CSCI203 ASSIGNMENT TWO

Due 23:55 Sunday 19/09/2021

Task:

You should write a program which simulates the queuing and service of bank customers. The simulation should be **discrete**, i.e. **event driven**.

Program:

Your program should

1. first read in the number of servers or tellers for the simulation,
2. read in a file name from standard input and then open this file,
3. read data record by record from the named file,
4. perform the simulation according to the input records, and
5. output the statistics of the services as specified below.

The ***input text file*** (`a2-sample.txt`) will contain the following data:

- A set of customers, one customer per line and each line consisting of a customer's arrival time and their corresponding service time.
- The set of customers is sorted in ascending order of their arrival times
- This set is terminated by a dummy record with arrival time and service times all equal to 0.

The ***simulation*** is to be of a bank branch with ***multiple tellers (i.e. servers) and a single queue***. The simulation should be run until the last customer has left the system.

The simulation will run multiple times to gathering the statistics on the efficiency of the service with different number of tellers or servers in duty. You are required to run the simulation ***three times with the number of servers being set to 1, 2 and 4***, each running on the same input file and output the statistics as specified below.

Output to standard output for each run will consist of the following data:

- Number of people served.
- Time last service request is completed.
- Average total service time per customer (this ***includes*** the time spent in a queue).
- Average total time in service per customer (this ***excludes*** the time spent in a queue)
- Average length of the queue (this can be estimated as the ratio between total queuing time and the time last service request is completed)
- Maximum length, i.e. number of customers, of the queue.
- Total idle time for each server.

Implementation Requirement:

1) Part of the purpose of this subject is to gain an in-depth understanding of data structures and algorithms. As such, all programming tasks in this subject require you to choose appropriate data structures and algorithms, and to implement them yourself.
2) You may use any data structures or algorithms that have been presented in class up to the end of week 6. If you use other data structures or algorithms appropriate references must be provided.
3) Programs must compile and run under `gcc` (C programs), `g++` (C++ programs), java or python. Programs which do not compile and run will receive no marks.
4) Programs should be appropriately explained with comments.
5) All coding must be your own work.
6) Use of STL, Java Collection, collection framework and any third party libraries of data structures and algorithms in your language choice is disallowed.
7) If you use any references other than the lecture notes, ensue you cite them or you may receive 0 for plagiarism. A clear comment in your code is sufficient.
8) Code be sourced from textbooks, the internet, etc may also not be used unless it is correctly credited. In the event that you use code sourced in this way you will not receive marks for that part of the program.
9) If needed, you may assume the queue will never have more than 500 people.

Report:

A pdf file describing your solution and program output should be produced. This file should contain:

1. A high-level description of the overall solution strategy.
2. A list of all of the data structures used, where they are used and the reasons for their choice.
3. A list of any standard algorithms used, where they are used and why they are used.
4. The output produced by your program on the provided "`a2-sample.txt`" file. ***Use screenshot(s) to show the running and output of your program***.
5. A discussion on the output statistics corresponding to different number of tellers (servers).
6. The report should be no more than **3** pages. If it is more than **3** pages, marking will be only based on the first three pages.
7. The report pdf file should be called `<your login>_a2.pdf`.

Marking Guide:

- Programs submitted must work! A program which fails to compile or run will receive a mark of zero.
- If your program produces different output from what is reported in the pdf file, a mark of zero will be graded.
- A program which produces the correct output, no matter how inefficient the code, will receive a minimum of 50% of the program component of the mark.
- Additional marks beyond this will be awarded for the appropriateness, i.e. efficiency for this problem, of the algorithms and data structures you use.
- Programs which lack clarity, both in code and comments, will lose marks. The total mark will be determined based on both your code and the accompanying report.

Submission via Moodle:

- Source code for the assignment should be typed into a single file called `<your login>_a2.ext`, where `ext` is the appropriate file extension for the chosen language.
- Your submission will consist of two files:
  `<your login>_a2.ext` and `<your login>_a2.pdf`
  where `ext` is one of `c, cpp, java` or `py`. `<your login>` is your UOW login name