Homework 5

Refer to the documenting and submitting homework from here.

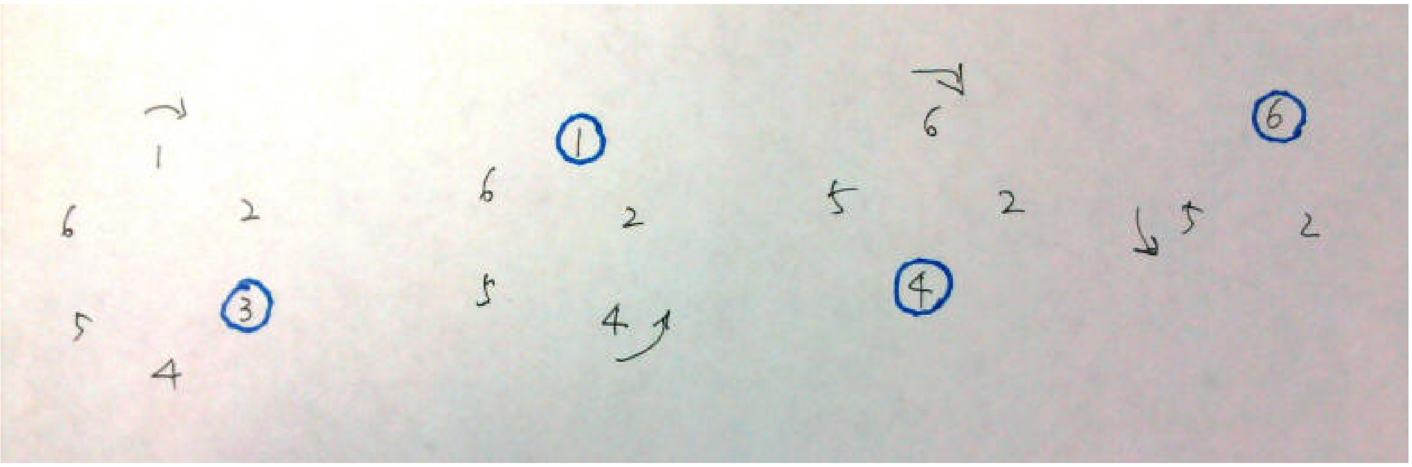
Using the node1.h node1.cpp, (refer to the new bag and sequence examples in the unit 6 and Section 5.4 in the textbook to get an idea how the node is utilized)

Modify the node class to implement a circular doubly linked list.

Solve the circular drawing problem described below using the circular doubly linked list.

A travel agent selects n customers to compete in the finals of a contest for a free world cruise. The agent places the customer in a circle and then draws a number m(1<= m <= n) from a hat. The game is played by having the agent walk **clockwise and counter-clockwise** around a circle, stopping at every mth contestant. **At an odd turn, the agent walk clockwise, and at an even turn the agent walk counter clockwise, alternatively.** The agent asks the selected person to leave the game and then continues the walk. Over time, the number of remaining contestants dwindles until only one survivor remains. The person is the winner of the world cruise.

Following figure illustrates the problem for n=6 and m=3.

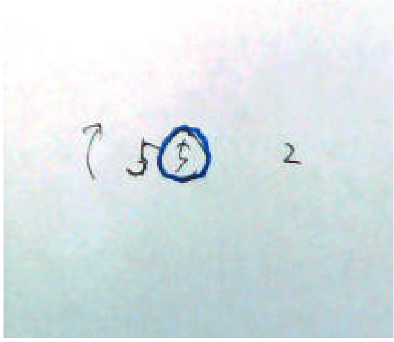


start at 1 forward
visit 1,2,3 remove 3

start at 4 backward
visit 4,2,1, remove 1

start at 6 forward
visit 6, 2, 4, remove 4

start at 5 backward
visit 5,2,6, remove 6



start at 5 forward
visit 5,2,5, remove 5

The winner is 2

The program assumes that n contestants compete for the cruise and that the deselection process removes every mth contestant. The value n and m are function arguments and provided by the user. A circular doubly linked list stores the contestants with integer values 1, 2, 3, ...n. In an iterative process, the algorithm sequences through the ring of remaining nodes and removes the mth node from the list. Because there are n contestants for the cruise, the function uses n-1 iterations. The process starts by assigning a node pointer to reference the first node. Starting with this node, a series of m-1 steps moves the pointer forward to the location of the node to remove. To avoid invalidating the pointer when we delete the node, we first move the pointer forward one position and then delete the predecessor node. The next iteration picks up at the current pointer position and repeats the process of moving the pointer forward m-1 steps and deleting a node. At an even turn, all the previous processes are same except that the forward movements are substituted by backward movements. At the end of the n-1 iterations, the pointer identifies the winner of the cruise.

Of course, you can get an answer without using a circular doubly linked list. For example, you can somehow store the predecessor node pointer every time you advance/reverse to the next node.

However, in this homework, you need to implement it using a circular and doubly linked list.

You need to develop a short main function that takes the user's n and m values (n>m). At each iteration, you should print the deselected contestant. The function terminates by displaying the winner.

Grading guideline

- Correct and complete functionality 70%
- Reasonable fault tolerance, i.e. it should not crash often. 10%
- Makefile and Readme 10%
- Documentation and comments in the code 10%

Due on Feb 29th at 4PM.