3. On Leetcode find a problem that can be solved with Bellman-Ford algorithm and solve it.

```
#include<bits/stdc++.h>
using namespace std;


class Solution {
public:
   int networkDelayTime(vector<vector<int>>& times, int N, int K) {
       int MAX_TIME = 101 * 100;
       vector<int> dist(N, MAX_TIME);
       dist[K - 1] = 0;
       for (int i = 0; i < N; ++i)
          for (const auto& time : times) {
             int u = time[0] - 1, v = time[1] - 1, w = time[2];
             dist[v] = min(dist[v], dist[u] + w);
          }

       int max_dist = *max_element(dist.begin(), dist.end());
       return max_dist == MAX_TIME ? -1 : max_dist;
   }
};
```

❖ For Solving :

```
#include <bits/stdc++.h>
using namespace std;

class Solution {
public:
    int networkDelayTime(vector<vector<int>>& times, int N, int
K) {
        int MAX_TIME = 101 * 100;
        vector<int> dist(N, MAX_TIME);
        dist[K - 1] = 0;
        for (int i = 0; i < N; ++i)
            for (const auto& time : times) {
                int u = time[0] - 1, v = time[1] - 1, w = time[2];
                dist[v] = min(dist[v], dist[u] + w);
            }

        int max_dist = *max_element(dist.begin(), dist.end());
        return max_dist == MAX_TIME ? -1 : max_dist;
    }
};

int main() {
    Solution solution;
    vector<vector<int>> times = {
        {2, 1, 1},
        {2, 3, 1},
        {3, 4, 1}
    };
    int N = 4;
    int K = 2;
```

```
    int result = solution.networkDelayTime(times, N, K);
    cout << "The network delay time is: " << result << endl;
    return 0;
}
```

Input:

cpp Copy code

```cpp
vector<vector<int>> times = {
    {2, 1, 1},
    {2, 3, 1},
    {3, 4, 1}
};
int N = 4;
int K = 2;
```

Output:

arduino Copy code

```arduino
The network delay time is: 2
```