Lab 5: Vectors/Deques

UWYO COSC 2030

Introduction:

Welcome back to lab! This lab is meant to be a bit less complex than the last two, as instead of learning a new concept, you're just going to translate a previous concept into a new implementation. By now, vectors should be pretty familiar and you should at least understand what a deque is, if not exactly how it works. If not, these resources about vectors and deques should at least be enough to get you started. How I personally think of them is that a deque is like a chain, where you can work your way up the links from either end, and a vector is basically just an array that you don't have to know the exact size of before you declare it.

Lab 5:

First, you'll need to accept the assignment here: https://classroom.github.com/a/lP9Z9E5W

In this repo, you'll see a .cpp file, a .py file, and the usual README.md. You can start with whichever file you prefer, but I'm going to give the .cpp instructions first.

buildAndDeal: This function is going to use a vector to build a standard 52-card deck, then deal cards from that deck. If you don't gamble often, this is what that ought to look like. You'll need to store the suit and the face; as such, you're going to have trouble making a vector of any primitive data type. That's what card.h is for, use that file to make card objects and declare your vector to be of type card. As you can see, the part of the function that actually deals the deck is written for you, so you just have to load it.

parenCheck / stringReverse: These will work exactly how they did in lab 4, except this time you'll use deques. Reference your old work as needed, but you should find that deques make life much easier here. Perhaps there is a reason that the deque is the "industry standard" after all...

python parenCheck / stringReverse: Things are going to get just a little more complicated here, because Python, being the perpetual pain-in-the-ass that it is, has its own implementation of deque. The reference guide I used for that is here, and the logic should be mostly the same, just make sure you cross the t's and dot the i's on syntax.

Before you submit, remember to update your Readme and push it all to Github.

Submission:

For this week, you'll need to update your README.md and submit your completed Lab5.cpp and Lab5.py files. Don't bother including your executables that you've used for testing, unless for some reason you want them in your repo for your sake. This lab will be due this Sunday, March 9th, at 11:59pm.

Edited by Michael Stoll for UWYO COSC 2030