Dijkstra's Algorithm - ChatGPT Explanation

Dijkstra's algorithm is used to find the shortest path from a source node to all other nodes in a weighted graph (with non-negative weights). Here's a simple and clean implementation in Python using a priority queue (heapq):

Python Code: Dijkstra's Algorithm

```
import heapq

def dijkstra(graph, start):
    # Initialize distances from start node to all nodes as infinity
    distances = {node: float('inf') for node in graph}
    distances[start] = 0

    # Priority queue to store (distance, node)
    pq = [(0, start)]

    while pq:
        current_distance, current_node = heapq.heappop(pq)

        # Skip if we found a better path already
        if current_distance > distances[current_node]:
            continue

        # Explore neighbors
        for neighbor, weight in graph[current_node]:
            distance = current_distance + weight
            # If found a shorter path to neighbor
            if distance < distances[neighbor]:
                distances[neighbor] = distance
                heapq.heappush(pq, (distance, neighbor))

    return distances
```

Example Usage:

```
graph = {
    'A': [('B', 1), ('C', 4)],
    'B': [('A', 1), ('C', 2), ('D', 5)],
    'C': [('A', 4), ('B', 2), ('D', 1)],
    'D': [('B', 5), ('C', 1)]
}

start_node = 'A'
shortest_distances = dijkstra(graph, start_node)

for node in shortest_distances:
               print(f"Shortest    distance    from    {start_node}    to    {node}    is
{shortest_distances[node]}")
```

Output:

```
-------
Shortest distance from A to A is 0
Shortest distance from A to B is 1
Shortest distance from A to C is 3
Shortest distance from A to D is 4
```