**Lab 6**

**Arrays/Vectors**

**Lab Objectives**

* Be able to declare and instantiate arrays
* Be able to fill an array using a for loop
* Be able to access and process data in an array
* Be able to write a search method
* Be able to use parallel arrays
* Be able to use vectors

**Introduction**

Everyone is familiar with a list. We make shopping lists, to-do lists, assignment lists, birthday lists, etc. Notice that though there may be many items on the list, we call the list by one name. That is the idea of the array, one name for a list of related items. In this lab, we will work with lists in the form of an array.

It will start out simple with a list of numbers. We will learn how to process the contents of an array by finding the average of the numbers in the list. We will also explore searching arrays by finding the minimum and the maximum values in the list.

We will then move onto parallel arrays, arrays containing information related by index number. We will also investigate using dynamic arrays in the form of vectors.

**Task #1 Printing Arrays**

1. Copy the file classics.cpp. Compile, debug, and run after each item is completed.
2. Write a for loop to print the contents of the artists array.
3. Write a for loop to print out the contents of the artists array in reverse order.
4. These two arrays are parallel arrays. Write a for loop that will give you the following output:

1. Ode to Joy by Bach  
2. The Sleeping Beauty by Tchaikovsky  
3. Lullaby by Brahms  
4. Symphony No. 5 by Beethoven  
5. The Blue Danube Waltz by Strauss

**Task #2 Processing Arrays**

1. Copy the file arrayAverage.cpp and numbers.txt.
2. Declare a double array of size ARRAY\_SIZE.
3. Read the values from the file into the array.
4. Write a function called Average that takes constant parameters of the array and the size of the array, then returns the average of the values in the array.
5. Call the Average function in the main after the array was filled.
6. Print out the average rounded to 4 decimal places, appropriately labeled.
7. Compile, debug, and run the program. It should output the average of the values from the data file. Compare the computer’s output to your hand calculation using a calculator. If they are not the same, do not continue until you correct your code.

**Task #3 Searching Arrays**

1. Write a function called FindMax that takes constant parameters of the array and the size of the array, then returns the index of the maximum value in the array.
2. Call the FindMax function and store the result in maxIndex.
3. Write a function called FindMin that takes constant parameters of the array and the size of the array, then returns the index of the minimum value in the array.
4. Call the FindMin function and store the result in minIndex.
5. Use the maxIndex and minIndex to print the min and max values in the list, appropriately labeled.
6. Compile, debug, and run the program. It should output the largest and smallest of the values from the data file. Compare the computer’s output to your observation of the data file. If they are not the same, do not continue until you correct your code.

**Task #4 Vectors**

1. Copy the file arrayAverage.cpp from task #1 and rename as vectorAverage.cpp. In your Visual Studio project, remove the arrayAverage.cpp file and add the existing file vectorAverage.cpp that you just created.
2. Modify to use vectors instead of arrays.
   1. Add #include <vector>
   2. Declare a vector of doubles called value
   3. In the while loop, remove the index < ARRAY\_SIZE condition
   4. Read the values from the file into a double variable. Add that value to the vector. Continue until you hit the end of the file. The vector will dynamically adjust the size as you read in the data.
3. Modify the Average, FindMin and FindMax functions to take a vector instead of an array. Note: Only one parameter should be used.
4. Compile, debug, and run the program. It should output the average of the values from the data file. Compare the computer’s output to your hand calculation using a calculator. If they are not the same, do not continue until you correct your code.
5. Delete the last 3 elements in the vector. Call the Average function again and print the result.
6. Compile, debug, and run the program. It should output the average of the values from the data file. Compare the computer’s output to your hand calculation using a calculator. If they are not the same, correct your code.