ECE250: Lab Project 1

Due Date: January 31st 2020, 11:00 pm

1. Project Description

The goal of this project is to design and implement a dynamic deque data structure. A deque is a queue that stores elements in an ordered list and allows insertions and deletions at both ends in O(1) time, and has no limitations in its size. This means a deque can dynamically grow in size. You have to implement a deque using a linked list data structure. In order to simplify the implementation of a deque for this project, we will assume that the deque can store only elements of type *int*.

Design your solution using the object-oriented paradigm. You will create a design in which a deque is represented by a class that: (i) stores properties (data members) and (ii) provides services (function members).

Do not use classes from the C++ STL library for this project. Instead, we ask that you write your own linked list class, and use it in your implementation.

2. Program Design

Write a short description of your design. You will submit this document along with your C++ solution files for marking. This document must include your design decisions. Please refer to the course website for "Programming Guidelines" and the expected content for your design document.

3. Input output requirements

Write a test program named **dequedriver.cpp** that reads commands from standard input and writes the output to standard output. This program will respond to the commands described in this section.

Command	**Parameters**	**Description**	**Output**
enqueue_front	*i* *i is an integer*	Add element at the front	success
enqueue_back	*i* *i is an integer*	Add element at the end	success
dequeue_front		Remove element from the front	success or failure*
dequeue_back		Remove element from the end	success or failure*
clear		Clears the content of the deque if the queue is not already empty	success

Command	Parameters	Description	Output
front	*i* *i is the expected element at the front of deque*	Access the first element and compare to *i,* if equal, return success, else failure	success or failure*
back	*i* *i is the expected element at the end of deque*	Access the last element and compare to *i*, if equal, return success, else failure	success or failure*
empty		Test if deque is empty	success or failure success if empty failure if not empty
size		Return the size of the queue	size is *s*
print		Print all the values in the deque two times: • from front to back • from back to front **Note:** For an empty deque, print nothing	An example: 0 1 2 3 … 9 9 8 7 6 … 0

* Failure must be detected by handling an exception. That is, the function will "throw" an exception and the driver program will "catch" the exception, creating the output "failure" for these cases. You can define an exception called "deque_empty" to be used on these cases.

4. Test Files

The course web site contains example input files with the corresponding output. The files are named test01.in, test02.in and so on with their corresponding output files named test01.out, test02.out and so on.

3. Performance

All operations must be implemented in O(1) times, except *clear* and *print* operations which have linear time complexity (O(n)).

4. How to Submit Your Program

Once you have completed your solution and tested it comprehensively in your computer or on the lab computers, you have to transfer your files to the *eceUbuntu server* and test there

since we perform the automated testing using this environment.

Once you finish testing in the *eceUbuntu server*, you will create a compressed file (tar.gz) that should contain:

- A typed document (maximum two pages) describing your design. A document beyond 2 pages will not be marked. Submit this document in PDF format. The name of this file should be:

  xxxxxxxx _design_ p***n***.pdf

  where ***xxxxxxxx*** is your UW user id (e.g., jsmith) and ***n*** is the project number that is 1 (one) for this submission.

- A test program (dequedriver.cpp*)* containing your *main()* function
- Required header files and classes (ending in *.h .cpp*)
- A make file (named Makefile), with instructions to compile your solution and create an executable named **dequedriver**

The name of your compressed file should be ***xxxxxxxx***_p***n***.tar.gz, where ***xxxxxxxx*** is your UW user id (e.g., jsmith) and ***n*** is the project number that is 1 (one) for this submission.