1. Consider an array of size nine with the numbers in the following order

30, 10, 20, 50, 70, 80, 90, 60, 40.

(a) Phase 1: Form the heap. Show the heap as a tree. Show the heap as an array. Exactly how many comparisons did heap creation use?

(b) Phase 2: Start with the heap created in Part (a). Show the array after each element sifts down after heap creation. How many comparisons does each sift use? What is the total number of comparisons excluding heap creation?

2. Assume you are given a min-heap of size $n$ (so the smallest element is on top) stored in an array in the standard way, and you are also given a value $x$. Design an algorithm to determine whether the $k$th smallest element in the heap is less than or equal to $x$. The worst-case running time of your algorithm should be $O(k)$, independent of the size of the heap. Notice that you do not have to find the $k$th smallest element; you need only determine its relationship to $x$.

(a) Explain your algorithm (briefly and clearly) in English. You do not need to give code.

(b) How many comparisons does your algorithm use in the worst case? Be as exact as possible.