Assignment 1 Instructions

Relevant Textbook Reading

1. Chapter 3, except linked lists (3.2.2, 3.5) and iterators (e.g., 3.3.1, 3.3.2, 3.3.3).
2. Chapter 1, Section 1.4; Section 1.6 up to end of 1.6.2; Section 1.6.

General

A deque is an ADT that generalizes a queue, allowing insertions and deletions at both ends. The C++ STL (standard template library) has a list class which provides this capability, using a linked list implementation. (It is described in Chapter 3, but is not required reading at this time.) For this assignment, you are to complete a simple implementation of an array-based deque. The implementation uses the "circular array" scheme, as in the array-based queue implementation we sketched in class. In the queue implementation, the front and back indexes into the array wrap around when they reach the "upper" end of the array. For the deque, you need look after wrapping around at both ends of the array.

Files Provided

A collection of files is provided to help you get started. The most important is the file Deque.h, which contains a partial implementation that you are to complete. The files are:

- `Vector.h` The textbook's implementation of Vector class.
- `dsexceptions.h` A file that must be included to compile any of the texbook code
- `Vest.cpp` A simple test program for the Vector class.
- `Ivector.h` An minimal implemation of a Vector class that only stores ints.
- `Itest.cpp` A basic test program for the Ivector class.
- `Tvector.h` A templated version of the same minimal vector class implementation.
- `Ttest.cpp` A basic test program for the Tvector class.
- `Stack.h` A simple stack class, based on the same code.
- `Stest.cpp` A simple test program for the Stack class.
- `Deque.h` A partial implementation of the deque class.
- `Makefile` A make file to compile all the test programs (and some others)

Vector.h is included to illustrate the complexity of the code provided with the book. (In C++, a Vector is essentially a growable array.) Ivector.h and Tvector.h illustrate minimal versions of a similar class, and Stack.h illustrated using the same code base to make a (sort of) different ADT.

For you to do

There are three parts to the assignment.

1. Complete the implementation of the Deque class. If you don't have C++ experience, I recommend as a first step completing it just as a queue, and then adding the other operations only after you have the basic circular-array queue implemention completely correct.
2. Design and implement a test program that demonstrates the correctness of an implementation of the deque class. Obviously, you should be writing various small test programs as you develop your deque implementation, but at the end you should be thinking of a suite of tests that will convince even a skeptic of the correctness. You must write up a short description and explanation of your design.
3. Extend the implementation with declarations of the [] operator, allowing array-like access to all elements in the deque. If D is a deque, then D[i] should return the i-th element in the deque (not the i-th element in the array), and D[i]=object should change the i-th element to be object. Write a small program that demonstrates correctness.

Submission and Grading

Submit a .zip file containing your code and write-up to Coursys. A grading breakdown will be provided shortly.