Data Structures and Algorithms
LAB #5 - Queues

Fall 2018

Objectives

After this lab, the student should be able to:

- Use class templates to implement queue (array-based or linked list-based).
- Pass and return queue (template) to/from functions
- Write code to use queues in some common applications.

Code Examples

- Open "Lab5.sln"
    1- Run project "***1-Queues***" and see how Queue is implemented as class template using linked lists.
    2- Run project "***2-Passing_Queues***" and see how Queue can be passed to a function.

Practice Exercises

Exercise 1

The given code example shows how to implement the Queue using linked lists
You are required to implement the Queue the using arrays.
What is the problem with array implementation?
What is circular queue? Implement it using array-based implementation

Note:

In the following exercises you are allowed to use only queues or stacks as auxiliary memory if needed.

Exercise 2

Given a queue of sorted integers, write a C++ program that deletes duplicates from the list **without changing the order of the remaining elements.** The original queue should be modified to contain only non-duplicates

Exercise 3

Write a function ***catQueues*** that takes two queues and concatenates them together. The second queue should be placed at the end of the first queue.

Exercise 4

Write a function called ***queueToStack*** that creates a stack given a queue. At the end of the algorithm, the queue should be ***unchanged***; the front of the queue should be the bottom of the stack, and the rear of the queue should be the top of the stack

Exercise 5

Write a function ***compressString*** that takes a string stored in a queue and compresses it by deleting all space characters from the string.

Exercise 6

Write a function **reverseQueue** that reverses the contents of a queue.

Exercise 7

Write a function **isIdentical** that checks the contents of two queues and returns true if they are identical and false if they are not.