Algorithms. Fall 2005. Closed book.

1. (*i*) Give an $O(k^2 \lg k)$ algorithm to merge k sorted lists, each having k elements, into a single sorted list with k^2 elements.

   15 pts

   (*ii*) Assume that you have an $O(k^2 \lg k)$ algorithm M to merge k sorted lists, each having k elements, into a single sorted list with k^2 elements. Consider the following idea to sort a list of n elements: Split the n elements into $\sqrt{n}$ lists of $\sqrt{n}$ elements each, sort each group recursively and then merge the $\sqrt{n}$ sorted lists into one sorted list of size n using the algorithm M.

   Give pseudocode for an algorithm that uses the above idea to sort an array of n elements. Derive a recurrence relation for the running time of your algorithm and solve this recurrence relation to obtain its worst-case running time.

   30 pts

   (*iii*) Is it possible to devise a merge procedure that merges k sorted lists, each having k elements, into a single sorted list of size k^2 in time $O(k^2)$. Provide a proof for your answer.

   20 pts

2. Consider the following "change-making" problem: Given a set of n denominations of coins $a_1 = 1 < a_2 < a_3 \ldots < a_n$, an infinite supply of coins of each denomination and a value A, determine how to make the value exactly A using the **minimum** number of coins. For example, if we have four coin values 1 (penny), 5(nickel), 10(dime) and 25(quarter) and we have to make a value 82, the best way to do it is to use 3 quarters, 1 nickel and 2 pennies which uses six coins.

   A natural greedy algorithm for making exact change A is the following (here S is a multiset or "bag" which is assumed to be empty in the beginning):

algorithm MAKE-CHANGE(A, S)
  if $(A = 0)$ **return** S;
  else
    pick a coin with the largest denomination a_i such that $a_i \leq A$
    and add it to the bag S;
    MAKE-CHANGE$(A - a_i, S)$;

(i) Suppose we have the coin denominations which satisfy the following conditions: $a_1 = 1$ and for $i = 1, 2, \ldots, n-1$ $[a_{i+1} = 3 * a_i]$. Does the algorithm MAKE-CHANGE always work correctly in this case. Provide a proof for your answer.

25 pts

(ii) Suppose we have the coin denominations which satisfy the following conditions: $a_1 = 1$ and for $i = 1, 2, \ldots, n-1$ $[a_{i+1} \geq 3 * a_i]$. Does the algorithm MAKE-CHANGE always work correctly in this case. Provide a proof for your answer.

10 pts