Answer to Question no: 3

If there are N places and M roads, it means there are M number of edges and N number of vertices. For task 1 and task 2, the time complexity will be $\boxed{M \log N}$. For both tasks we used Dijkstra Algorithm. For traversal, here we used BFS and to store the shortest path we used min heap. For extracting it takes $O(\log N)$ time in min heap. When we traverse using BFS it takes $O(N+M)$ time. So, the total time complexity will be $O(M \log N)$ for task 1. For same reason, the time complexity of task 2 will be $O(M \log N)$

If the number of Titans in all road becomes 1, then we should not think about the ~~shortest~~ weight at all. For those we just have to care about the shortest path as it will give me less titan.

Pseudocode:

```
Dijkstra(Graph, source):

    distance : [ ]
    visited : [ ]
    for each v in Graph:
        distance[v] ← ∞
    push in min heap (distance[v], v)
        visited → false.
    while minheap != empty
        u ← min heap . extract
    If visited [v]:
        continue.
    for each neighbour in graph:
        if distance[n] > distance[u[i]] + Graph[ ]
            distance[n] = u
            parent[c] = u[i].
```