

CSE 332: Data Structures and Parallelism

Exercises (Heaps)

Directions: Submit your solutions using <https://grinch.cs.washington.edu/cse332/heaps>. Your score on Canvas will automatically update. You should log on to this tool with your UWNetID so your grades can be submitted to canvas. Here are a few pointers about this cool tool:

- Clicking "Check Heap" will give you feedback on whether what you have drawn obeys the heap ordering property.
- "Copy Prev" will copy your diagram from the previous step into the current step - no need to redraw a diagram from scratch each time!
- Backspace will delete the values in a node and then the node itself.
- As you add new nodes, more spaces will be added to the array AFTER the node has been created.
- Also note that "Check Heap" is checking that the heap is a valid heap. It is not checking if you have typed in the values that the question is asking for. (e.g. if I mean to type 10 in a node and I type 110 instead, but 110 still makes a valid heap, it won't tell me about my mistake when I hit "Check Heap".) When I finally try to submit my answer to the problem it will check the values and tell me if I made a mistake and which insert you made a mistake on. Thus this program can be used to check heaps generally!
- Also note that clicking "Check Heap" will give you feedback on the node diagram you have drawn, NOT on the array. You will not be given feedback on your array, but it WILL be graded. So check it yourself! Check to be sure that your array is correct after each insert as well as your final array. You can lose all points on the question if your array is incorrect at any point in the process! Start filling your array at location 0. Note that you do have multiple tries to submit, and your grade is automatically entered in canvas. So if you got a score of 0 in canvas, you can still go back and fix your array/heap and resubmit if you have more tries remaining. We hope you like using this neat tool!

EX03. Heap Me Once, Shame On You! (20 points)

This problem will give you some practice with the basic operations on binary min heaps. For problems asking you to give an array, you should begin filling the array at index 0. That is, the first element in the heap should be in index zero, we will NOT be leaving that location empty or using it to store the size of the heap. You will submit the answers to this exercise using the tool above, you will not submit them on paper/scanned!

(a) [10 Points] Show the result of inserting

11, 17, 3, 9, 5, 6, 14, 2, 13, 8, 1

one at a time into an originally empty binary min heap.

You should show the heap after each insertion. Make sure to give the array after each step as well.

(b) [5 Points] Show the result of running Floyd's buildHeap on the following array:

[11, 17, 3, 9, 5, 6, 14, 2, 13, 8, 1]

You should show the resulting heap and array after the method finishes.

(c) [5 Points] Perform **two** deleteMin operations on the resulting binary min heap from part (b). Show the binary min heaps that result from each deletion.