Assignment 1

Due Date: March 6th, 2019
(No late submissions will be accepted)

Contact: TA Seunghwan Lee (shlee95@postech.ac.kr)

General Instructions

Each assignment has a written part and a programming part. For a written part, please write your answers in a pdf file, and for a programming part, follow the instructions below:

- Write your code in submission.cpp
- TA will test your code with Visual Studio on Windows OS, so please write your code in the same environment.
- Obviously, you must NOT use a library like the Standard Template Library (STL)
- Submit only C ++ files, not the entire project
- You should modify the code in submission.cpp between

  ```
  /* BEGIN_YOUR_CODE */
  ```

  and

  ```
  /* END_YOUR_CODE */
  ```

You can add other helper functions outside this block if you want.

Written Problems

Do the following problems in the textbook and note that you need to show your work (i.e., not just the answer) for exercises.

Problem 1 [2 points]

Do the exercise *R-3.3* in the textbook.

Problem 2 [3 points]

Do the exercise *C-3.11* in the textbook.

Problem 3 [2 points]

Do the exercise C-*3.12* in the textbook.

Problem 4 [2 points]

Do the exercise C-*3.16* in the textbook.

Problem 5 [3 points]

Do the exercise C-*3.22* in the textbook.

Programming Problems

Problem 1. Josephus problem

Josephus problem is a math puzzle with a grim description. n prisoners are standing on a circle, sequentially numbered from 1 to n. An executioner walks along the circle, starting from prisoner 1, removing every k-th prisoner and killing him. As the process goes on, the circle becomes smaller and smaller, until only one prisoner remains, who is then freed. For example, if $n = 5$ and $k = 2$, then the safe position is 3. Firstly, the person at position 2 is killed, then person at position 4 is killed, then person at position 1 is killed. Finally, the person at position 5 is killed. so the person at position 3 survives. If $n = 7$ and $k = 3$, then the safe position is 4. The persons at positions 3, 6, 2, 7, 5, 1 are killed in order, and person at position 4 survives.

Problem 1a [3 points]

Implement a circularly linked list to manage the circle.

Problem 1b [2 points]

Return the safe position successfully from the *Solve* function in submission.cpp.