Problem 1. Consider an array of size eight with the numbers in the following order 40, 20, 80, 60, 30, 10, 70, 50.

(a) What is the array after heap formation? Make sure to form the heap bottom up as done in class. How many comparisons does the algorithm use?

(b) Show the array after each element sifts down during the remainder of heapsort, and state how many comparisons each sift takes. What is the total number of comparisons for the remainder of heapsort (i.e., the sum of the comparisons for all of the sifts)?

Problem 2. A $d$-ary heap is like a binary heap, but instead of two children, nodes have $d$ children.

(a) How would you represent a $d$-ary heap in an array?

(b) What is the index of leftmost child of the node stored at index $i$?

(c) What is the index of parent of the node stored at index $i$?

(d) What is the height of a $d$-ary heap of $n$ elements in terms of $n$ and $d$.

(e) Explain loosely (but clearly) how to extract the maximum element from the $d$-ary heap (and restore the heap). How many comparisons does it require?

(f) How many comparisons does it take to sort? Just get the high order term exactly, but show your calculations.

(e) What value(s) of $d$ are optimal? Justify your answer.