Programming Assignment 6: Graph Algorithms & Dynamic Programming

Objective

Implement algorithms for shortest path and dynamic programming challenges to solve real-world-inspired problems, understand the complexities of each, and explore different problem-solving approaches in algorithm design.

Part 1: Shortest Path Algorithms

Problem 1: Dijkstra's Algorithm

You are given a network of cities with roads connecting some of them. Each road has a non-negative distance. You need to find the shortest path from a starting city to every other city.

- **Input**: A list of edges in the format (u, v, w), where u and v are cities, and w is the distance between them. The total number of cities n and a starting city s.
- **Output**: An array of minimum distances from the starting city to each city. If a city is unreachable, mark its distance as ∞.
- **Requirements**: Implement Dijkstra's algorithm using a priority queue.

Problem 2: Bellman-Ford Algorithm

In some networks, distances may include negative values. Use the Bellman-Ford algorithm to find the shortest path from a starting city to all other cities, handling cases where negative cycles exist.

- **Input**: A list of edges in the format (u, v, w), where u and v are cities, and w is the distance (possibly negative). The total number of cities n and a starting city s.
- **Output**: An array of minimum distances from the starting city to each city. If a city is unreachable, mark its distance as ∞. If there is a negative-weight cycle that can be reached from the start city, detect it and return a message indicating the presence of a negative cycle.

Part 2: Dynamic Programming Problems

Problem 3: Longest Common Subsequence (LCS)

Given two strings, find the length of their longest common subsequence (LCS). The LCS of two strings is the longest sequence that appears in both strings in the same order, but not necessarily consecutively.

- **Input**: Two strings X and Y.
- **Output**: The length of the longest common subsequence. A string showing the LCS itself.
- **Requirements**: Implement the dynamic programming approach to solve the LCS problem. Analyze the time complexity of your solution and compare it with other approaches (e.g., recursive solutions).

Problem 4: Rod Cutting

You are given a rod of length n and an array prices where prices$[i]$ represents the price of a rod of length $i + 1$. Determine the maximum obtainable value by cutting up the rod and selling the pieces. Assume you can make cuts at any integer position and can sell all parts for the maximum profit.

- **Input**: An integer n representing the rod's length. An array prices where prices$[i]$ represents the price of a rod of length $i + 1$.
- **Output**: The maximum profit obtainable. The list of lengths to cut to achieve the maximum profit.
- **Requirements**: Implement the dynamic programming solution for the rod cutting problem. Analyze the time complexity.