Lab 2A

A Stack Class

Introduction

The goal of this lab is to enhance the **IntStack** to deal with the cases of empty and full stacks. You are provided with starter files in the working directory consisting of the first version of the **IntStack** class along with a test program containing a new command "check" that is intended to invoke the **IsEmpty** and **IsFull** functions that you will provide. **STACKSIZE** should be changed to 5 to make testing easier.

Suggested Time: 30 minutes.

Instructions:

1. Change STACKSIZE to 5. Build and run the starter application. Try a number of pushes and pops, including popping from an empty stack and pushing onto a full stack.
2. Modify the specification file **IntStack.h** to add prototypes for functions **IsEmpty()** and **IsFull()** member functions. These functions should return 1 (TRUE) if the stack is empty/full and 0 (FALSE) otherwise.
3. Modify the code file **IntStack.cpp** to implement the functions **IsEmpty()** and **IsFull().**
4. Modify the test program (file **TstStack.cpp)** to call the new member functions in response to the "check" command and print appropriate messages. Build and test your program.
5. Modify the code file **IntStack.cpp** to check for stack being full in **Push()** and for the stack being empty in **Pop().** If an error is encountered, just print an error message and abort. (If you use the **exit()** function, you will need an include of the <**cstdlib**> header file.) Build and test your program.

Lab 2B

Multiple Stacks

Introduction

This lab will give you more practice with the **IntStack** class. In your program you will create two instances of the **IntStack** class. The goal is to write a program with a loop that will move data from one stack to the other by popping from the first stack and pushing onto the second stack, until the first stack is empty.

Suggested Time: 30 minutes.

Instructions:

1. Create an empty console application project **MoveStack**.
2. Add the files **IntStack.h** and **IntStack.cpp** from the starter folder to your project.
3. Add a new file **MoveStack.cpp** to your project.
4. In this file implement a main program that will define two instances of **IntStack**, initialize them, push three different elements onto the first stack, and print the contents of both stacks. Build and run your program at this point.
5. Add code that in a loop will pop elements from the first stack and push them onto the second stack, until the first stack becomes empty. Then again print the contents of both stacks. In what order will the stack elements appear for the second stack compared with the original order of the elements in the first stack?
6. Build and run your program.