

Data Structures and Algorithms

**Lab 2: DSA\_Lab2.h**

**The Scenario**

The Dynamic Array you created last lab was a smashing success! However, your next task is to make a software using vectors. Specifically, to determine whether a given number is a palindrome. You gaze out across the ocean from the company yacht and begin to mull over your solution before opening your laptop. You open Visual Studio and begin.

**Vectors** are very similar to the **DynArray** class that you wrote last lab. They’re an array that has added functionality such as **random access**, **resizability**, and **adding/subtracting elements**. While last lab was learning how to create this data structure, a large part of this lab will be learning to use it.

A **Palindrome** is any pattern that is identical to itself when reversed. Palindromes have many applications in math and computer science and are even found naturally occurring in DNA and RNA. They can also be commonly found in language. An example of a palindrome is the word “racecar” and an example of a number that is a palindrome is “307703”. Some more examples are “UFO TOFU”, “Now I see bees I won”, and “Was it a car or a cat I saw?”. While trivial, a single digit is also a palindrome, such as “7”.

**What To Do…**

Open DSA\_Lab2.h. There will be instructions written in the comments on what is expected. Below is the gist of each function and variable.

***Variables:***

**mValues** A Vector that contains all the numbers to check as well as numbers that have been checked but aren’t palindromes.

**mPalindromes** Another Vector.Is it a Palindrome? If yes, it goes in here.

***Functions:***

**IsPalindromeNumber** Determines whether or not a given number is a Palindrome.

**Fill (Binary File)** Fill is an **overloaded function** in this program, meaning there are two versions of it that take different parameters. This version will read in numbers from a binary file and fill **mValues** with those numbers for later testing.

**Fill (Array)** This version of Fill fills **mValues** with numbers to test from an array. It should be simpler to write than the previous.

**Clear** Removes all elements from **mValues** and and shrinks it down to a capacity of 0.

**Sort** Sorts **mValues** in ascending or descending order based on the Boolean passed in.

**Access Operator** Overloads **[ ]** to give access to elements of **mValues**.

**Contains** Returns true if **mValues** has a given number stored in it anywhere.

**MovePalindromes** The only method that touches **mPalindromes**. Iterates through **mValues** and removes anything that is a Palindrome and adds it to **mPalindromes**. Remember, you wrote a method to do some of the heavy lifting for you in this lab.

**Tips, Tricks, and Resources**

Functions/Data Members available in the Vector class ( i.e. myVector.size() ) can be found on the Cplusplus.com documentation for vectors:

<http://www.cplusplus.com/reference/vector/vector/>

In case you need them, here are a few more:

<http://www.cplusplus.com/reference/array/array/>

<http://www.cplusplus.com/reference/string/string/>

<http://www.cplusplus.com/reference/fstream/ifstream/>

**Hint**: If you find yourself stuck, think outside the box while working on **IsPalindromeNumber.** Data types aren’t set in stone!

**Plagiarism**

Plagiarism and Academic Dishonesty are considered a **very** serious offense in this class and can have a range of consequences including suspension, and in very serious cases, expulsion. If you either share your code or copy someone else’s code, you will be given a **0** on your lab and can face further disciplinary action.

In other words, don’t cheat please!