Lab 7: Heapsort

1. Introduction

In the lecture today we talked about the heap data structure. In this lab, you will implement a min heap as a class (MinHeap), and implement Heapsort which uses your MinHeap class to sort an unsorted array.

The code skeleton is under: **/comp/15/files/l7**

1. Log in to the CS homework server.
2. Move to your **comp15** directory that you created in Lab 1.
3. Create a directory named **lab7** under the comp15 directory.
4. Move to the lab directory.
5. Copy the code skeleton to the current working directory.
6. Leverage the features of Git to manage your progress toward writing the program.

The code skeleton provides you with three files: **test.cpp**, **MinHeap.hpp**, **MinHeap.cpp**.

- Note that the copy constructor and assignment operator of the MinHeap class are commented out, so that this lab can be completed within our lab time. However, you are encouraged to work on them later on.
- **test.cpp** provides you with the skeleton of a function that you will implement (See Requirements section) and two tests with assert() that your implementation is supposed to pass.

2. Requirements

1. Implement the **MinHeap** class.
    a. An instance of the MinHeap class can hold an arbitrary number of integers.
    b. The insert() method: In order to focus on the structure of a min heap in this lab, you can assume that no duplicate numbers will be inserted.
    c. The extractMin() method: You will define what should happen when this heap does not have any items.

2. Implement the **heapsort()** function in **test.cpp**
    a. The function takes a pointer to an unsorted array of integers and the size of the array. The goal is to sort the array in ascending order using your MinHeap.
    b. The idea is that you first insert all the items that are in the given array into your min heap and then retrieve the items from your min heap in the particular order to

fill out the given array.
(Note: The version you will implement in this lab is not in-place.)

3. Pass the tests with assert() written in **test.cpp** with no memory leaks and no memory errors.

3. README

Create the README file that includes the following categories with appropriate section headers.

1. **Name**: Your name.
2. **Date**: The last updated date.
3. **Summary**: A brief summary of the lab.
4. **Files**: A list of files that are necessary to build and test the program.
5. **Instructions**: A sequence of commands to compile and test the program. Note that you are expected to report procedures without using the make command.
6. **References**: A list of citations to information used to complete the lab.

4. Submission

Submit your files listed below using Gradescope.
Files: **test.cpp Heap.cpp Heap.hpp README**