

3. [16 points] For each of the following implementations of the ADT Set, state the **worst-case** run-time complexity of the given operations for a set with n elements. You may assume a pointer-based implementation for both lists and heaps.

	add	rem_min	has	min
Unsorted List	$\mathcal{O}[\text{_____}]$	$\mathcal{O}[\text{_____}]$	$\mathcal{O}[\text{_____}]$	$\mathcal{O}[\text{_____}]$
Sorted List	$\mathcal{O}[\text{_____}]$	$\mathcal{O}[\text{_____}]$	$\mathcal{O}[\text{_____}]$	$\mathcal{O}[\text{_____}]$
Min Heap	$\mathcal{O}[\text{_____}]$	$\mathcal{O}[\text{_____}]$	$\mathcal{O}[\text{_____}]$	$\mathcal{O}[\text{_____}]$
Max Heap	$\mathcal{O}[\text{_____}]$	$\mathcal{O}[\text{_____}]$	$\mathcal{O}[\text{_____}]$	$\mathcal{O}[\text{_____}]$

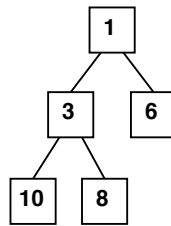
4. [5 points] Show the **final** result of inserting $\{10, 1, 6, 8, 3, 7, 11, 2\}$, one at a time, in an initially empty binary **minimum** heap. Draw the heap as a rooted tree, with the root at the top. You should **not** draw every step, but you **should** indicate which insertions initially violate the heap order. Use **only** the space provided immediately below.

Answer Key for Exam A

The questions below are **SHORT ANSWER** questions. Please write in a **LEGIBLE** way or your answers will **NOT** be graded. There are 4 questions. No negative points are given for wrong answers. Good luck!

1. [2 points] Explain the structural property of a binary heap. You may draw a picture if that helps you, but a picture alone is not a complete explanation. Use **only** the space provided immediately below.

Answer: The structural property of a binary heap is the same as that of a complete binary tree, namely, every level is complete except possibly for the deepest one. In the event that the deepest level is incomplete, it must be filled from left to right with no missing nodes in between. See the figure below for an example of a complete binary tree (which, since it also satisfies the heap property, is also a binary heap).



2. [2 points] Explain the ordering property of a binary **minimum** heap. Use **only** the space provided immediately below.

Answer: The ordering property of a binary minimum heap requires that the value stored at a node be less than or equal to the values stored in each of its immediate children. See the figure above for an example.

3. [16 points] For each of the following implementations of the ADT **Set**, state the **worst-case** run-time complexity of the given operations for a set with n elements. You may assume a pointer-based implementation for both lists and heaps.

Answer:

	add	rem_min	has	min
Unsorted List	$\mathcal{O}[1]$	$\mathcal{O}[n]$	$\mathcal{O}[n]$	$\mathcal{O}[n]$
Sorted List	$\mathcal{O}[n]$	$\mathcal{O}[1]$	$\mathcal{O}[n]$	$\mathcal{O}[1]$
Min Heap	$\mathcal{O}[\log n]$	$\mathcal{O}[\log n]$	$\mathcal{O}[n]$	$\mathcal{O}[1]$
Max Heap	$\mathcal{O}[\log n]$	$\mathcal{O}[n]$	$\mathcal{O}[n]$	$\mathcal{O}[n]$

4. [5 points] Show the **final** result of inserting $\{10, 1, 6, 8, 3, 7, 11, 2\}$, one at a time, in an initially empty binary **minimum** heap. Draw the heap as a rooted tree, with the root at the top. You should **not** draw every step, but you **should** indicate which insertions initially violate the heap order. Use **only** the space provided immediately below.

Answer: Each thick box represents one insertion. The shaded entries show insertion steps where the heap order is violated and needs to be fixed. The insertion order is left-to-right, top-to-bottom.

