Assignment

smccd.mrooms.net/mod/assign/view.php

Optional Assignment # 5

Heap Sort

Please find our fifth *optional* bonus lab assignment posted. Only one submission allowed. This assignment topic covers sorting techniques. **Note:**There is no late submission option available on this lab. This is an optional only assignment to be used to shore up a low lab score for a previously submitted lab assignment.

This bonus assignment is worth 10 points. Don't worry about the fact that it may say "1 point possible." Do include a statement in your submission file header to let me know what lab assignment you'd like this optional one to replace.

Make sure you have read and understood

- ***lesson modules weeks 12 - 14***
- ***chapters 11-13***
- ***Coding Style Guidelines (module week 1)***

before submitting this assignment. Hand in only one program, please.

Background:

As we have seen, the heap is a data structure with a very special feature of always positing the priority element at the top of the heap. Because of the order property of heaps, we can take advantage of this situation by using a heap to help us sort. Additionally Heap Sort is one of the best sorting methods being in-place and with no quadratic worst-case scenarios.

Objective:

Design and implement a heap sort solution that sorts the items in an array into ascending order.

Requirements:

Define a heap and develop a set of operations for creating and manipulating a heap that satisfies a heap sort implementation.

Test Run Requirements: Provide a copy of your run of the test driver shown below to trace the heap sort as it sorts the following arrays into ascending order. Use multi-line comment delimiters to encase your run so that your solution will compile in the grader test bed.

```
int main()

{

  string a[6] = {"D", "B", "A", "C", "F", "E"};
  heapSort(a, 6);
  for (int i = 0; i < 6; i++)
     cout << a[i] << " ";
  cout << " Sorted array" << endl;
  string b[6] = {"25", "30", "20", "80", "40", "60"};
  heapSort(b, 6);
  for (int i = 0; i < 6; i++)
     cout << b[i] << " ";
  cout << " Sorted array" << endl;
}  // end main


/* Example Test Run Output Display
 D B A C F E  Original array
 F D E C B A  Initial rebuild to form a heap
 A D E C B F  After swapping A and F
 E D A C B F  rebuild(0, anArray, 5)
 B D A C E F  After swapping B and E
 D C A B E F  rebuild(0, anArray, 4)
 B C A D E F  After swapping B and D
 C B A D E F  rebuild(0, anArray, 3)
 A B C D E F  After swapping A and C
 B A C D E F  rebuild(0, anArray, 2)
 A B C D E F  After swapping A and B
 A B C D E F  Sorted array
```

```
25 30 20 80 40 60  Original array

80 40 60 30 25 20  Initial rebuild to form a heap

20 40 60 30 25 80  After swapping 20 and 80

60 40 20 30 25 80  rebuild(0, anArray, 5)

25 40 20 30 60 80  After swapping 25 and 60

40 30 20 25 60 80  rebuild(0, anArray, 4)

25 30 20 40 60 80  After swapping 25 and 40

30 25 20 40 60 80  rebuild(0, anArray, 3)

20 25 30 40 60 80  After swapping 20 and 30

25 20 30 40 60 80  rebuild(0, anArray, 2)

20 25 30 40 60 80  After swapping 20 and 25

20 25 30 40 60 80  Sorted array

*/
```

Notes: *Optional:* Your solution can be in one file **heapSort.cpp** to include the heap data structure, heap operations, test driver and commented run display *or* in separate files. If you choose to submit your solution in separate files still include your test run output in your driver file (as a multi-line comment /* …. */).

Grading Criteria:

Heap Sort is correctly defined and implemented.

Program compiles and runs.

The assignment spec test driver is included to satisfy the test run demonstration.