Data Structures (2028C) -- Spring 2025 – Lab 6

Topics covered: Stacks and Queues

Lab due: ***Sunday, Mar 2 at 11:55PM for Monday Section***
Tuesday, Mar 4 at 11:55 PM for Wednesday Section

Objective:

The objective of this Lab is to examine stacks and queues built using C++. This will reverse the letters in words but not the order of the words.

Task 1: Create a stack class that will be used as the basis for the remainder of the lab.

1. Create a new project. You can name this whatever you like.
2. Design a stack class using an array (do not use a vector). This class should be a template.
   a. The constructor should include a parameter to indicate the size of the array.
   b. The array holding the data should be an array of pointers.
   c. The push function should accept a pointer and add that pointer to the top of the stack. It doesn't need to create any memory.
   d. The pop function should return a pointer. It doesn't need to delete any memory.
   e. The top function should return a pointer to the item on the top of the stack without removing it.
   f. The length function will return an int indicating the number of items in the stack.
   g. The empty function should empty the stack of all contents. It needs to call delete to avoid memory leaks. It doesn't need to return any value.
   h. Create the implementation code for the above functions as required. **Do not** use cout in the class. Any errors such as overflow or underflow should throw a custom class error.

Task 2: Create a queue class that will be used as the basis for the remainder of the lab.

1. Create a queue class. This class should be a template.
2. Design the queue class using an array to store data. Include the standard functions for a queue. This should be very similar to the functionality described in Task 1.

Task 3: Working with text

1. Give the user the choice to get input from a file or from the terminal.
2. If user chooses a file:
   a. Ask the user to provide the name of a file. If the file doesn't exist, notify the user and prompt for a file again.
   b. Write code to read a series of strings from the above file, reverse the order of the text in each word while keeping the words in the same order, then print the result to the screen. This must utilize both the stack and queue classes from tasks 1 and 2.

Example input:

```
This is a test.
```

Output:

```
sihT si a tset.
```

3. If the user chooses the command line:
    a. Prompt the user for a line of text (enter key indicates end).
    b. Perform the same reverse on the text and output it as Task 3.2.b
4. Prompt the user to continue (repeat step 1) or quit (exit program).
5. Include try/catches for the stack and queue custom errors.
6. Include in the lab report a screen shot(s) of the output of a test. Include a diagram of what the stack and queue look like in solving the sample above (you may need to show various views of a single stack and queue).

Lab Submission:

1. Write a lab report including the following information:
    a. A description of the objectives/concepts explored in this assignment including why you think they are important to this course and a career in CS and/or Engineering.
    b. The sections from each task indicated to be included in the lab report.
2. Include all source code from all tasks, input and output files (if any), and any special instructions to compile and run those programs.
3. Package all files in a single zip folder and submit the file to canvas.

Lab Grading:

1. 20% - Lab attendance
2. 20% - Task 1 has been correctly implemented and meets all requirements.
3. 20% - Task 2 has been correctly implemented and meets all requirements.
4. 20% - Task 3 has been correctly implemented and meets all requirements.
5. 20% - Lab report contains all required information and is well written.

If program fails to compile, 0% will be given for that Task.