NOTE: This file contains sample solutions to the quiz together with the marking scheme and comments for each question. Please read the solutions and the marking schemes and comments carefully. Make sure that you understand why the solutions given here are correct, that you understand the mistakes that you made (if any), and that you understand *why* your mistakes were mistakes.

Remember that although you may not agree completely with the marking scheme given here, it was followed the same way for all students. We will remark your quiz only if you clearly demonstrate that the marking scheme was not followed correctly.

For all remarking requests, please submit your request **in writing** directly to your instructor. For all other questions, please don't hesitate to ask your instructor during office hours or by e-mail.

GENERAL MARKING SCHEME:

- ***A****: All Correct*, except maybe for very few minor errors.
- ***B****: Mostly Correct*, but with a few serious errors, or many small errors.
- ***C****: Mostly Incorrect*, but with a few important elements, or many small elements, done correctly.
- ***10%****: Completely Blank*, or clearly crossed out.
- ***D****: All Incorrect*, except maybe for very few minor elements done correctly.

MARKER'S COMMENTS:

- Some students gave a greedy algorithm, which does not work.
- "I don't know" was treated the same as a blank answer.
- Many students used a one-dimensional array and iterated over the denominations, for each array entry. This is slightly less efficient, but otherwise fine.
- Many students seem confused about the difference between finding an actual solution (the question was *not* asking for this), and using the recursive structure of the problem to define a recurrence relation for the array values.

INSTRUCTOR'S COMMENTS:

- This quiz took place early in the "dynamic programming" section of the course, so it makes sense that students found it a bit more difficult than the rest of the quizzes so far. However, I am concerned that the performance was so poor.
  I plan to take a closer look at the relationship between the tutorial materials and the quizzes, as well as how the tutorials are delivered.

1. Consider the following problem of "making change", given an unlimited supply of coins with various values:

Input: A positive integer *amount* A and positive integer *denominations* $d_1 < d_2 < \cdots < d_m$. (For example, using pennies, dimes, and quarters, we have $d_1 = 1$, $d_2 = 10$, and $d_3 = 25$.)

Output: A list of coins $c_0, c_1, \ldots, c_{n-1}$ where each $c_i \in \{d_1, d_2, \ldots, d_m\}$ and *repeated coins are allowed*, such that $c_0 + c_1 + \cdots + c_{n-1} = A$ and n is as small as possible. (For example, making change for amount 30 is done with $n = 3$ and $c_0 = 10$, $c_1 = 10$, $c_2 = 10$.)

Define an array that could be used to solve the general problem of making change using a dynamic programming algorithm. (Your solution should work for **any** denominations.) Then, give a recurrence relation for the values in your array, including a brief justification that your recurrence is correct. *Do not write any algorithm* — we want just the recurrence along with a brief English explanation.

Step 0: Describe the recursive structure of sub-problems.

In every optimal solution, either there is at least one coin worth d_m (in which case the rest of the solution is optimal for amount $A - d_m$), or there is no coin worth d_m (in which case the rest of the solution is optimal for denominations $d_1, d_2, \ldots, d_{m-1}$).

Step 1: Define an array that stores optimal values for arbitrary sub-problems.

Let $N[a, j]$ represent the minimum number of coins required to make change for amount a using denominations $d_1, \ldots, d_j$, for $0 \leqslant a \leqslant A$ and $0 \leqslant j \leqslant m$.

Step 2: Give a recurrence relation for the array values.

$N[a, 0] = \infty$ for $a = 0, 1, \ldots, A$.

$N[0, j] = 0$ for $j = 1, 2, \ldots, m$.

$N[a, j] = N[a, j-1]$ if $a < d_j$, for $1 \leqslant a \leqslant A$ and $1 \leqslant j \leqslant m$.

$N[a, j] = \min\{N[a, j-1], 1 + N[a - d_j, j]\}$ if $a \geqslant d_j$, for $1 \leqslant a \leqslant A$ and $1 \leqslant j \leqslant m$.