ITP4510 Data Structure and Algorithms

ASSIGNMENT
Deadline: (refer to Moodle)

Membership Counter Simulation

Introduction

Prime Food Limited is a chain retail store selling dried marine products mainly. The company is currently having 8 retail stores around Hong Kong and employing a total of 80 full time employees and a number of part time employees.
To retain customers, they launched a membership system and let members to save their purchase records as marks in the system, which allows them to exchange small gifts later. The management would like to reserve a special counter for members to interact with their staffs. You are required to write a program to simulate the counter queuing status. There will be only one membership counter in the retail store. Each member (customer) who would like to interact with the staff will have to queue up in front of the counter.

The interaction and estimated time for different tasks are as follows:

- New membership application – 4 mins
- Exchange small gifts – 2 mins
- Check marks balance – 1 min
- Transfer marks balance from member to member – 3 mins

Program Structure

First, the program simulates the ordering situation in initial status (at time 0). Then, during the simulation there is a basic program loop that each of the iteration (cycle) represents one minute. During each cycle, members may arrive at the counter.

Pseudocode:

Simulate the situation at time 0 (initial status)

While (currentTime < TIME_NEEDED_FOR_SIMULATION) {

1. Simulate at beginning of each iteration
   - If a member arrived, he/she can request one out of the four services, then the task is put in the queue
   - More than one member can arrive the counter during one iteration

2. Pass one minute
   - currentTime + 1

3. Simulate action
   - If there is/are task(s) in the queue, retrieve the first task from the queue. The remained processing time of this task minus one.
   - When the remained time to process is zero, this task can be removed from queue.

4. Show the task queue status

}

After end of simulation, a summary will be printed

- Total minutes of the simulation.
- Total number of the members have served.
- Total number of tasks have been completed during the simulation.
- Total outstanding tasks remained at the end of simulation.

Execution Sample

Here is a sample of one simulation.

```
Input simulation time (mins):5
---------------- START OF SIMULATION -----------------
Action [1-New Mem, 2-Gift, 3-Check Bal, 4-Trans Bal]2
Action [1-New Mem, 2-Gift, 3-Check Bal, 4-Trans Bal]3
Action [1-New Mem, 2-Gift, 3-Check Bal, 4-Trans Bal]0
After 1 min(s): [ (1 gift) (1 check bal) ]
-----------------------------------------------------
Action [1-New Mem, 2-Gift, 3-Check Bal, 4-Trans Bal]4
Action [1-New Mem, 2-Gift, 3-Check Bal, 4-Trans Bal]0
After 2 min(s): [ (1 check bal) (3 trans bal) ]
-----------------------------------------------------
Action [1-New Mem, 2-Gift, 3-Check Bal, 4-Trans Bal]0
After 3 min(s): [ (3 trans bal) ]
-----------------------------------------------------
Action [1-New Mem, 2-Gift, 3-Check Bal, 4-Trans Bal]1
Action [1-New Mem, 2-Gift, 3-Check Bal, 4-Trans Bal]3
Action [1-New Mem, 2-Gift, 3-Check Bal, 4-Trans Bal]0
After 4 min(s): [ (2 trans bal) (4 new member) (1 check bal) ]
-----------------------------------------------------
Action [1-New Mem, 2-Gift, 3-Check Bal, 4-Trans Bal]0
After 5 min(s): [ (1 trans bal) (4 new member) (1 check bal) ]
-----------------------------------------------------
----------------- END OF SIMULATION ------------------
Total mins simulated: 5 minutes
Number of members served: 2
Action queued during the simulation: 5
Outstanding item at the end of simulation: 3
```

Random Seed Function

When user input zero value for the simulation length at initial stage, it means the program will generate input value randomly. In addition, user need to input the random seed value for the simulation process. The example is shown in Appendix I. **The simulation length is fixed to 10 minutes.**

Task Specification

1. You should use the **LinkedQueue** and relevant classes correctly.
2. Implement the program using Java.
3. Handle exceptional/abnormal cases. You may create your own Exception class.

Instructions to Students

This assignment is an individual assignment.

You are required to submit:

- Well-documented program listings and **the executable results (screen dumps).**
- Upload your files to Moodle including all your programs and report. It is required that your programs can be successfully compile.

Deliverables

1. All program source code (.java files)
2. Test report

Marking

The marking of this assignment is based on (but not limited to) the followings:

1. Successful compilation of program source code
2. Correct use of data structures
3. Functions:
    - Place task correctly
    - Simulation of time correctly
    - Task being removed from queue correctly
    - Simulation summary
    - Random seed feature
    - Handle incorrect user input
4. Program structure and coding style
5. Test report presentation

Any plagiarism found will result in zero marks. All assignments that have been found involved wholly or partly in plagiarism (no matter these assignments are from the original authors or from the plagiarists) will score ZERO marks. Further, disciplinary action will be followed.

Appendix I

Example of simulating in random data using seed number **4444**:

```
Input simulation time (mins):0
Input seed number:4444
---------------- START OF SIMULATION -----------------
Action [1-New Mem, 2-Gift, 3-Check Bal, 4-Trans Bal]1
Action [1-New Mem, 2-Gift, 3-Check Bal, 4-Trans Bal]1
Action [1-New Mem, 2-Gift, 3-Check Bal, 4-Trans Bal]4
Action [1-New Mem, 2-Gift, 3-Check Bal, 4-Trans Bal]0
After 1 min(s): [ (3 new member) (4 new member) (3 trans bal) ]
-----------------------------------------------------
Action [1-New Mem, 2-Gift, 3-Check Bal, 4-Trans Bal]0
After 2 min(s): [ (2 new member) (4 new member) (3 trans bal) ]
-----------------------------------------------------
Action [1-New Mem, 2-Gift, 3-Check Bal, 4-Trans Bal]0
After 3 min(s): [ (1 new member) (4 new member) (3 trans bal) ]
-----------------------------------------------------
Action [1-New Mem, 2-Gift, 3-Check Bal, 4-Trans Bal]0
After 4 min(s): [ (4 new member) (3 trans bal) ]
-----------------------------------------------------
Action [1-New Mem, 2-Gift, 3-Check Bal, 4-Trans Bal]2
Action [1-New Mem, 2-Gift, 3-Check Bal, 4-Trans Bal]0
After 5 min(s): [ (3 new member) (3 trans bal) (2 gift) ]
-----------------------------------------------------
Action [1-New Mem, 2-Gift, 3-Check Bal, 4-Trans Bal]0
After 6 min(s): [ (2 new member) (3 trans bal) (2 gift) ]
-----------------------------------------------------
Action [1-New Mem, 2-Gift, 3-Check Bal, 4-Trans Bal]4
Action [1-New Mem, 2-Gift, 3-Check Bal, 4-Trans Bal]3
Action [1-New Mem, 2-Gift, 3-Check Bal, 4-Trans Bal]1
Action [1-New Mem, 2-Gift, 3-Check Bal, 4-Trans Bal]3
Action [1-New Mem, 2-Gift, 3-Check Bal, 4-Trans Bal]0
After 7 min(s): [ (1 new member) (3 trans bal) (2 gift) (3 trans bal) (1 check bal) (4 new member)
(1 check bal) ]
-----------------------------------------------------
Action [1-New Mem, 2-Gift, 3-Check Bal, 4-Trans Bal]0
After 8 min(s): [ (3 trans bal) (2 gift) (3 trans bal) (1 check bal) (4 new member) (1 check
bal) ]
-----------------------------------------------------
Action [1-New Mem, 2-Gift, 3-Check Bal, 4-Trans Bal]1
Action [1-New Mem, 2-Gift, 3-Check Bal, 4-Trans Bal]0
After 9 min(s): [ (2 trans bal) (2 gift) (3 trans bal) (1 check bal) (4 new member) (1 check bal)
(4 new member) ]
-----------------------------------------------------
Action [1-New Mem, 2-Gift, 3-Check Bal, 4-Trans Bal]2
Action [1-New Mem, 2-Gift, 3-Check Bal, 4-Trans Bal]1
Action [1-New Mem, 2-Gift, 3-Check Bal, 4-Trans Bal]4
Action [1-New Mem, 2-Gift, 3-Check Bal, 4-Trans Bal]0
After 10 min(s): [ (1 trans bal) (2 gift) (3 trans bal) (1 check bal) (4 new member) (1 check bal)
(4 new member) (2 gift) (4 new member) (3 trans bal) ]
-----------------------------------------------------
----------------- END OF SIMULATION ------------------
Total mins simulated: 10 minutes
Number of members served: 2
Action queued during the simulation: 12
Outstanding item at the end of simulation: 10
BUILD SUCCESSFUL (total time: 3 seconds)
```

Appendix II

Example of using class **Random**.

```
import java.util.*;

public class RandomSeed {
        public static void main(String[] args) {
                long seed = 9999;
                Random rnd = new Random();
                rnd.setSeed(seed);

                //generate number from 0 to 10
                for (int i=0; i<20; i++) {
                        System.out.println(""+i+": "+ rnd.nextInt(11));
                }
        }
}
```