CSE30: Lab #2 - Arrays

Overview

This lab will explore arrays in the C++ programming language. The goal of this lab is to create three simple programs. The knowledge required to complete this lab includes output to the console, read input from the user, if statements and for loops, and simple error handling, in addition to programming with arrays and strings with some simple operations.

Note: You need to have a separate program for each of the parts of this lab. When you submit your assignment through CatCourses, make sure that ALL PARTS are included.

Getting started

Create a new directory in your main development directory (CSE030) called Lab_02. Try to use the terminal on your own, without getting help from the TA to setup the new directories (try to learn/remember the terminal commands).

Reminder. Once you have created a file and understood its content, you want to compile the source file (ex. main.cpp) into an executable so that you can run it on your computer. **Note: EVERYTIME you modify your source file, you MUST re-compile your source file for the changes to take effect.** You will compile from the command line by running:

```
g++ <source> -o <executable>
```

where g++ is a program (already installed on your Linux system) that compiles C++ source files, **source** is the source file for your program (main.cpp in this example), **-o** tells the compiler that you want to give the executable its own name, and **executable** is the name you want to give your program. In order to compile your first program, you would have to run:

```
g++ main.cpp -o main
```

Assuming that your program compiled successfully (i.e. no errors were found), you can run your program by typing **"./main"** in the terminal/console.

Good coding practices (worth 2 points!)

Writing code that is understandable by humans is as important as being correct for compilers. Writing good code will help you complete the code, debug it and … get good grades. It is very important to learn as soon as possible, because bad habits are hard to get rid of and good habits become effortless. Someone (guess who) reads your code will be in a better mood if it is easy to understand … leading to better grades! This lab will include 2 points (10% for code quality):

- Explanations with comments
- Meaningful names
- Indenting of blocks { } and nesting …
- Proper use of spaces, parentheses, etc. to
- Visible, clear logic
- One / simple statements per line
- Anything that keeps your style consistent

(Exercise) Create – array1.cpp

In this part of the lab, we want to check if an array of numbers input by the user is increasing. This happens if each element of the array contains a value that is larger than the value contained in previous elements. The program will work as follows:

First, you should ask the user to enter the size of the array, by outputting:

"Enter the size of the array: "

to the console. If the user enters an incorrect size, you should output the error message

"ERROR: you entered an incorrect value for the array size!"

and exit the program. If the input is a valid size for the array, ask the user to enter the data, by outputting:

"Enter the numbers in the array, separated by a space, and press enter: "

Hints: Think about how to enter individual elements in an array. **"cin"** can only read one word/number without space.

Once the input is complete, check if the array is increasing:

- If it is, write **"This IS an increasing array!"** to the console output.
- If it is not, write **"This is NOT an increasing array!"**.
- Print your array **in one line** with elements separated by a space.

Before starting to write your program, use a piece of paper or a text editor to write the pseudocode of **this algorithm**. You will need to submit the pseudocode in order to receive full credit. Again, there is no unique way to write pseudocode. It will be good enough as long as you can understand it and translate it to C++ code. Ask your TA if you are not sure about the structure of the pseudocode.

Example runs (input is in italic and bold):

Enter the size of the array: ***5***
Enter the numbers in the array, separated by a space, and press enter: ***1 2 3 4 5***
This IS an increasing array!
1 2 3 4 5

Enter the size of the array: ***6***
Enter the numbers in the array, separated by a space, and press enter: ***1 3 5 2 4 6***
This is NOT an increasing array!
1 3 5 2 4 6

Enter the size of the array: ***-5***
ERROR: you entered an incorrect value for the array size!

(Exercise) Create – array2.cpp

In this part of the lab, we want to create and output a string that is the reverse of a string that has been input by the user. Write a program that asks the user to input a string, by writing:

"Enter the string to reverse: "

to the console output. Read the string input by the user, create a string that contains the characters in the reverse order, and then output:

"The reverse of the string is: "

and the new string. For example:

- If the input string is **"lab2"** the new string and output is **"2bal"**
- If the input string is **"homework"** the new string and output is **"krowemoh"**
- If the input string is **"1"** the new string and output is **"1"**

Remember that in C++, a string is simply an array of characters. Consequently, array indexing can be used to get desired characters in the string.

(Exercise) Create – array3.cpp

In this part of the lab, we want to check how many negative values are in a two-dimensional (square) array, if any. Write a program that asks the user to input the dimension (n) of the square (n x n) array, and then asks the user to input the values 1 row at a time. For example:

"Enter the size of a 2D array: "

"Enter the values in the array for row 1, separated by a space, and press enter: "

Limit the size of the array to maximum 10 x 10 and check for errors. Once the array is initialized, check if there is any negative element in the array and display the result:

- If there is no negative value: **"All values are non-negative!"**
- If there are # negative values: **"There are # negative values!"** … where # is the number of negative values found.

Example runs (input is in italic and bold):

Enter the size of a 2D array: ***4***
Enter the values in the array for row 1, separated by a space, and press enter: ***1 5 6 3***
Enter the values in the array for row 2, separated by a space, and press enter: ***-5 6 -12 5***
Enter the values in the array for row 3, separated by a space, and press enter: ***9 4 -3 1***
Enter the values in the array for row 4, separated by a space, and press enter: ***7 5 -3 9***
There are 4 negative values!

Enter the size of a 2D array: ***12***
ERROR: your array is too large! Enter 1 to 10.

Enter the size of a 2D array: ***-10***
ERROR: you entered an incorrect value for the array size!

Enter the size of a 2D array: ***3***
Enter the values in the array for row 1, separated by a space, and press enter: ***5 9 1***
Enter the values in the array for row 2, separated by a space, and press enter: ***7 5 3***
Enter the values in the array for row 3, separated by a space, and press enter: ***6 5 4***
All values are non-negative!

What to hand in

When you are done with this lab assignment, you are ready to submit your work. Make sure you have included the following ***before*** you press Submit:

- Your array1.cpp, array2.cpp, array3.cpp, and a list of Collaborators.
- File attachments of the pseudocode you create in this lab. If you write your pseudocode on papers, scan them as images and attach the images. If you write your pseudocode in text editor, save it as a text file and attach it in your submission.