

Problem 1: Design 3 algorithms based on binary min heaps that find the k th smallest # out of a set of n #'s in time:

- a) $O(n \log k)$
- b) $O(n + k \log n)$
- c) $O(n + k \log k)$

Note: For all problems, an array will be used to represent the heap.

a) For part a, we initially allocate an array of size k for our heap with all the elements initialized to $-\infty$ (according to CLRS, a heap containing all duplicate keys can still be classified as a valid min-heap). Then, we will make a pass through our array A from $A[1 \dots \text{length}[A]]$. For each A_i where $i \in \{1 \dots \text{length}[A]\}$, check if $-A_i$ is greater than the minimum of the heap. If such is the case, call remove on the heap, and insert $-A_i$ into the heap. We apply this operation for $\text{length}[A]$ iterations. Once we've scanned through all the elements in A , the k th smallest element should be $-1 * \text{smallest}(\text{heap})$.

Correctness for a: