This will be an exam over Data Structures; programming will be minimized. Here are the structures we have looked at since the first exam:

- Binary Search Trees
- AVL trees
- Heaps and Priority Queues
- Hashing, Hash Tables, and Hash Maps
- Tries

We have also talked about the $O( n\cdot \log(n) )$ sorting algorithms:

- MergeSort
- QuickSort
- HeapSort

and why $O( n\cdot \log(n) )$ is the best possible worst-case running time for a sorting algorithm that sorts by comparing data values.

For each data structure you should know how it is implemented, how it works, what is is good for, and a Big-O estimate of running times for its algorithms. I will expect you to know details about how the structures work. For example, I could give you a picture of an AVL tree and ask what would happen if we insert another value that makes it unbalanced – you need to know how to rotate it back into balance. You should know all of the structures with this level of detail.

Although