CS-212: LAB 4 – Sorting Arrays

Learning Objective:

- Sorting Arrays
- Writing a lab in C++ using Object Oriented Programming (OOP)

This lab has three parts:

- Part-A: The data
- Part-B: The functions
- Part-C: Writing the main function

Lab 4 Setup:

1. Open a new terminal window to SSH into your account on the [bingweb.binghamton.com] server. When logged in, make sure you are in your home directory by typing the following command if you are not already there. The "~" is just a shortcut symbol that means home directory:
   cd ~

2. Change your current directory to the "CS212" directory by typing:
   cd CS212

3. Confirm that you are in the "CS212" directory by typing:
   pwd

4. Create a directory for Lab4.
   mkdir Lab4

5. Move into the new directory
   cd Lab4

6. Create a new c file for you to write your lab.
   > <Last Name><FirstInitial>_Lab4.cpp
   > <Last Name><FirstInitial>_Lab4_Functions.cpp
   > <Last Name><FirstInitial>_Lab4.h

INTRODUCTION:

In this lab you are going to have 3 arrays (size 50). Data for two of the arrays will be read from data files. One data file I will give you and the other data file you will create (*You must hand this in*). The third array will be random integers created at run-time. (*You may copy some of your last lab for this part or look at Appendix A for more information.*)

You will write a function that will print the array in table format. (*Again, you may start this function with some of the code from your last lab, but there is an add-on*.)

Then you will write a swap function and three sorts. One sort for each array. Your sort functions must use your swap function when appropriate.

Your output file will show your array in table format both before and after calling your sort function are called.

This lab will be done with C++. You have to use g++ instead of gcc in Unix.

It is required that you **MUST** use a class.

Part A: The data

Your class you will have an integer array and a double array. Both of size 50. (Looking ahead: In the main you instantiate the class three times, two times you will use the integer array and the other time you will use double array)

The first set of integers for the array will come from an input file that you can download from Brightspace.

You are to create your second input file. Create a text document and put 50 different numbers in it. Numbers can be in the range of 0 to 1000 and MUST HAVE SOME DECIMALS. This will be for the double array.

Your third set of numbers will be random integers from a random number generator. The numbers must be between 0 and 200.

Part B: Functions / Methods (A method is a function inside a class).

FUNCTION: PrintHeader, CenterString, PrintDivider

- Your header must be printed to the screen **and** to your output file
- Do not put these functions in the class.

FUNCTION: ReadDataFromFile

- Do not put these functions in the class.
- This function will read in the data into an array that is declared in the main.
- Overload this function so that one instance will read data into an integer array and the other will read into a double array.

Looking ahead: You will create three instances of your class in the main. For one instance, you will pass in the integer array. For another instance, will pass in a double array. For the third instance, you will use the default constructor (this will generate random integers).

METHOD: Constructor

You will have three constructors:

One will accept an integer array. The constructor will copy the array into the array in the class.

One will accept a double array. The constructor will copy the array into the array in the class.

One will have no parameters, this will call GenerateRandomNumbers for the third array (integers).

NOTE: In your main you will create three instances of your class.

NOTE: If you do the extra credit, you will only have two constructors.

METHOD: GenerateRandomNumbers

- This function will fill the integer array with random numbers

METHODS: PrintIntegerTable, PrintDoubleTable and CenterString

- This function will print an array in rows of ten
- All columns and rows must have a label.
- This function must take string parameter that will be used for the title of the table. The title must be centered across the length of the table

METHOD: Swap

- This function will take two integers and swap them
- This function needs to be called by your sorts when applicable

METHOD: SelectionSort

- See playbook notes for details
- You may assume that this sort will use the integer array.

METHOD: BubbleSort

- See playbook notes for details
- You may assume that this sort will use the integer array.

METHOD: InsertionSort

- See playbook notes for details
- You may assume that this sort will use the double array.

PART C: Your main function

- Create a double array and an integer array that will be passed into your class constructors.
- Open your files
- Print your header to the screen and the output file
- Read in the data files for the first two arrays

- Create three instances of your class.
- The first one will pass in an integer array.
- The second class will pass in a double array.
- The third class will use the default constructor.

- Print array 1 – unsorted
- Call the Selection Sort with the first class.
- Print array 1 – sorted

- Print array 2 – unsorted
- Call the Bubble Sort with the second class.
- Print array 2 – sorted

- Print array 3 – unsorted
- Call the Insertion Sort for the third class.
- Print array 3 – sorted

EXTRA CREDIT: 20%

If your lab does not run with proper output, no extra credit may be earned.

Teach yourself how to use template for classes and functions.

Your class will be a template class and only need one class.

Your swap function must use the template type.

You will only need one printTable method

Your class will only need one array.

Your ReadDataFromFile function must be a template function. This function will no longer be overloaded.

Minimal help will be given from CA's and your teacher for the extra credit.

Executing your lab:

Your program must run on the Unix operating system with 0 errors and 0 warnings.

To execute a lab in C++, **you must use g++ instead of gcc.**

The syntax is the same:

```
g++ *.cpp -Wall
```

The executable file will be a.out, just like with gcc

Valgrind must run with 0 errors and 0 warnings.

To run Valgrind:

```
valgrind --leak-check=full ./a.out
```

Submission Files:

- <Last Name><FirstInitial>_Lab4.cpp
- <Last Name><FirstInitial>_Lab4_Methods.cpp
- <Last Name><FirstInitial>_Lab4.h
- Numbers2.txt
- Lab4_Output.txt

NOTE: Functions will be placed in the lab4.cpp file with the main. Only methods will be in the Methods.cpp file.

APPENDIX A:

Code to generate a random number:

```c
// Include for the srand and rand function
#include <stdlib.h>

// Include for the time function
#include <time.h>

int main (void)
{

   int randomNumber;
   time_t t;

   // Intializes random number generator to the clock
   srand((unsigned int) time(&t));

   // Create a random number between 1 and 20
   randomNumber = rand() % 20 + 1;
```

Note: Don't forget to include the necessary header files.

Note: Your header file includes, and prototype should be in your header file.

ADDITIONAL PARTIAL CREDIT GRADING RUBRIC:

NOTE: The full rubric is on Brightspace under Misc

NOTE: If the lab does not use a class, then the lab is incomplete and will receive a 50%.

Infraction	Points Lost
Using pointers instead of reference parameters	-5 points
Random numbers are not in the correct range	-5 points
Methods and Files are not in the correct location	-5 points

Note to Grader: Any partial credit outside of the rubric must be approved by the teacher.