CS2430 SPRING 2016

Program #4 (30 points)

Dr. Lily Chang

Due Dates

1. <u>Work Plan</u> in SE Tools due: Monday, March 14 @10pm, or **-5 points**.
2. <u>Class Diagram</u> and the <u>GUI design</u> due on the K drive: Friday, March 18 @10pm, or **-5 points**.
3. <u>All Java files</u> due on the Grader: Wednesday, March 30 @ 10pm.
4. Grace date: Friday, **April 1 @10pm**, or **0 points**.
5. **Extra credit: +2 points** if demo by March 30 @5pm.
6. You MUST demo by Monday, April 4 @5pm, <u>given that **item#2, #3 and Java Doc** are ready on K drive in your group folder by the grace date, and the grader shows a submission log submitted by the grace date with 0 differences.</u>

** Files due on K drive must be zipped into a single file. The file name must begin with **group_name_prog4**.

Program Description

Platteville Burger has been offering take-out service. Recently, there is always a long line for take-out during the lunch hours. The administration wanted to know the average wait time for the customers who use the take-out service, so they can better improve the service. You are going to write a simulation program to help the administration find out the information.

You are given a set of input data, which represents the customer visits during a typical lunch hours. The input data should be implemented as follows.

`A`	– customer arrive the take-out window
`L`	– customer complete the order and leave
`C` *n*	– forward the clock by *n* time units
`S`	– print the current statistics for the simulation.
`Q`	– stop the simulation

You will use the input data to find out the following information:

- Average wait time
- Total wait time
- Number of customers that did not have to wait
- Number of customers completed the order

Program Requirement

1. Prog4 is a **group assignment**, 2 people per group. You MUST sign up with your partner by **March 8**. If you don’t sign up, I will pair you up with another student.
2. You MUST DEMO Prog4 to get the credit for this program. Both group members must attend the demo. And you MUST have everything ready on the K drive before you can demo. If you have a 0 differences submission on the grader without demoing, you **will get 0 points**!
3. Each member of the group must do a fair share of work. I will be asking questions when you demo. If it is obvious that one person has a significantly lower level of understanding than the other, that person can expect to receive a lower grade. There will be standard demo steps for all groups. The detail of the demo schedule will be announced on D2L.
4. You are required to create a work plan for Prog4. The work plan is due **Monday, March 14 @10pm**. Each person enters their own work plan. However, you MUST indicate the time you are planning to work together, and the time you will be working alone. Up to **-5 points** if you didn’t create a proper work plan.
5. You MUST use **SE tools** to log your time. Up to **-5 points** if this is not done or you did not log all your time or you did not provide specific comments as to what you were working on.
6. You must create the class diagram and the GUI design with JFrame for this program. You must identify class members and show the relationships between classes in the class diagram. The diagram and GUI design must be in **a single Word document**, which is due on the K drive on **Friday, March 18 @10pm**.

7. You are required to have the following classes. Each class must go in a separate Java file.

- `Prog4.java`—similar to Prog3.java that calls the run method of `ConsoleSimulator` class.
- `Queue.java`—a generic Queue class to simulate the waiting line. You must use the "circular array" implementation. **-2 points** if you use other version.

```
public class Queue<E>
{
   private E [] elements;
   private int front, rear, count;

   public Queue(int capacity)
   {
      elements = (E[])new Object[capacity];
      ...
   }
   ...
}
```

- `Customer.java`—keep the information of a customer: (1) a time stamp when the customer arrives, (2) a customer arrival sequence number, and (3) the total number of customer arrival during the simulation. You MUST implement one constructor, which increments `numCustomer` by 1 each time an instance of Customer is created. And you must implement `getTimeStamp()` method and `toString()` method, which returns a String with the following format: `Customer#`*n* `arrives @time` *t*, where *n* is the `CustomerNo`, and *t* is the `timeStamp.` **-2 points** if these are not done.

```
public class Customer
{
   private static int numCustomer = 0;
   private int customerNo;
   private int timeStamp;

   public Customer(int time)
   {
      ...
   }

   public int getTimestamp()
   {
      ...
   }

   public String toString()
   {
      ...
   }
}
```

- `Simulation.java`—this class handles all the details of the simulation. The class starts out as follows.

```
public class Simulation
{
   private static final int MAX_CUSTOMER = 5; //capacity of the line
   private Queue<Customer> q = new Queue<>(MAX_CUSTOMER);
   private int clock;
   private Customer being_served = null;  //the customer being served

   private int finished;  //number of customer finished
   private int nowait;    //number of customers that didn't have to wait
   private int waited;    //number of customers waited
   private int totalwait; //total wait time for customers that had waited in the queue
   ...
}
```

You must make some "natural" methods. No other class can have a queue of customers. No other class can keep track of the simulation information. ONLY THIS CLASS can have a queue and keep track of the simulation information. And you CANNOT have a method that returns the queue itself. **-1 points** for each violation, **up to -5 points**.

You MUST have a **testbed main**() that thoroughly tests the Simulation class. This class CANNOT do I/O, except the testbed main(). **Up to -5 points** if you don't have a testbed main() for this class.

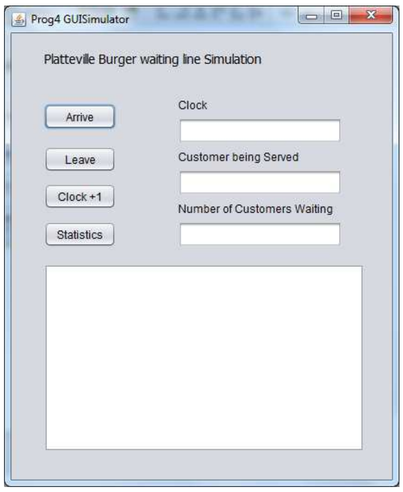
Simulation class must maintain the following information:

- o Average wait time—this is for those customers that had to wait in the queue
- o Total wait time—the sum of all wait time of customers who finished and left the restaurant
- o Number of customers did not have to wait
- o Number of customers finished and left the restaurant

- `ConsoleSimulator.java`—this is the console version of the simulator. You are not allowed to use or access a Queue in this class. **-2 points** if you did. In addition, you CANNOT keep track of the simulation data in this class. **-2 point** if you did.

  In this class, you handle all I/O for the simulation. You must have a **public run()** method to handle input commands. A command consists of a single character followed by an optional parameter. One command per line in the input file **Prog4_1.in.** You MUST check and handle bad commands. The commands are case-sensitive. For output messages, please refer to the Sample Output at the end of this pdf file.
  - o `A` command—customer arrival. If the queue is full, print out an error message. Otherwise, print out a message showing at what time the customer entered the queue, and how full the queue is.
  - o `L` command—customer leave the restaurant. If there is no customers in the restaurant, print out an error message. Otherwise, print out the customer, the time when it finished, and the number of customers waiting in the queue.
  - o `C` *n* command—increase the value of `clock` by *n* time units. If *n* is zero or negative, print an error message.
  - o `S` command—print the current statistics of the simulation. See Sample Output.

- `GUISimulator.java`—this is the GUI version of the simulator. You are not allowed to use or access a Queue in this class. **-2 points** if you did. You MUST create a **JFrame** for this class. And include the following components.
  - o 4 buttons: (1) Arrive; this button is the **A** command. (2) Leave; this button is the **L** command, (3) Clock +1 is the **C** command, but increase 1 time unit every time it is clicked, and (4) Statistics is the **S** command; it displays the statistics of the simulation.
  - o 3 text fields for displaying simulation data: (1) clock, (2) the customer being served, and (3) number of customers waiting. 3 labels: one for each text fields.
  - o One text area and label to display simulation statistics. (NO console output! **-2 points** if you use **System.out** and produce output on the console! A sample GUI is on the right.



8. Submit the following Java files to the grader:
   - Prog4.java
   - Queue.java
   - Customer.java
   - Simulation.java
   - ConsoleSimulator.java
   - GUISimulator.java
9. You must run and generate JavaDoc, and have the resulting HTML document files on the K drive before you come to demo. **-3 points** if you didn't generate HTML Java document.

10. You will get **0 points** if you submit Prog4 after **April 1, @10pm**, OR if you submit Prog4 with differences OR if you DID NOT DEMO. You MUST submit Prog4 with 0 differences to pass this course.
11. You MUST follow the software development ground rules.
12. If you need help from me, place your project in **K:\Courses\CSSE\changl\cs2430\your_login_id**. So I can access it from my office when you stop by.

Sample Input

```
S
C 3
L
A
C 6
A
C 1
A
L
S
C -10
L
A
C 2
A
Blah blah blah
A
C 1
A
S
C 3
A
C 5
A
L
S
C 3
L
L
C 1
L
L
C 3
L
C 2
L
S
Q
U
A
C
K
```

Sample Output

```
The average wait time for the customers who finished waiting: 0.0.
The total wait time is 0.
The number of customers finished: 0.
The number of customers who did not have to wait: 0.

Time updated by 3 units; current time is 3.
Nobody is being served @time 3.
A customer has arrived @time 3. Number of customers waiting in the line: 0
Time updated by 6 units; current time is 9.
```

```
A customer has arrived @time 9. Number of customers waiting in the line: 1
Time updated by 1 unit; current time is 10.
A customer has arrived @time 10. Number of customers waiting in the line: 2
Customer#1 arrived @time 3 finished @time 10. Number of customers waiting: 1.

The average wait time for the customers who finished waiting: 1.0.
The total wait time is 1.
The number of customers finished: 1.
The number of customers who did not have to wait: 1.

Time NOT updated with -10.
Customer#2 arrived @time 9 finished @time 10. Number of customers waiting: 0.
A customer has arrived @time 10. Number of customers waiting in the line: 1
Time updated by 2 units; current time is 12.
A customer has arrived @time 12. Number of customers waiting in the line: 2
Blah is NOT a valid command!
A customer has arrived @time 12. Number of customers waiting in the line: 3
Time updated by 1 unit; current time is 13.
A customer has arrived @time 13. Number of customers waiting in the line: 4

The average wait time for the customers who finished waiting: 0.5.
The total wait time is 1.
The number of customers finished: 2.
The number of customers who did not have to wait: 1.

Time updated by 3 units; current time is 16.
A customer has arrived @time 16. Number of customers waiting in the line: 5
Time updated by 5 units; current time is 21.
A customer couln't get in the line @time 21 because the line was full.
Customer#3 arrived @time 10 finished @time 21. Number of customers waiting: 4.

The average wait time for the customers who finished waiting: 4.0.
The total wait time is 12.
The number of customers finished: 3.
The number of customers who did not have to wait: 1.

Time updated by 3 units; current time is 24.
Customer#4 arrived @time 10 finished @time 24. Number of customers waiting: 3.
Customer#5 arrived @time 12 finished @time 24. Number of customers waiting: 2.
Time updated by 1 unit; current time is 25.
Customer#6 arrived @time 12 finished @time 25. Number of customers waiting: 1.
Customer#7 arrived @time 13 finished @time 25. Number of customers waiting: 0.
Time updated by 3 units; current time is 28.
Customer#8 arrived @time 16 finished @time 28. Number of customers waiting: 0.
Time updated by 2 units; current time is 30.
Nobody is being served @time 30.

The average wait time for the customers who finished waiting: 8.142857142857142.
The total wait time is 57.
The number of customers finished: 8.
The number of customers who did not have to wait: 1.

Simulation statistics:
The average wait time for the customers who finished waiting: 8.142857142857142.
The total wait time is 57.
The number of customers finished: 8.
The number of customers who did not have to wait: 1.

Simulation terminated.
```