**Lab Experience Eleven**

**Objectives:**

1. Using Array initialization lists.
2. Understanding memory management within a programming environment.
3. Understanding what happens when bounds checking is not performed.
4. Performing a hand trace of array manipulation.

**Background**

Whenever a program is executed the memory is allocated for the variables declared within the program. All variables have two components associated with them: Address and contents. Therefore it is possible to change the contents of a variable just by referencing the variable’s address and not the name of the variable.

**Lab Exercises**

**Directions:**

This lab consists of writing computer programs using arrays, sorting and searching techniques. Answer the following exercises based on material presented in lecture and found in chapters 1-7 of the textbook.

**Exercise 1**

Answer the following questions by performing a hand-trace of the code by making a table of values using Microsoft Word’s table feature.. **Write down the contents of the array after each pass of the *for loop*.**

NOTE: ***A screen shot of the output or just a listing of the output will be awarded a grade of zero.*** You need to perform a hand-trace, which means you are the computer. **See pages 130-132 for details about hand-tracing a program.**

1. Trace the following program.

const int SIZE = 6;

int main() {

int values[SIZE] = { 7, 2, 3, 8, 9, 1 };

bool swap;

int temp;

int limit = SIZE ;

do {

limit--;

swap = false;

for (int count = 0; count < limit; count++)

{

if (values[count] > values[count + 1])

{

temp = values[count];

values[count] = values[count + 1];

values[count + 1] = temp;

swap = true;

}

}

} while (swap);

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Values | | | | | |
| **Values[0]** | **Values[1]** | **Values[2]** | **Values[3]** | **Values[4]** | **Values[5]** |
| 7 | 2 | 3 | 8 | 9 | 1 |
| 2 | 3 | 7 | 8 | 1 | 9 |
| 2 | 3 | 7 | 1 | 8 | 9 |
| 2 | 3 | 1 | 7 | 8 | 9 |
| 2 | 1 | 3 | 7 | 8 | 9 |
| 1 | 2 | 3 | 7 | 8 | 9 |

**Exercise 2**

Download the program arrayExample1.cpp from D2L. The file is listed below with the line numbers being used as a reference.

//

// An array manipulation example

// Programmer: your name here

// Subject: CSCI 1106

//

1. #include <iostream>

2. #include <iomanip>

using namespace std;

3. const int MAXELEMENTS = 8;

4. typedef int integerArray[MAXELEMENTS]; // create an alias for a one dimensional

// array containing integers

5. int main(){

6. integerArray prime = {2, 3, 5, 7, 11, 13, 17, 19};

7. for(int i = 0; i < MAXELEMENTS ; i++){

8. cout << setw(3) << "prime[" << i <<"] = " << prime[i] << endl;

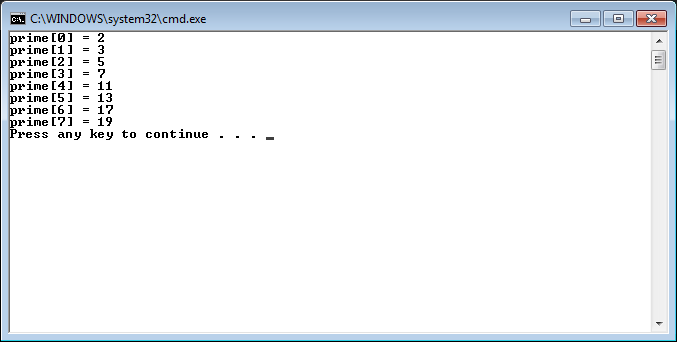
}

9. return 0;

}

Statement #4 creates an alias for the array name. The statement is called a **typedef statement** and it defines a synonym for the specified type-declaration. The identifier in the type-declaration becomes another name for the type, instead of naming an instance of the type. **You cannot use the typedef specifier inside a function definition.**

1. Statement #6 initializes the array elements to whatever is declared in the list. Execute the program to see what the output is.



1. It is an error to give too many values in an initializer list of an array. Increase the number of initializers in statement #6 to determine if our compiler catches the error. Describe what happens.

**When the compiler gets to the initializer list, I’d guess it loops through one by one to add them, since initialization lists are created at compile time. Once it finds the last value, it is unable to add any more because the max size was previously set to 8, and so spits out an error.**

1. Change the number of initializers in statement #6 to only 4 items and describe what happens when you run your program.

**Because an initialization list automatically initializes unspecified elements to 0, the compiler just set the remaining 4 values to 0. Because of this, the output for prime[4]-prime[7] Is now 0.**

1. When completed close the solution.

**Exercise 3:**

Download the program subscriptManipulation.cpp from D2L.

1. Note: Visual Studio “pads” arrays with extraneous storage locations.
2. What is the output when you execute the program? Is it what you expected? Why or why not?

**It outputs all 0s for first, all 1s for second, and all 2s for last. This is what I expected as initialization lists automatically set unspecified locations to 0.**

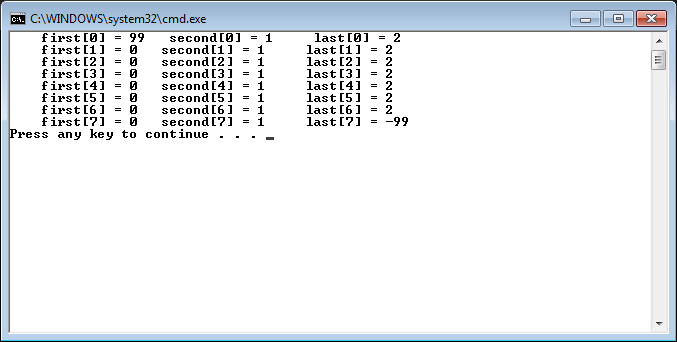
1. What happens when indices and subscripts get out of range? Add the assignment statements listed below before the for loop to determine what happens when subscripts are out of range.

**second[-3] = -99;**

**second[10] = 99;**

Re-compile the program and execute the program. Paste the output generated by the program into a word document.

**When indices and subscripts are out of range, it modifies other memory locations. In this case, it modified first[0] and last[7], though that’s odd since if second[10] is just 1 int-size in front of second, it seems like it should have modified last[0] instead.**



1. Modify the for loop to display the elements in the array second with indices from ­10 to 18. The loop should execute from ­10 to 18.

* Change the initialization value of the for loop to ­10; i.e. **for(int i = ­10 ; i < 18; i++)**
* **Comment** out the statement:

**cout << setw(10) << "first[" << i <<"] = " << first[i] ;**

* **Comment** out the statement:

**cout << setw(10) << "last[" << i <<"] = " << last[i] << endl ;**

Are some of the values being displayed from the array first and last?

**Yes.**

Why does this occur?

**It occurs because trying to access things outside of the normal subscripts will just access memory locations based on an offset of the size of the type in the array. Because these are all ints, the offset will always be the same and allow it to correctly read the memory locations (other than the 3\*4 bytes of garbage/another variable type in between each array).**

How are consecutive arrays stored in memory? Create a memory map as shown in class and in your textbook

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Last | (3\*4 bytes of garbage) | Second | (3\*4 bytes of garbage) | First |

**Due Dates:**

As indicated on the Lab Experience Eleven assignment folder.

**What to hand in:**

1. Hand in a print out of your word document.
2. Place your word document into the assignment Lab Experience Eleven folder.