Lab Assignment 2

Start Assignment

Due Tuesday by 11:59pm **Points** 24 **Submitting** a file upload
Available Sep 15 at 8am - Sep 23 at 11:59pm 9 days

Sorting

Application: Rainfall Statistics

In this week's lecture on sorting we discussed a stable, straightforward algorithm in selection sort as well as a recursive algorithm known as merge sort. This lab will provide an opportunity to appreciate the significance of a stable sort in analyzing data for rainfall statistics accrued over a calendar year.

Make sure you have read and understood

- ***Unit module 4***

- ***C++ Coding Style Guidelines*** (https://ohlone.instructure.com/courses/18987/files/3097729/download?download_frd=1)

before submitting this assignment. Hand in only one submission.

Lab Assignment Objectives

1. Implement a stable O(n^2) sort with efficient space requirements.
2. Apply the selection sort heuristics to a practical statistical analysis.

Understand the Application

Write an interactive program that obtains user entry for the total monthly rainfall in their geographical area. Use a selection sort algorithm to sort the statistical rainfall data in increasing order.

The Program Specification

Selection Sort Algorithm Description

The first iteration of selection sort selects the smallest element and swaps it with the first element. The second iteration selects the second-smallest element(which is the smallest remaining element) and swaps it with the second element. This process continues until the last iterations selects the second-largest element and swaps it with the second-to-last index, leaving the largest element in the last index. At the *ith* iteration, the smallest *i* elements are sorted into the first *i* elements.

Interactive Data Collection

Obtain user input for rainfall figures for each of 12 months. Based on the user input data generate the following statistical report values over the 12 month period: ***total*** amount of rainfall, ***average*** amount of rainfall, the ***largest*** recorded monthly rainfall, the ***smallest*** recorded monthly rainfall.

Use a selection sort to arrange the data for your report.

Generate a report summary to display your monthly rainfall statistical findings.

Testing Specification

Input Errors

Negative input figures are not to be accepted for rainfall figures. ***Pigeonhole*** the user to obtain valid input data.

Test Run Requirements

Submit a test run validation to demonstrate the successfully collected and sorted statistics. Include a demonstration of user input validation.

Example Test Run

Here is an example run:

```
/* ---------------------- Sample run ---------------------------------------
Enter the rainfall (in inches) for month #1: 0
Enter the rainfall (in inches) for month #2: 0
Enter the rainfall (in inches) for month #3: 1
Enter the rainfall (in inches) for month #4: -1
Rainfall must be 0 or more.
Please re-enter: 3
Enter the rainfall (in inches) for month #5: 6
Enter the rainfall (in inches) for month #6: 8
Enter the rainfall (in inches) for month #7: 77
Enter the rainfall (in inches) for month #8: 2
Enter the rainfall (in inches) for month #9: 1
Enter the rainfall (in inches) for month #10: 1.5
Enter the rainfall (in inches) for month #11: 3.5
Enter the rainfall (in inches) for month #12: 66.9

The total rainfall for the year is 169.90 inches.
The average rainfall for the year is 14.16 inches.
The largest amount of rainfall was 77.00 inches in month 7.
The smallest amount of rainfall was 0.00 inches in month 1.

Here are the rainfall amounts, sorted in ascending order:
---------------------------------------
0.00
0.00
1.00
1.00
1.50
2.00
3.00
3.50
6.00
8.00
66.90
77.00
------------------------------------------------------------------------ */
```

What to Turn In

Hand in **4** files: **Makefile**, **a2.cpp**, **rain.cpp** and **rain.h** **No zip files.**

- **Makefile :** make text file to compile and link your solution
- **a2.cpp :** test driver
- **rain.cpp :** user defined functions to determine statistical data
- **rain.h :** user defined header

Tips and Requirements

1. Define and use named constants in lieu of literal values.
2. Ensure that your solution is well organized. Provide a program header and comments to document

and organize your source code. User defined functions need be documented.

3. **Provide a description in your program header why a stable sort is important in this particular sorting application.**
4. Provide a commented out copy of your program run. Enclose the run inside of multi-line comment delimiters so that your program will run in the grader test bed. Place the run after your program source code in the main.cpp file.

Submission Resources

For more information on how to submit your assignment, please visit:

- **How do I submit an online assignment? Canvas Student Guide (https://community.canvaslms.com/docs/DOC-9539)**
- **Assignments Overview Canvas Video Guide (https://community.canvaslms.com/videos/1122-assignments-overview-students)**
- **Assignments Submissions Canvas Video Guide (https://community.canvaslms.com/videos/1121-assignment-submissions-students)**

Submitting multiple files to an assignment

Your lab 2 assignment requires uploading more than one file; you should upload these 4 files as one submission. In this assignment you need to upload 1 text Makefile, 2 **.cpp** files (a2.cpp and rain.cpp) and 1 **.h** file (rain.h). To add these files, the **Add Another File button** is clicked to **upload the two files one by one. Check to make sure that both files uploaded okay.** When finished click **Submit Assignment.**

Questions?

Feel free to **ask** in the forum!

Lab 2 Rubric

Criteria	Ratings			Pts
On time submission	**4 pts** **On time**	**2 pts** **One day late**	**0 pts** **Two days late**	4 pts
Header file Satisfies the strpkg.h source statements as specified in the assignment.	**2 pts** **Full Marks**		**0 pts** **No Marks**	2 pts
Source file Satisfies the strpkg.cpp source statements as specified in the assignment.	**6 pts** **Full Marks**		**0 pts** **No Marks**	6 pts
Driver file Satisfies the main.cpp source statements as specified in the assignment.	**6 pts** **Full Marks**		**0 pts** **No Marks**	6 pts
Commented out test validation run included A copy of the program test validation run output is attached after the source code in the main.cpp test driver file (enclosed within comment delimiters). The run matches the source code submitted. The test run demonstrates validation of user input, pass by value and pass by reference.	**4 pts** **Full Marks**		**0 pts** **No Marks**	4 pts
Coding Style Includes a program header. Is correctly formatted (i.e. follows code style rules). User defined functions are documented.	**2 pts** **Full Marks**		**0 pts** **No Marks**	2 pts
				Total Points: 24