COMP220 — Lab 7

Fall 2017

Abstract

In this lab you'll implement your own specialized `stack` class. You'll use the C++ `vector` class and some of its methods to do so, gaining some experience creating classes.

The Problem

So far we've used four new collection classes to solve various problems: `stack`, `queue`, `map`, and `set`. However, we haven't yet seen exactly how these classes are implemented.

Later we will see how to implement these classes from scratch. For now, however, let's use `std::vector` to implement a stack. Actually, we'll implement a very special stack — it only holds non-negative (≥ 0) integers and it has a fixed *capacity*. That is, it can only hold so many elements. Our stack should provide the following methods:

- `size()` — return the number of elements currently in the stack.
- `isEmpty()` — returns `true` if the stack is empty; `false` otherwise.
- `isFull()` — returns `true` if the stack is full; `false` otherwise.
- `top()` — Returns the value at the top of the stack without removing that value from the stack. If `top` is called on an empty stack, you should return -1.
- `pop()` — Removes the element at the top of the stack (it does not return the element). Returns `true` if the pop was successful and `false` otherwise (a pop is unsuccessful if it is called on an empty stack).
- `push(int n)` — If the stack is not yet full, add `n` to the stack and return `true`. If the stack is full, do not change the stack, and return `false`

Our stack will also have a constructor that takes in an integer representing the maximum capacity of the stack. You will implement the stack by storing the integers in a vector, using vector methods to simulate the required methods. In other words, you will have a `std::vector<int>` as an *instance variable* of your class, and each of the above methods will operate on this instance variable.

However, you are restricted in what vector methods you may use: you may only use the `push_back`, `pop_back`, `at`, and `size` methods. You can find the documentation for these methods here: `http://www.cplusplus.com/reference/vector/vector/`.

During lab you should first focus on writing the header file for this class (the `class` definition, which includes the instance variables and method declarations, plus documentation), the method stubs in the implementation file, and several tests that use your stack class. Only then should you work on the actual method implementations.

Due Monday, 10/9. Submit the entire assignment as *lab7*. Your header file, documentation, and tests will be graded as lab 7, while your implementation will be graded as homework 6.