COMP 2011, Assignment #1 **DUE ON: October 12, 2020 11:59pm**

Grader: KE Yuping (yuping.ke@connect.polyu.hk)

## Notice

- Make sure your codes can be compiled by command line. If you've problems, this and this web pages may help, and you're welcome to ask the graders.
  No marks shall be awarded for programs that do not compile.
- Your submission should contain a .java source file and a screenshot of its execution. See this website for how to take a screenshot.
- Submission procedure (deviations will result in deduction of up to 20 points):
  1) Create a folder with name `<student_no>_<name>`, e.g., 12345678d_CHANTaiMan
  2) Name the .java file as `A1_<student_no>_<name>_<class_name>.java`, e.g., `A1_12345678d_CHANTaiMan_Elizebath.java`
     **But please don't change the class name in the Java file.**
  3) Put all your files into this folder, including java files (only .java files in the `src` folder, not the entire project folder) and screenshots.
  4) Compress this folder as a .zip or .jar file, and submit the compressed file to learn@polyu. (Similar as the distributed codes.)
  5) An example file is provided for your reference.
- You are not allowed to modify the signatures of the methods you are asked to implement. On the other hand, you can add new methods, and add variables to the classes and methods freely.

1. (40 points) An urgent level ("Urgent", "Semi-Urgent", and "Non-Urgent") is assigned to each patient after pre-evaluation. The next patient to be seen by a doctor is the one with the highest urgent level, and if there are more than one patient of the highest urgent level, then first come first serve.

   Implement class `Elizebath` in two ways. In the first, `enter` uses $O(1)$ time; in the second, `leave` uses $O(1)$ time.

   Analyze the running time of `enter` and `leave` in both your implementations.

2. (40 points) This question is about the class `CustomerQueue`, which represents a queue of customers waiting as a line at a cashier. It uses the class `LinkedList` of lecture 4 as the internal storage. (Hint: This problem cannot be solved with the current version of `LinkedList.java`, so you have to augment it.)

   - Implement methods `enqueue`, `dequeue`, and `isEmpty` in $O(1)$ time.
   - Implement the `split` method.

   It splits the queue into $k$ queues of almost equal sizes, such that the first, $(k+1)$st, ... customers goes to the first queue, the second, $(k+2)$nd, ... customers goes to the second queue, and so on. It should run in $O(n)$ time, where $n$ is the number of customers waiting in the queue, and use $O(1)$ extra space (not counting the list itself). For example,

Eason, Denise, Jennifer, Joey, Kay, Cheung, Winnie, Mickey, Teddy, and Peppa

are waiting to pay at the only cashier of the cafeteria. One more cashier starts working, and then the queue will immediately dissolve into two queues:

Eason, Jennifer, Kay, Winnie, and Teddy.

Denise, Joey, Cheung, Mickey, and Peppa.

3. (20 points) Find the last node of a linked list of $n$ elements whose index is a multiple of $k$ (counted from 0). For example, if the list is

$$12 \to 75 \to 37 \to 99 \to 12 \to 38 \to 99 \to 60 \downarrow$$

and $k = 4$, then you should return the second 12 (with index 4). Your algorithm should take $O(n)$ time and use $O(1)$ extra space. Implement the following method in `LinkedList.java`.

```
public T lastK(int k)
```

## Challenging question

You're suggested to try this to get bonus points and better understanding.

1. Given an array of $n$ integers, you are asked to give an algorithm to find the second smallest of them with at most $n + \lceil \log n \rceil - 2$ comparisons. Note that this requirement is stronger than linear-time: We're not using the Big-Oh notation, so $2n$ comparisons wouldn't count.

   Hint: divide and conquer to find the smallest; where is the second smallest?