**Übungsbeispiel: Graphen in Java**

**Aufgabe: Graphen-Traversal und kürzester Weg**

Schreiben Sie eine Java-Klasse „Graph“, die einen ungerichteten gewichteten Graphen repräsentiert. Die Klasse sollte Methoden enthalten, um den Graphen zu erstellen, Traversals durchzuführen und den kürzesten Weg zwischen zwei Knoten zu finden.

Die Klasse sollte folgende Methoden enthalten:

1. „addVertex(int v)“: Fügt einen neuen Knoten zum Graphen hinzu.

2. „addEdge(int v1, int v2, int weight)“: Fügt eine gewichtete Kante zwischen den Knoten „v1“ und „v2“ mit dem angegebenen Gewicht hinzu.

3. „breadthFirstTraversal(int start)“: Führt eine Breitensuche (Breadth-First Search, BFS) ab dem angegebenen Startknoten durch und gibt die besuchten Knoten aus.

4. „depthFirstTraversal(int start)“: Führt eine Tiefensuche (Depth-First Search, DFS) ab dem angegebenen Startknoten durch und gibt die besuchten Knoten aus.

5. „shortestPath(int start, int end)“: Findet den kürzesten Weg zwischen dem Startknoten „start“ und dem Endknoten „end“ mittels Dijkstra-Algorithmus und gibt die kürzeste Distanz sowie den Pfad aus.

**Beispiel:**

Ein Bild, das Text, Elektronik, Screenshot, Display enthält.

Automatisch generierte Beschreibung

- Verwenden Sie geeignete Datenstrukturen zur Speicherung von Knoten, Kanten und deren Gewichte.

- Implementieren Sie die BFS und DFS mithilfe von Schleifen oder rekursiven Funktionen.

- Für den Dijkstra-Algorithmus können Sie eine Prioritätswarteschlange (PriorityQueue) verwenden, um die nächsten zu untersuchenden Knoten nach ihrer aktuellen Distanz zu ordnen.