programming, but not within the scope of the syllabus. One of the objectives of an “open” project is to let students do something that they are interested in.

1. Grading Rubric:

This project is worth 20% of your final IS303 grade. You will be graded on the following:

|  |  |  |
| --- | --- | --- |
| Item | Weight | Comments |
| Justification for multi-threading | Very important | It is important that multi-threading confers some kind of advantage over single-threading for your application (otherwise, why do multi-threading?) |
| Transactional integrity | Very important | All race conditions (if any) must be resolved. There is no point if your program can run very quickly but produces incorrect reports. |
| Performance | Important | Performance will be degraded by synchronizing unnecessary code (e.g. you have a lot of code in your critical section that need not be in your critical section). Bad practices such as putting all your **run()** code into a huge synchronized block should be avoided (unless there really is a good reason). Think about what can be done to improve performance of your program (the focus of this course is not on algorithms, so improving performance by choosing a better algorithm is not the focus of this course). |
| Evidence of exploration | Fairly Important | There are MANY multi-threading-related topics that are beyond the scope of the syllabus. You may want to read up about other concepts and try to implement them. Or did you expand on certain topics that were only cursorily covered in class?  Credit will be given if you show evidence of exploration of topics beyond the scope of the syllabus. |
| Innovation | Important | Credit will be given if your project is interesting. |
| Adherence to coding conventions & good practices | Important | Do adhere to standard Java coding standards (e.g. choose descriptive method/variable/class names, indent your code, insert useful comments, place classes in relevant packages etc.) Avoid convenient but potentially problematic practices (e.g. using a static variable like a global variable, writing poorly-encapsulated classes with public attributes, placing hundreds of classes into the same source file, hard-coding file paths or values that may change down the road etc.) |
| Good documentation | Important | Do insert comments in your code where relevant. Each method should be commented with a brief description of what it does.  Your project documentation (see “What to Submit”) needs to be clear, concise and accurate. |

1. What to Submit:

Please submit these items to eLearn “Assignment” by the deadline:

1. A zip file containing all your source code (including 3rd party code, if any).

In your zip file, include **compile.bat** and **run.bat**, which contains the commands required to compile and run your program. If you are using a 3rd-party jar file, make sure you include the **-cp** (classpath) option in your commands to point to the jar file.

Your instructor will unzip the file into his hard drive (e.g. c**:\temp**), and run **compile.bat** and **run.bat**. Your program should run without further “tweaking” required.

If your program requires two or more terminal windows to run (e.g. client/server scenario), please include a **readme.txt** in your zip file with clear instructions on how to run each part of your program.

1. Your project write-up in a Word document of up to 2 pages (NO MORE THAN 2 PAGES PLEASE). This word document should contain the following table properly filled-up:

|  |  |
| --- | --- |
| Item | Answer |
| Section |  |
| Name(s) | *(include both members’ names if this is a team project)* |
| Brief description of project | *Short description here* |
| Justification for multi-threading | *Tell me why your project should be multi-threaded instead of single-threaded* |
| Transactional integrity | *Were there any potential race conditions; if so, how did you resolve them? Or are there still unresolved race conditions that you are aware of?* |
| Performance | *Is there anything noteworthy you did that improved performance? (include stats if any)* |
| Evidence of exploration | *Is there anything noteworthy that you did that required additional research (of topics not covered in class)?* |
| Innovation | *Why is your project innovative?* |
| Adherence to coding conventions & good practices | *Is there anything noteworthy that you want to mention? E.g. usage of a well-known multi-threading “best-practice”, or OO design pattern etc.* |
| References/Acknowledgement | *Insert your references here.*  *If you got help from someone, do include an acknowledgment.* |

Keep your “answer” column concise. Use point form if necessary to make reading easier. It is OK to have “nil” answers. For example, you will not be penalized if you did not do something beyond the syllabus’s scope, and place “nil” in “Evidence of Exploration”. Insert diagrams if necessary.

1. Warning

**Plagiarism is a form of dishonesty. Plagiarism cases will treated as disciplinary cases, and be reported to the SIS Dean’s Office for filing. THIS IS YOUR FINAL WARNING.**

**You are allowed to use 3rd-party code in your solution, but you MUST acknowledge all 3rd-party code in your write-up. It must be very clear which part of your submission is your original work, and which part is not. Mark all code that did not originate from you using comments. Acknowledge ideas, algorithms and any other form of help you received as well. Include references to the sources (e.g. URL to GitHub, pages from a textbook, website etc.)**

END