1. Consider the change making problem where we would like to make change for \( n \) cents using the fewest number of coins. Suppose we have \( k \) coins, each worth \( d_1, d_2, \ldots, d_k \) cents. (30 points)

- Describe a greedy method that works for the case when our coins are the familiar, American denominations; i.e., 1 cent, 5 cents, 10 cents, and 25 cents. Prove that this method is optimal.
- Now suppose that our denominations are less familiar. For example, suppose we have coins worth 1 cent, 10 cents, 14 cents, and 23 cents, and our \( n \) is 28. Does the same method work? Why or why not?
- Describe a dynamic program that can solve the problem in \( O(nk) \) time for any denominations. Prove that your dynamic program is optimal.

2. Given two strings, \( C_1 \) and \( C_2 \), we wish to compute the number of times that \( C_1 \) appears as a subsequence of \( C_2 \). For example, if \( C_1 = 'ab' \) and \( C_2 = 'aacccb' \), \( C_1 \) appears in as two distinct subsequences of \( C_2 \) (one in 'aacccb' and the other 'acccb'). Explain how we can use a Dynamic Program similar to the one we used to compute the Longest Common Subsequence to solve this problem. (20 points)

3. Describe how we can implement a queue using two stacks with both Enqueue and Dequeue operating in \( O(1) \) amortized time. (20 points)

4. CLRS Problem 17-2 (30 points)