Metropolitan University of Tirana
Data Structures
Stacking Up: The Art of Combining Data Structures

7.1 Navigation System

You are tasked with implementing a navigation system that computes the shortest path between two cities using **Dijkstra**'s algorithm. Your implementation should use appropriate data structures to represent the **cities** and the **roads** that connect them. You should use Java programming language to implement the solution.

Your implementation should have the following classes:

1. **City**: represents a city
   - `name`: the name of the city.
   - `edges`: a list of Edge objects representing the roads that connect this city to other cities.
2. **Edge**: represents a connection between two cities
   - `dest`: a City object representing the destination city of the edge.
   - `weight`: an integer representing the distance between the two cities.
3. **NavigationSystem**: the main class that contains the graph data structure and the implementation of navigation system
   - `addCity`: adds a new city to the graph.
   - `addConnection`: adds a new road between two cities to the graph.
   - `shortestPath`: computes the shortest path between two cities using Dijkstra's algorithm.

Your implementation should satisfy the following requirements:

- Use a priority queue to keep track of the cities to visit next and a set to keep track of the visited cities.
- Use two maps to keep track of the distance from the start city to each city visited so far and the previous city on the shortest path to each city visited so far.
- Return the shortest distance between the start city and the end city.