**Critical Thinking 7: Dijkstra’s Algorithm**

Chioma Chance

Colorado State University Global

CSC506: Design and Analysis of Algorithms

Dr. Banerjee

03/30/2025

**Route Optimization with Dijkstra's Algorithm**

**Introduction**

This critical thinking assignment involved creating a route optimization simulation for a food delivery service using Dijkstra's algorithm. The primary objective was to find the shortest delivery path from a warehouse to a customer destination. The model, though basic, mimics real-world courier navigation using a weighted graph. Each location is represented as a node, and roads with distance values form the edges. While the simulation does not yet include real-time data like traffic or road conditions, it lays the groundwork for a scalable delivery system.

**Purpose of the Program**

The simulation uses Dijkstra’s algorithm to compute the shortest path between the warehouse and delivery location. By evaluating all potential routes and prioritizing the lowest cumulative distances, the algorithm ensures the delivery route is optimal. The output clearly shows both the shortest total distance and the ordered path taken. In this example, the optimal route was: 'Warehouse' → 'B' → 'A' → 'C' → 'Destination' with a shortest distance of 8.

**Development Challenges**

One early hiccup was the lack of feedback when running the code—although the algorithm worked correctly, no output appeared due to missing print() statements. After adding proper print functions, the results became visible in the terminal. Another challenge was reconstructing the path from the destination back to the source. This required storing each node's predecessor during traversal to accurately piece together the shortest path.

**Skills Gained**

Through this assignment, I deepened my understanding of Dijkstra’s algorithm and its underlying mechanics. I became more comfortable using Python’s heapq library to manage priority queues and reinforce how efficient graph traversal can aid real-world applications. Additionally, I developed stronger debugging habits and gained confidence in working with dictionary-based graph representations.

**Conclusion**

Dijkstra’s algorithm is an effective tool for route optimization in delivery applications, especially where real-time efficiency is crucial. This simulation, though basic, successfully demonstrates how the algorithm can be implemented in a practical context. Improvements could include integrating live traffic data, using GPS coordinates for dynamic graph updates, and building user interfaces for route visualization. These upgrades would allow for more realistic courier navigation systems in future iterations.

**References**

Dijkstra, E. W. (1959). A note on two problems in connexion with graphs. *Numerische Mathematik*, 1(1), 269–271.

Python Software Foundation. (n.d.). *heapq — Heap queue algorithm*. <https://docs.python.org/3/library/heapq.html>

GeeksforGeeks. (n.d.). *Dijkstra’s shortest path algorithm*. <https://www.geeksforgeeks.org/dijkstras-shortest-path-algorithm-graph-data-structure/>

**Screenshot of Successful Execution**

A screenshot of a computer program

AI-generated content may be incorrect.