SE 461 – Spring 2021
Assignment #2- System Composition

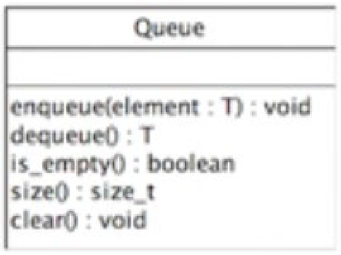
Due: 3/2/2021

In the first assignment you were introduced to some of the various features of the C++ programming language (possibly learning what you did and didn't really know about the language) as you created an ADT: an Array. In this assignment you will build upon that knowledge gained and use the Array you created to create three additional ADT's:

- **Stack**: An ADT that contains a list of elements such that the list has last-in, first-out (LIFO) semantics, i.e., elements are inserted into the front of the list and removed from the front of the list.
- **Queue**: An ADT that contains a list of elements such that the list has first-in, first-out (FIFO) semantics, i.e., elements are inserted into the front the list and removed from the front of the list.
- **Fixed Array**: An ADT that is an array, but cannot grow or shrink.

In addition, you will implement each of the ADTs listed above using **C++ templates**. This will allow each ADT to work with arbitrary data types. When you download the assignment files (see below), you will notice that there are no shell or skeleton files provided for Queue. This is because it is your job to define and implement the Queue class in C++.

When defining the Queue, please make sure you follow the design in the figure below, and implement the default constructor, copy constructor, destructor, and assignment operator although they are not highlighted in the figure:



More specifically, you must implement each method in the above figure using the following specification:

- **enqueue**: Adds the element to the end of the list.
- **dequeue**: Removes the element at the front of the list. If there are not elements in the queue, this method throws empty_exception, similar to the stack.
- **is_empty**: Test if the queue is empty. If the queue is empty, then this method returns true. Otherwise, this method returns false.
- **size**: Returns the number of elements in the queue.
- **clear**: Removes all the elements in the queue.

Exception Handling: Your ADTs are expected to offer **basic exception safety**.

Note: The same rules apply to use of the std library or any other libraries from assignment one.

Assignment Files:
The files for the assignment can downloaded from Canvas.

Development Process:
For this assignment, all development must take place on the **master branch**. It is strongly recommended that you commit and push often!

Submission:
All assignments must be submitted on GitHub (github.butler.edu). The name of your Butler GitHub repository must be as follows: **se461_spring2021_composition**.

You should submit all source, project (MPC), and Valgrind files.

Any additional files and directories can be added to the GitHub repository – for instance you will need a **driver.cpp** file (**Note**: Please include this name in your MPC file for compilation purposes.) for testing purposes, but they will not be reviewed or graded. Make sure the filenames have the correct capitalization. Failure to do so can result in our automated system not compiling and testing your code. You don't want to receive a 0 on the assignment for an incorrect filename!