CSC 364H	**Solutions to Problem Set 1**	**Summer 99**

Total marks: 25

[0] 1. b), c), d), g) and h).

[4] 2. (a) Let T_1 by any min cost ST of G. We show that if T_2 is the output of Kruskal's algorithm on G, then $T_2 \subseteq T_1$. To this end we show the following loop invariant: $T \subseteq T_1$.

Initially $T = \emptyset$, so $T \subseteq T_1$. Suppose now that $T \subseteq T_1$. We show that this continues to be true after one more execution of the loop.

If e_i is rejected, then T remains unchanged, and so $T \subseteq T_1$. Otherwise, we must show that $T \cup \{e_i\} \subseteq T_1$. So suppose, by way of contradiction, that $e_i \notin T_1$. Let T_2 be the result of Kruskal's algorithm. By the Exchange Lemma there is an edge e_j such that $e_j \in T_1 - T_2$, and $T_3 = T_1 \cup \{e_i\} - \{e_j\}$ is a ST.

$i \neq j$ since $e_i \notin T_1$.

Suppose $j < i$. Then e_j must have been rejected, so it would have formed a cycle in T, and so it would have also formed a cycle in T_1, hence $e_j \notin T_1$, contradiction.

Suppose $i < j$. Then $c(e_i) < c(e_j)$, and it would follow that $c(T_3) < c(T_1)$, contradiction.

Thus, there can be no such e_i.

Therefore $T_2 \subseteq T_1$, and so $T_2 = T_1$, so T_2 is in fact the unique min cost ST for G.

[2] (b) G has a unique min cost ST in this case as well. The proof is the same as in the previous case, but this time when considering the case $i < j$, we have to deal with the possibility that $c(e_i) = c(e_j)$, i.e. that $i = 1$ and $j = 2$. But if $e_1 \notin T_1$, then if we add e_1 to T_1 we get a cycle, and a cycle must contain at least 3 edges, so it contains some e such that $e \neq e_2$, so if we replace e by e_1, we get a ST of a smaller cost, contradiction.

[2] (c) In this case G doesn't have a unique min cost ST: consider the case where G has 3 vertices and edges of the same cost between them. Then any two edges form a min cost ST for G, so G has 3 min cost spanning trees.

[5] 3. *Basis Case*: Initially, L is empty, so any optimal solution extends L. Hence L is promising.
Induction Step: Assume that L is promising, and show that L continues to be promising after one more execution of the loop:

Suppose L is promising, and $s < N$. Let L' be the list that extends L to the optimal solution, i.e. $L, L' = L_{\text{opt}}$. Let x be the largest item in C such that $s + x \leq N$.

case (a) $x \in L'$. Then $L' = x, L''$, so that L, x can be extended to the optimal solution L_{opt} by L''.

case (b) $x \notin L'$. We show that this case is not possible. To this end we prove the following claim:

Claim: If $x \notin L'$, then there is a sublist L_0 of L' such that $x =$ sum of elements in L_0.

Proof of Claim: Let B be the smallest number such that $B \geq x$, and some sublist of L' sums to B. Let this sublist be $\{e_1, e_2, \ldots, e_l\}$, where $e_i \leq e_{i+1}$ (i.e. the elements are in non-decreasing order). Let $D = x - (e_2 + \ldots + e_l)$. By definition of B we know that $D > 0$.

Each of the numbers $x, e_2, \ldots, e_l$ is divisible by e_1 (here is where we use the fact that our denominations are of the form $\{1, p, p^2, \ldots, p^n\}$), so $D \geq e_1$. On the other hand $x \leq e_1 + e_2 + \ldots + e_l$, so we also know that $D \leq e_1$, so in fact $D = e_1$. Therefore $x = e_1 + e_2 + \ldots + e_l$, and we are done.

Therefore, $\{e_1, e_2, \ldots, e_l\}$ can be replaced by the single coin x. If $l = 1$, then $x = e_1 \in L'$, which is a contradiction. If $l > 1$, then $L, x, L' - \{e_1, e_2, \ldots, e_l\}$ sums up to N, but it has less coins than $L, L' = L_{\text{opt}}$ which contradicts the induction hypothesis. Thus case (b) is not possible.

[2] 4. (a) $R(j)$ = length of the longest subsequence (where any two consecutive members differ by at most 1) which ends in a_j. Then, the answer is $L = \max_{1 \leq j \leq n} R(j)$.

[2] (b) Let $R(1) = 1$, and for $j > 1$,

$$R(j) = \begin{cases} 1 & \text{if } |a_j - a_i| > 1 \text{ for all } i,\ 1 \leq i < j. \\ 1 + \max_{1 \leq i < j}\{R(i) \,|\, |a_j - a_i| \leq 1\} & \text{otherwise} \end{cases}$$

[2] (c) Write the algorithm:

$R(1) \leftarrow 1$
for $j : 2..d$
  max $\leftarrow 0$
  for $i : 1..j-1$
    if $(R(i) >$ max & $|a_i - a_j| \leq 1)$ then
      max $\leftarrow R(i)$
    end if
  end for
  $R(j) \leftarrow$ max $+1$
end for

[2] 5. (a) Assume $n \geq 2$, and for $2 \leq i \leq n$, define $D(i)$ = length of shortest path from 1 to i.

[2] (b) Let $D(2) = C(1,2)$, and for $i > 2$,

$$D(i) = \min\{C(1,i), \min_{2 \leq j < i}\{D(j) + C(j,i)\}\}$$

[2] (c) Write the algorithm:

$D(2) \leftarrow C(1,2)$
for $i : 3..n$
  $D(i) \leftarrow C(1,i)$
  for $j : 2..(i-1)$
    if $D(j) + C(j,i) < D(i)$ then
      $D(i) \leftarrow D(j) + C(j,i)$
    end if
  end for
end for