1. (a) Describe briefly but clearly how to efficiently merge four sorted lists, each of size $m$, into a single sorted list. Try to minimize the number of comparisons.
   (b) Exactly how many comparisons does your algorithm use in the worst case?

2. Consider a merge-sort-like algorithm in which the list of size $n$ is split into four equal-sized lists, each list is sorted recursively, and the four lists are then merged into a single sorted list.
   (a) Write pseudo code for the algorithm. You may assume a four-way merge algorithm is available (as described in Problem 1). Your algorithm should work even if $n$ is not nice.
   (b) Write a recurrence for exactly how many comparisons your algorithm uses, assuming $n$ is nice (i.e., $n$ is a power of 4).
   (c) Solve the recurrence exactly using the tree method.
   (d) How does the number of comparisons compare with standard merge sort?