task 03

We know the time complexity of Dijkstra algorithm is $O(V^2)$ but when we we priority queue, the time complexity become $O(V+E \log V)$. In this algorithm, priority queque was used so the time complexity would be $O(V+E \log V)$ where V refers to vertices and E refers to edges. So, if there are N places and M roads, the time complexity is ~~O M+N~~ $O(N+M \log N)$. If the number of titans of each road is exactly 1, then the weight on number of titans is negligible and then can be solved using BFS which has the time complexity of $O(M+N)$.

```
def BFS (visited, graph, source, destination)
    Do visited [int (source) - 1] ← 1
    Do queue ← append (source)
    While queue not empty
    Do m ← pop ()
    If m = destination break
    For each neighbor of m in graph

    If visited [int (neighbor) - 1] = 0
        Do visited [int(neighbor) - 1] ← 1
        Do queue ← append (neighbor).
```