Kripash Shrestha

04/27/2017

Project 7 Documentation

CS 202 Spring 2017

The objective of this project was to create Array and Node based queues with the use to template classes. Queue's work on a First in First Out basis, so the first data inside the structure would be the first one to be removed when trying to add new data. However when trying to access a queue, you may only access the first and last elements of the queue.

Before I had read the full instructions on popping the array queue, I did array manipulation through which the first element of the queue would now be the second element and then iterating through that way. So m_array[i] = m_array[i+1]; Then I would manipulate m_back to be equal to the last m_array and decrement size. But after reading the instructions, I realized the easiest way to do this would be to just change m_front. The way I worked it was that, the m_front would always be what it starts off as. However, both ways do technically work. Inserting was as easy for the array queue, we just had to make sure that the array was not full, since there is a max size to the array, because we are not dynamically allocating memory to insert data into the array.

The overloaded << operator was confusing at the beginning because I was not sure if we were supposed to keep the data from the original queue or not. So to be on the safe side and after Professor PapaChristos, I made a copy of the object and then continuously popped the object and printed the copied object's m_front every time it was popped, to print the data inside of the queue. This way, the data inside the original queue is still there. Professor PapaChristos mentioned that either way was okay. That confusion was also why I had get m_next function

with my node class but then after creating a copy of the object that was to be printed, there was no need for that.

Templating the classes were not as difficult as expected. However, when I was first doing so, I realized that I was missing two critical lines for the entire thing to work. I was missing template <class T> class Node; and template <class T> class NodeQueue respective to each other to make them work. I was not aware of this until I went into the lecture slides. I was thinking of trying to template the Node class into the NodeQueue class, so I am not sure where that would have led to. Also, I had troubles with the overloaded << templating. I tried to do it the hard way at first and then I took the “easy way” as said by Professor PapaChristos and just template it into the class declaration without using scope resolution operator. One thing to keep in mind for my << overloaded function is that, every time you call it, it will print the value of the object’s m_front and m_back, its size and the data inside the object, if it is not empty. If the queue is empty, then there will be a warning issued to the terminal.