CS 260
Programming Lab 2

This exercise builds upon the simple queue created in Part A to create a double ended queue.

This lab is worth a total of 100 points, 10 points are the self-evaluation, 50 points for the basic lab as specified below, 30 points for the advanced version of the lab, and 10 points for a solution to the thinking problem.

Base Lab Specification

For the base lab, you will implement a single ended queue on an array. Create a class Deque that has the following methods.

- Constructor (Python init) creates an array of **n** integer where **n** is the size passed in. If no size is passed in, or if the size is <1, create an array of 20 items (default constructor). ***Do not initialize the elements of the array unless you are using Python for the assignment. For Python only, the array must be initialized to zero.***
- void addTail(int value) – adds a new value to the tail of the queue, wrapping if necessary. If there is no room to add a new value, call the resize() method.
- int removeHead() – save the value at the head of the queue, update it to remove access to that value, and return the saved value, wrapping if necessary. If the queue is empty throw an exception. (C++ use ***out_of_range*** , C# use ***IndexOutOfRangeException***, Python use ***IndexError***).
- string dumpArray() – return an array containing the contents of the array from index 0 to size-1. This is for debugging purposes and to verify that wrapping is working properly.
- resize() – creates a new array twice as large and copies the elements to it. This should properly deal with situations where there has been wrapping and the tail is at an index less than the head. Note, this is not a simple exercise where you can copy index 0 old array to index 0 new array and so forth. You will need to end up after copying with the head at index 0 and tail appropriately located. There are several ways to accomplish this.
- string listQueue() – that returns a string listing the elements from head to tail. This should properly deal with situations where there has been wrapping and the tail is at an index less than the head.
- bool isEmpty() – returns true when the double ended queue is empty, false otherwise.

Advanced Lab Specification

For the advanced lab, start with your base lab class and add the following methods.

1) void addHead(int value) – adds a new value at the head. Wrapping if necessary. This should not overwrite data that was previously in the queue. If the array is full, call resize() to double the array and copy the data as needed.

2) int removeTail() – saves the value at the tail of the queue, updating the queue to remove access to the item at the tail, and returns the saved value. Wrapping if necessary. If the queue is empty, throw an exception. (C++ use ***out_of_range*** , C# use ***IndexOutOfRangeException***, Python use ***IndexError***).

Thinking problem

How could you implement a stack using your double ended queue? Use this concept to list a provided set of numbers in reverse order in the appropriate location in the driver.

Development and Testing

There is a driver provided in Moodle for this lab. It has multiple sections; your class needs to pass all of them to get full credit.

Complete each section of testing before you begin working on implementing the next features in your class.

There is a place in the driver for you to add code for the thinking part of the lab.

For this lab, there are explanations of how such a double ended queue would work, but there are no details, you need to work it out on your own. Suggestion, draw pictures of the queue and work through the problem with pseudocode before you start actually coding it.