CS3100/5100: Data Structures and Algorithms

Fall 2017

Programming Assignment #2

1. Project Description

For this assignment, you will implement a stack for storing and retrieving employee records using a Singly Linked List. You should begin by creating class Employee, which will contain information about a single employee (see below). Then you will implement a Singly Linked List based Stack, in which each linked list node stores an Employee record. An Employee record inside a linked list node should support the following fields: Last Name (string), First Name (string) and Employee ID (integer). I will provide a data file that contains lines with three items: Last Name, First Name and Employee ID. Last Name, First Name and Employee ID are separated by spaces. The valid Employee ID is a number from 0 - 9999999.

A stack is a container of objects that are inserted and removed according to the last-in-first-out principle. The header file SStack.h will be provided to you. SStack.h is the header file that specifies the interface of the abstract data type Singly Linked List based Stack that you need to implement. You must implement all the functions in SStack.h.

To test the Singly Linked List based Stack, a small employee database with many employee records is provided to you. In the beginning of your main() function, you should open the input file, and then create a Singly Linked List based Stack: Then read a specified number (for example, 100) of the employee records from the input file, and push these employee records onto the stack. After you pushed the employee records onto the stack, you should test ALL the other functions in SStack.h. You should print out a message describing the operation you tested.

Then in your main() function, you will provide a user interface (MENU) to test the member functions of the stack ADT and to support the following operations.

1. Push: prompt user for all fields, create an employee record and push it onto the top of the stack, then print the number of the elements in the stack, and also the top 5 elements in the stack to the screen if the number of elements in the stack is larger than 5.

2. Pop: prompt user for a number, and if the number of the elements in the stack is larger than the number entered by the user, pop the specified number of the elements from the stack, and print the employee records popped from the stack, and the remaining number of the elements in the stack to the screen. Otherwise, pop all the elements from the stack.

3.Top: print the employee record on the top of the stack and the remaining number of the elements in the stack to the screen.

4. Print: Print all the elements in the stack onto the screen.

5.Save: save all the elements in a stack to a file on the disk.

6. Quit: clear all the elements in the stack, so there is no memory leak in your program, and quit.

2. Requirements

The main.cpp file includes code that will test the implementation of your member functions of your Stack ADT.

1). In SStack.cpp, you must implement all the functions in SStack.h.

2). Your code should follow the Code Standards posted on Pilot. Your code will be graded according to its correctness, efficiency, organization, and readability.

3). Make sure that each file includes your name and email address in the header comments.

3 Code Submission and Testing

1). In order to use the c++ compiler environment installed under the school's unix server, unixapps1.wright.edu, you need to connect to this unix server remotely using a secure shell client, putty. You can remotely connect to this unix server, unixapps1.wright.edu, on campus from a Wright State computer or use your own laptop connecting to the WSU wifi network named “WSU-Secure”. Note that you cannot remotely connect to this computer using ssh using computers outside Wright State University without installing VPN or use the campus “WSU\_EZ\_CONNECT” wifi network.

If you want to edit your C++ source programs under windows, download notepad++. Then edit your source programs using notepad++. After you finish editing the c++ source programs, using the secure file transfer client (WinSCP, you can download it online, and install it on your personal computer) to transfer your C++ source programs to the school’s unix server, unixapps1.wright.edu.

2). You must submit an ELECTRONIC COPY of your source program through Pilot before the due date. If for some reason Pilot is unavailable, submit your source code to the instructor Meilin Liu.

3). Submit the source code files including main.cpp, Qqueue.cpp, makefile, possibly a README file. You are recommended to submit a README file. It is also requied that you write down your name and email address in all the C++ source programs and in the README file and also explain your programs clearly in the README file.

4). All the submitted project files should have: Course Number / Course Title, Your Name, Prof.’s Name, Date, and the Project Name. If you did not include these required contents in your submitted files, then 5 points will be deducted. You also need to submit a makefile or a compiling command to compile your source codes. If not, another 5 points will be deducted.

5). The instructor will test your programs under the schools UNIX environment, e.g., unixapps1.wright.edu. It is YOUR responsibility to make your programs workable and runnable by others under school’s UNIX environment.

6). The programming assignment is individual. You must do the project by yourself. If you allow others to copy your programs or answers, you will get the same punishment as those who copy yours.