ECE 606, Fall 2021, Assignment 7
Due: Tuesday, November 2, 11:59pm

Submission: submit your solutions for the written problems to crowdmark. There are no **[python3]** problems in this assignment.

1. Consider the change-making problem from Lecture 7a. The input, as it states there, is $\langle a, c_0, c_1, \ldots, c_{k-1} \rangle$ with the properties stated there.

   (a) MinCoins-DP outputs the minimum total number of coins only, not the number of coins of each value. Change the pseudo-code for MinCoins-DP into a new algorithm, call it MinCoinsTuple, whose output is a k-tuple $\langle m_0, m_1, \ldots, m_{k-1} \rangle$, which is the number of coins of each value that add up to the input amount a, such that the total number of coins is minimized. For example, on input $\langle 6, 1, 3, 4 \rangle$, MinCoinsTuple should output $\langle 0, 2, 0 \rangle$, and on input $\langle 100, 1, 12, 28, 114 \rangle$, MinCoinsTuple should output $\langle 0, 6, 1, 0 \rangle$.

   Credit: your colleague Zile Deng for pointing out that my earlier example was broken.

   (b) Suppose you are given an *oracle*, MinTotalCoins, that given input $\langle a, c_0, \ldots, c_{k-1} \rangle$ outputs the correct minimum total number of coins m that corresponds to a. An oracle is a tamper-proof blackbox that outputs the correct answer in time $\Theta(1)$.

   Write new pseudo-code for MinCoinsTuple, but this time guaranteeing that it runs in polynomial-time, that outputs a k-tuple as stated in sub-problem (a) above.

   (Hint: binary search on a with repeated calls to the oracle.)

   (c) This sub-problem pertains to the original problem of handing out the minimum total number of coins and not sub-problems (a) and (b) above. Returning to that original problem, suppose, in addition to coin-values $c_0, \ldots, c_{k-1}$ where $c_0 = 1$ and $c_i < c_{i+1}$ for all $i = 0, 1, \ldots, k-2$, we are given a number of coins of each value, $n_0, n_1, \ldots, n_{k-1}$ that we have available. That is, our input is a non-negative integer a, and k pairs $\langle c_0, n_0 \rangle, \langle c_1, n_1 \rangle, \ldots, \langle c_{k-1}, n_{k-1} \rangle$.

   Rewrite the recurrence from Lecture 7a, but this time for $M[a, n_0, n_1, \ldots, n_{k-1}]$ which is the minimum total number of coins for amount a given that we have n_i number of coins of value c_i available to be handed out. Use the mnemonic "∞" for some amount a for which we are unable to make change.

   For example, on input $\langle 6, \langle 1, 5 \rangle, \langle 3, 3 \rangle, \langle 4, 2 \rangle \rangle$, the correct output is 2 because the minimum number of coins we would hand out is $0 \times 1 + 2 \times 3 + 0 \times 4$. On input $\langle 6, \langle 1, 5 \rangle, \langle 3, 1 \rangle, \langle 4, 2 \rangle \rangle$, the correct output is 3 — we no longer have two coins of value 3 to hand out, and the minimum is yielded by $2 \times 1 + 0 \times 3 + 1 \times 4$. On input $\langle 6, \langle 1, 0 \rangle, \langle 3, 1 \rangle, \langle 4, 2 \rangle \rangle$, the output is ∞ because we are unable to hand out any combination of coins that adds up to 6.

   Credit: your colleague Zhixuan Zhang for pointing out a crucial bug in an earlier version.

2. Consider the recurrences for $d_{ij}^{(k)}$ and the corresponding $\pi_{ij}^{(k)}$ in Floyd-Warshall on pdf page 42 and 44 of Lecture 7 of your textbook. Observe that when $d_{ij}^{(k-1)} = d_{ik}^{(k-1)} + d_{kj}^{(k-1)}$,

it does not matter which one of $d_{ij}^{(k-1)}$ or $d_{ik}^{(k-1)} + d_{kj}^{(k-1)}$ we choose for $d_{ij}^{(k)}$ because they are the same value. However, it turns out that this either-or option does not exist for $\pi_{ij}^{(k)}$. Specifically, show via counterexample that the following recurrence for $\pi_{ij}^{(k)}$ is not correct.

$$\pi_{ij}^{(k)} = \begin{cases} \pi_{ij}^{(k-1)} & \text{if } d_{ij}^{(k-1)} < d_{ik}^{(k-1)} + d_{kj}^{(k-1)} \\ \pi_{kj}^{(k-1)} & \text{if } d_{ij}^{(k-1)} \geq d_{ik}^{(k-1)} + d_{kj}^{(k-1)} \end{cases}$$

You will want to keep in mind the properties we adopt for the mnemonic "∞": (i) for every real number a, $a \neq \infty$ and $a + \infty = \infty$, and, (ii) if $a = \infty$ and $b = \infty$, then $a = b$ and $a + b = \infty$.