Problem 1. For this problem (as always) we will use Williams’s original heapsort algorithm with Floyd’s modification for heap creation. Consider an array of size eight with the numbers in the following order 20, 40, 60, 80, 10, 30, 50, 70.

(a) What is the array after heap creation? How many comparisons does heap creation use?
(b) Starting from the heap, show the array after each sift operation and state how many comparisons each sift takes. How many comparisons does the algorithm use in total after heap creation?

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 height of a $d$-ary heap of $n$ elements in terms of $n$ and $d$. Just get the exact high order term.
(c) 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?
(d) 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.

Problem 3. Recall the Bubble/Merge sorting algorithm from Homework 2. Here is a skeleton:

```plaintext
procedure BubbleMergeSort(A,n)
    k ← (√8n + 1)/2
    BubbleSortPhase(A,k)
    MergePhase(A,k)
end procedure
```

(a) Write the pseudo code for a recursive version of the Bubble Sort phase (using $k$ as your parameter).
(b) Give a recurrence for the exact number of comparisons the Bubble Sort phase uses (as a function of $k$).
(c) Write the pseudo code for a recursive version of the Merge phase (using $k$ as your parameter).
(d) Give a recurrence for the exact number of comparisons the Merge phase uses (as a function of $k$).