Programming Assignment 3

Start Assignment

Due Wednesday by 11:59pm **Points** 100 **Submitting** a file upload

COP 4520 Spring 2022

Programming Assignment 3

Note 1:

Please, submit your work via Webcourses.

Submissions by e-mail will not be accepted.

Due date: Wednesday, April 13th by 11:59 PM

Late submissions are not accepted.

Note 2:

This assignment is individual.

You can use a programming language of your choice for this assignment.

If you do not have a preference for a programming language, I would recommend C++.

Problem 1: The Birthday Presents Party (50 points)

The Minotaur's birthday party was a success. The Minotaur received a lot of presents from his guests. The next day he decided to sort all of his presents and start writing "Thank you" cards. Every present had a tag with a unique number that was associated with the guest who gave it. Initially all of the presents were thrown into a large bag with no particular order. The Minotaur wanted to take the presents from this unordered bag and create a chain of presents hooked to each other with special links (similar to storing elements in a linked-list). In this chain (linked-list) all of the presents had to be ordered according to their tag numbers in increasing order. The Minotaur asked 4 of his servants to help him with creating the chain of presents and writing the cards to his guests. Each servant would do one of three actions in no particular order:

1. Take a present from the unordered bag and add it to the chain in the correct location by hooking it to the predecessor's link. The servant also had to make sure that the newly added present is also linked with the next present in the chain.
2. Write a "Thank you" card to a guest and remove the present from the chain. To do so, a servant had to unlink the gift from its predecessor and make sure to connect the predecessor's link with the next gift in the chain.
3. Per the Minotaur's request, check whether a gift with a particular tag was present in the chain or not; without adding or removing a new gift, a servant would scan through the chain and check whether a gift with a particular tag is already added to the ordered chain of gifts or not.

As the Minotaur was impatient to get this task done quickly, he instructed his servants not to wait until all of the presents from the unordered bag are placed in the chain of linked and ordered presents. Instead, every servant was asked to alternate adding gifts to the ordered chain and writing "Thank you" cards. The servants were asked not to stop or even take a break until the task of writing cards to all of the Minotaur's guests was complete.

After spending an entire day on this task the bag of unordered presents and the chain of ordered presents were both finally empty!

Unfortunately, the servants realized at the end of the day that they had more presents than "Thank you" notes. What could have gone wrong?

Can we help the Minotaur and his servants improve their strategy for writing "Thank you" notes?

Design and implement a concurrent linked-list that can help the Minotaur's 4 servants with this task. In your test, simulate this concurrent "Thank you" card writing scenario by dedicating 1 thread per servant and assuming that the Minotaur received 500,000 presents from his guests.

Problem 2: Atmospheric Temperature Reading Module (50 points)

You are tasked with the design of the module responsible for measuring the atmospheric temperature of the next generation Mars Rover, equipped with a multi-core CPU and 8 temperature sensors. The sensors are responsible for collecting temperature readings at regular intervals and storing them in shared memory space. The atmospheric temperature module has to compile a report at the end of every hour, comprising the top 5 highest temperatures recorded for that hour, the top 5 lowest temperatures recorded for that hour, and the 10-minute interval of time when the largest temperature difference was observed. The data storage and retrieval of the shared memory region must be carefully handled, as we do not want to delay a sensor and miss the interval of time when it is supposed to conduct temperature reading.

Design and implement a solution using 8 threads that will offer a solution for this task. Assume that the temperature readings are taken every 1 minute. In your solution, simulate the operation of the temperature reading sensor by generating a random number from -100F to 70F at every reading. In your report, discuss the efficiency, correctness, and progress guarantee of your program.

Grading policy:

General program design and correctness: 50%

Efficiency: 30%

Documentation including statements and proof of correctness, efficiency, and experimental evaluation: 20%

Submission:

You will submit a link to your GitHub page. Your repositories **must be private** until the next morning. **You must still push your code before the deadline, because Github will record this time. No code pushes will be accepted after the deadline.** However, we wont start grading until the next morning.

If we cannot access your repositories, or if you provide an invalid link, you will receive a 0 - *please double check your submission once it has been made*.

Late submissions will receive a 0 as per the syllabus.

This GitHub management does two good things:

1. Removes the temptation to look at other's work, because they will all be private until after the deadline.
2. Makes your life easier, as you don't have to send us invites

Also, we cannot accept any late work as all the repos will be potentially public shortly after the deadline.

Happy coding!