COSC 2436 - Data Structures

Program 8 – Heaps

Be sure to read through Chapter 13 section 13.2 of the textbook before starting this assignment.

In this assignment you will write a program that converts an integer array into a heap and displays the results both as a binary tree and a list of numbers. You will be implementing algorithms from the text. Note that element 0 in the array is reserved.

Write the following functions:

heapify – write a function that takes an integer array and turns it into a heap using percolateDown (the heapify algorithm in your textbook). Your function should have 2 parameters; an integer array and the array size (the number of elements in the array to be heapified).

percolateDown – write a function that takes a semiheap and turns it into a heap (the percolateDown algorithm in your textbook). Your function should have 3 parameters; an integer array, the root of a subtree, and the array size (the number of elements in the array to percolate).

displayArray – write a function that displays the contents of the array, horizontal. Your function should have 2 parameters; an integer array (ie. address of the first item to be displayed) and the array size (the number of elements in the array to be displayed).

Use the following function to display a binary tree:

displayTree – Use the displayTree function in your text from Figure 13.1 to display a binary tree in tree format. Note that you will need to pass the address of the first item to be displayed because this function will not expect that element 0 is reserved. Note: **There is an error in the displayTree function in my copy of the texbook.** The calculation for the number of levels should be:

```
int(ceil(log(double(n+1)) / log(2.0)))
```

For 10% extra credit, write the following function:

heapsort – write a function to perform a heapsort on an integer array with O(n*logn) time complexity (the heapsort algorithm in your textbook). This function should have 2 parameters; the integer array and the size (the number of items in the array to be sorted).

Note 1: Do not create a Heap class. This will be a program with functions.

Note 2: The C++ Standard Template Library contains functions for creating and manipulating heaps. I WILL NOT accept an assignment that uses the STL. You must write your own heap functions.

Note 3: The *sort* algorithms in Chapter 13 of the textbook use index 0 of the array for special purposes. So, all the *sort* algorithms in the textbook assume that the data to be sorted are stored at indexes 1 through size (instead of the normal 0 through size - 1). So if you follow the algorithms in the book, the size of your array needs to be *number of items to sort* + 1.

Note 4: **There is an error in the percolate_down algorithm in my copy of the textbook.** Be sure to check if this error is in your book. The percolate_down algorithm is on page 739 of your textbook.

My book has:

2. While r <= n do the following:

The correct instruction is:

2. While c <= n do the following:

Note 5: There is also a quirk in the heapify algorithm. When it calls the percolate_down algorithm, r should be **passed by value**. In other words, the percolate_down algorithm modifies r, but this should not change the r in the heapify algorithm.

Be sure to document your functions with a precondition and a postcondition. These should clearly state whether the items to be heapified/sorted are in indexes 1 through size, or 0 through size - 1.

Put all your code in a single program file (.cpp) with your last name appended preceded by an underscore. Function prototypes go before main and function definitions go after `main`.

Write a driver program that does the following (assumes optional heapsort):

```
Repeat until user is ready to quit
   Input the size of the array from the user
   Dynamically allocate an array of integers of the size given
(+ 1 since element 0 is reserved)
   Populate the array with random numbers in the range 0 thru
999
   Call function displayTree
   Call function displayArray
   Input choice of either [h]eapify or heap[s]ort from the user
   If choice is heapify
        Call function heapify
   Else
        Call function heapsort
   EndIf
   Call function displayTree
   Call function displayArray
   Free the array
```

```
End repeat
```

Remove the logic for heapsort if you do not wish to do the extra credit.

Deliverables

For this assignment you will submit a single program file (.cpp) with your last name appended preceded by an underscore.

Example: heapdriver_doe.cpp