CSCI 310 – 02 (Fall 2019)

Programming Foundations

Programming Assignment 6

DUE: Sun, Nov 3, 11:59 PM (`turnin` time)

An Event–Driven Simulation with Command–Line Arguments

This is an updated variant of *Programming Problem 13–6, pp. 393–394, Carrano and Henry 7/e.*

For additional background information on this problem, be sure to read section 13.4 (*Queues and Priority Queues Application: Simulation*), pp.381–388 of our textbook.

Specifications

After having implemented the event–driven simulation of a bank that has only one teller, we are ready to expand this to a more flexible event-driven simulation. The following additional requirements are incorporated to the original problem:

1. Your program will accommodate anywhere from 1 to 10 tellers.
2. Your program will read the parameters of the simulation as command–line arguments.
3. Just as in a bank, arriving customers wait in a single line *first–come, first served* if no tellers are available. Assume tellers are numbered starting with the one closest to the front of the waiting line of customers, and customers always go to the first teller closest to the front of the line who is available.

For each teller, your program must keep track of the total number of transactions processed by that teller and the average transaction time. Since there is only one wait line to organize waiting customers, your program must keep track of the maximum wait line length, the average wait line length, and the average wait time. The statistics for each teller and for the wait line will be reported at the end of the simulation.

The only other adjustment is that arrival events are only reported once they are actually taken and served by a teller (see the sample output below).

Input

Your program will read the parameters of the simulation as command–line arguments. The first command–line argument is the filename of an input text file containing arrival times and transaction times. Each line of the file contains the arrival time and required transaction time for a customer. You may assume that the input data from the provided filename is correct. The second command–line argument is the number of tellers T to use in the simulation, $1 \leq T \leq 10$.

Your program must generate appropriate correct–use information if an incorrect number of command–line arguments are provided. Assume the if the number of command–line arguments is correct, each argument is also correct.

Sample Input

The following sample input file is from Programming Problem 13–6, pp. 393–394, Carrano and Henry 7/e. Let the text file named `simdata.txt` contain the following:

```
1 5
2 5
4 5
20 5
22 5
24 5
26 5
28 5
30 5
88 3
```

Then typing the command "`bankSim simdata.txt 1`" runs the simulation on the provided data using a single teller. Likewise, the command "`bankSim simdata.txt 2`" runs the simulation on the provided data using a two tellers.

Output

Your program will display a trace of the events executed and a summary of the computed statistics for each teller and for the single wait line.

All output should be sent to standard output and must exactly follow the format in the sample below.

Sample Output

The command "`bankSim simdata.txt 1`" generates the following output (compare this to the sample output from our P5 specs):

```
<<< BEGIN SIMULATION (1 tellers) >>>
Teller  1 processing   arrival event at time:    1    5
Teller  1 processing departure event at time:    6
Teller  1 processing   arrival event at time:    6    5
Teller  1 processing departure event at time:   11
Teller  1 processing   arrival event at time:   11    5
Teller  1 processing departure event at time:   16
Teller  1 processing   arrival event at time:   20    5
Teller  1 processing departure event at time:   25
Teller  1 processing   arrival event at time:   25    5
Teller  1 processing departure event at time:   30
Teller  1 processing   arrival event at time:   30    5
Teller  1 processing departure event at time:   35
Teller  1 processing   arrival event at time:   35    5
Teller  1 processing departure event at time:   40
Teller  1 processing   arrival event at time:   40    5
Teller  1 processing departure event at time:   45
Teller  1 processing   arrival event at time:   45    5
Teller  1 processing departure event at time:   50
Teller  1 processing   arrival event at time:   88    3
Teller  1 processing departure event at time:   91
<<< END SIMULATION >>>

>>> SIMULATION STATISTICS

           Teller number:       1   OVERALL
  Transactions processed:      10        10
Average transaction time:    4.80      4.80

        Maximum wait line length:         3
        Average wait line length:      0.80
               Average wait time:      5.60
```

Meanwhile, the command "`bankSim simdata.txt 2`" generates the following output:

```
<<< BEGIN SIMULATION (2 tellers) >>>
Teller  1 processing   arrival event at time:    1    5
Teller  2 processing   arrival event at time:    2    5
Teller  1 processing departure event at time:    6
Teller  1 processing   arrival event at time:    6    5
Teller  2 processing departure event at time:    7
Teller  1 processing departure event at time:   11
Teller  1 processing   arrival event at time:   20    5
Teller  2 processing   arrival event at time:   22    5
Teller  1 processing departure event at time:   25
Teller  1 processing   arrival event at time:   25    5
Teller  2 processing departure event at time:   27
Teller  2 processing   arrival event at time:   27    5
Teller  1 processing departure event at time:   30
```

```
Teller  1 processing   arrival event at time:   30    5
Teller  2 processing departure event at time:   32
Teller  2 processing   arrival event at time:   32    5
Teller  1 processing departure event at time:   35
Teller  2 processing departure event at time:   37
Teller  1 processing   arrival event at time:   88    3
Teller  1 processing departure event at time:   91
<<< END SIMULATION >>>

>>> SIMULATION STATISTICS

           Teller number:       1       2   OVERALL
  Transactions processed:       6       4        10
Average transaction time:    4.67    5.00      4.80

                Maximum wait line length:         0
                Average wait line length:      0.00
                       Average wait time:      0.80
```

Notice the improvement in wait times by simply adding a second teller.

Additional Requirements

You are welcome to modify the `Event` class we used in P5. You should define a `Teller` class to organize your simulation program.

Be sure to plan your design before working on your code. You should be able to reuse most of your P5 solution to solve this problem.

Deliverables

Your submission will consist of the following files, submitted using the Department of Computer Science's `turnin` facility:

- `Event.h` – header file for the required `Event` class
- `Event.cpp` – implementation file for the required `Event` class
- `Teller.h` – header file for the required `Teller` class
- `Teller.cpp` – implementation file for the required `Teller` class
- `bankSim.cpp` – driver code containing `main()` function

We want you to develop good code documentation habits. Source code solutions submitted without any meaningful documentation will receive a total score of zero (0). You may refer to the *Google C++ Style Guide* section on source code comments as a guide.

Be sure to also review and adhere to the **Coding Standards** for this course.