Circular Linked List

Goals

- Implement a circular linked list data type
- Learn how to modify data in a linked list

In this lab, we will create a **queue** data structure utilizing circular linked list.

Requirements

For simplicity, out circular linked list will only contain integer as values.

Note: You are not allowed to use any standard containers for this lab.

QueueNode Data Structures

Create a struct that is called **QueueNode** that represents each node. It contains:

- **next**, a pointer to the next Node object
- **prev**, a pointer to the previous Node object
- **val**, integer value the specific Node contains

Queue class

Queue class has the following member variables:

- **head,** a pointer to the **first QueueNode** object in the queue

Note: Constructor and destructor are needed for the Queue class, and destructor should free all the memories of nodes in the queue.

Note: You cannot have a variable to keep track of the size of the queue. (or the number of nodes the queue holds)

Queue class also has the following member functions:

- **bool isEmpty()** checks if the queue is empty. If so, returns true; otherwise, return false.
- **void addBack(int val)** takes a user-inputted integer, creates a QueueNode with user-inputted integer, and appends it to the back of the list.
- **int getFront()** returns the value of the node at the front of the queue.

- **void removeFront()** removes the front QueueNode of the queue and free the memory.
- **void printQueue()** traverses through the queue from head using **next** pointers, and prints the values of each QueueNode in the queue.

Note: Make sure the user-inputted integer is **validated in menu**, before it is passed into addBack(int).

Menu

The menu should provide the following options:

1. Add a value to the back of queue
2. Display the front value
3. Remove the front node
4. Display the queue’s content
5. Exit

For option #1, the menu should prompt user for an integer input to be added to the queue. The input must be validated before being passed into the addBack(int) function.

For option #2, #3, #4, what happens when the queue is empty? What function can help you out in this situation?

Note: It is important to know that after each function operation, you need to reassign the prev, next pointers for the QueueNodes inside the queue, so that linked list structure is preserved.

HINT: It is very important to sit down, design the whole program and understand the steps of each operation before you start coding. Think about edge cases carefully (when the queue is empty, or only have 1, or 2 nodes).

Important:

1. For this lab, you cannot implement the linked list operations using functions from outside source. You need to write all the functions by yourself!
2. If you do not use a circular linked list structure for queue, your lab will not be graded.

Example run of the program

Welcome to my queue!

Choose from following options:

1. Enter a value to be added to the back of queue
2. Display first node (front) value
3. Remove first node (front) value
4. Display the queue contents
5. Exit

1

Please enter a positive integer:

15

Choose from following options:

1. Enter a value to be added to the back of queue
2. Display first node (front) value
3. Remove first node (front) value
4. Display the queue contents
5. Exit

1

Please enter a positive integer:

5

Choose from following options:

1. Enter a value to be added to the back of queue
2. Display first node (front) value
3. Remove first node (front) value
4. Display the queue contents
5. Exit

1

Please enter a positive integer:

10

Choose from following options:

1. Enter a value to be added to the back of queue
2. Display first node (front) value
3. Remove first node (front) value
4. Display the queue contents
5. Exit

3

Choose from following options:

1. Enter a value to be added to the back of queue
2. Display first node (front) value
3. Remove first node (front) value
4. Display the queue contents
5. Exit

2

The first node value is: 5

Choose from following options:

1. Enter a value to be added to the back of queue
2. Display first node (front) value
3. Remove first node (front) value
4. Display the queue contents
5. Exit

4

Your queue is: 5 10

What you need to submit

- All the program files including header and source files (.cpp/.hpp)
- Makefile

Important: Put all the files in a single .zip file and submit it on Canvas.

Grading

- Programming style: – 10%
- Header file (QueueNode struct with only the data and the two pointer members & Queue class declarations): – 10%
- Queue class:
    - Necessary constructors/destructors: – 10%
    - No data members other than the head pointer: – 5%
    - Implement addBack():– 15%
    - Implement removeFront(): – 15%
    - Implement getFront(): – 10%
    - Implement printQueue(): – 10%
- menu function to test your queue: – 15%