CS 341: Algorithms @ Waterloo **Last Updated:** Mar 3, 2016
Homework 4 **Due Date: Mar 17, 2016, 11:59pm**

The full mark is 50. This homework is counted 5% of the course grade.

---

1. **Programming Problem: Longest Common Subsequence** (10 marks)

   Input: Three strings.

   Output: The longest common subsequence of the three strings.

   The input is from stdin and the output is to stdout. The input will be three lines, each for an input string. You can assume the input strings consists of only English letters, numbers, and symbols. And the length of each string is at most 300. The output contains only one line that is a longest common subsequence of the three strings (if there are multiple longest common subsequences, just print anyone).

2. **Written Problem: Facility Location** (10 marks)

   Suppose there is a new town where everyone lives in a long street. The government has the budget to build k hospitals. There are m possible locations and the government would like to choose a subset of k locations so as to minimize the total distance to the people. You are asked to solve the following problem.

   Input: n nonnegative numbers $a_1 \leq a_2 \leq \ldots \leq a_n$ representing the locations of the people, m nonnegative numbers $p_1 \leq p_2 \leq \ldots \leq p_m$ representing the possible locations of the hospitals, and an integer $1 \leq k \leq m$. We assume that $m \leq n$.

   Output: The minimum total distance to the people. Suppose $S = \{l_1, \ldots, l_k\} \subseteq \{1, \ldots, m\}$ is a subset of k hospitals. We assume that every person will go to the nearest hospital, and so we define $\text{dist}(i, S) = \min_{j \in S} |a_i - p_j|$ for person i. The objective is to choose a subset S of k hospitals so as to minimize the total distance defined as $\sum_{i=1}^{n} \text{dist}(i, S)$. You just need to output one number, the minimum total distance. There is no need to output an optimal subset S.

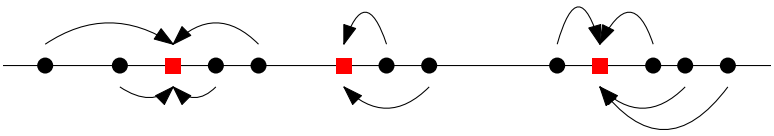

Figure 1: The black circles are people. The red squares are hospitals. Each person goes to the nearest hospital and this is how the distances are defined. In this example $k = 3$.

   Design an efficient algorithm to solve this problem. Prove the correctness and analyze the time complexity.

3. **Written Problem: Coin Game** (10 marks)

Suppose there are n coins of values $v_1, \ldots, v_n$ on a row where n is even. There are two players in the game, player 1 and player 2, and they take turns to pick coins with the following rule. In each turn, a player can pick either the leftmost coin or the rightmost coin from the row, removes it from the row, and receives the value of the coin. Assume both players play optimally. Design an efficient algorithm to compute the maximum total value that player 1 can earn (player 1 makes the first move). Prove the correctness and analyze the time complexity.

4. **Written Problem: Covering Set** (10 marks)

Given an undirected graph $G = (V, E)$, a subset $S \subseteq V$ is a covering set if for every vertex $v \in V - S$ there exists $u \in S$ such that $uv \in E$ (i.e. every vertex $v \in V - S$ has a neighbor in S). We are interested in finding a covering set of minimum total weight, but this problem is NP-hard in general graphs. Show that this problem is easy in trees.

Input: A tree $T = (V, E)$ in the adjacency list representation, a weight w_v for each vertex $v \in V$.

Output: A covering set $S \subseteq V$ that minimizes the total weight $\sum_{v \in S} w_v$.

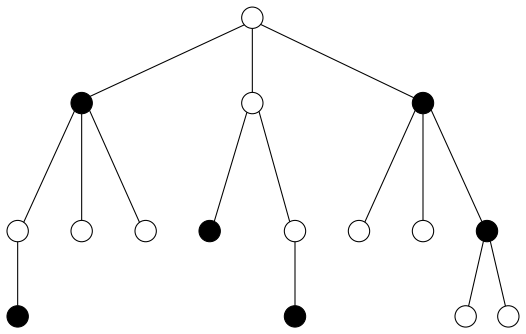

Figure 2: The black vertices form a minimum covering set assuming all weights are one.

Design an efficient algorithm to solve this problem. Prove the correctness and analyze the time complexity.

5. **Written Problem: Trading** (10 marks)

Suppose somehow we could use 1 unit of metal to trade for 2 units of wood, and 1 unit of wood to trade for 0.4 unit of glass, and 1 unit of glass to trade for 1.5 units of metal. Then we can start with 1 unit of metal, and end up with $1 \times 2 \times 0.4 \times 1.5 = 1.2$ units of metal, and make profit by just trading. Help your company to solve the following problem.

Input: n items $a_1, \ldots, a_n$ and an $n \times n$ table T of trading rates, such that one unit of item a_i can trade for $T[i, j]$ units of item a_j, where $T[i, j] \geq 0$ for $1 \leq i, j \leq n$.

Output: Whether or not there exists a sequence of items $a_{i_1}, \ldots, a_{i_k}$ such that $T[i_1, i_2] \cdot T[i_2, i_3] \cdot \cdots \cdot T[i_{k-1}, i_k] \cdot T[i_k, i_1] > 1$.

Design an efficient algorithm to solve this problem. Prove the correctness and analyze the time complexity.