Name: _____ Score: _____

March 17, 2022 CSC-340 Midterm: Foundations and Heapsort

CoCalc Portion

All programs submitted will be graded on Functionality and Code Quality: Expectations

- Appropriate description of the program at the beginning.
- Meaningful identifiers used.
- The logic of the code is easy to follow.
- Comments are used appropriately.
- The code has proper spacing.
- The code has proper indentation.
- Corresponding curly braces align (c++).
- The program compiles.
- User interface readable, clear, appropriate, and free of typos.
- The program logic is correct and thus behaves according to specifications.

6.4 The heapsort algorithm

Given an input array, the heapsort algorithm acts as follows:

- Builds a max-heap from the array.
- Starting with the root (the maximum element), the algorithm places the maximum element into the correct place in the array by swapping it with the element in the last position in the array.
- "Discard" this last node (knowing that it is in its correct place) by decreasing the heap size, and calling MAX-HEAPIFY on the new (possibly incorrectly-placed) root.
- Repeat this "discarding" process until only one node (the smallest element) remains, and therefore is in the correct place in the array.

Listing 1: heapsort

```
HEAPSORT(A,n)
    BUILD-MAX-HEAP(A,n)
    for i = n downto 2
        exchange A[1] with A[i]
        MAX-HEAPIFY(A,1,i-1)
```

1. (10 points) Using Figure 6.4 pg. 161 as a model, illustrate the operation of HEAPSORT on the array $A = \langle 5, 13, 2, 25, 7, 17, 20, 8, 4 \rangle$. Write your complete solution on paper (or in lucidchart or another digital format) and submit a pdf image (or a link in the assignment comments) of your solution in Canvas. Include your name on your submission!

2. (20 points) Open the file `heapsort.cpp` OR `heapsort.py` in your `Midterm-2-HEAPSORT` folder. Modify the program description appropriately. Add the missing code for the heapsort psuedocode (all of the other algorithms are provided) so that the program demonstrates sorting via the heapsort algorithm. ****DO NOT delete or modify existing code.****

   The final run:

```
Run max-heapify on vector: A = {-1000,16,4,10,14,7,9,3,2,8,1}
Heap A = <16,4,10,14,7,9,3,2,8,1>

After max-heapify: A = {-1000,16,14,10,8,7,9,3,2,4,1}
Heap A = <16,14,10,8,7,9,3,2,4,1>

Consider array: B = {-1000,5,13,2,25,7,17,20,8,4}
Heap B = <5,13,2,25,7,17,20,8,4>

After heapsort: B = {-1000,2,4,5,7,8,13,17,20,25}
Heap B = <2,4,5,7,8,13,17,20,25>
```