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

typedef long long int ll;
map<string,int> cidades;
map<int,string> vertices;
vector<vector<pair<int,int> > > Grafo(30);
int pai[30];
int n,m;

void dijkstra(string o,string des){

    int dist[n];
    for(int i=0;i<n;i++) dist[i] = 10101010;
    priority queue<pair<int,int>, vector<pair<int,int> >, greater<pair<int,int> > > pq;

    pq.push(make pair(0,cidades[o]));
    dist[cidades[o]] = 0;

    while(!pq.empty()){

        int d = pq.top().first;
        int v = pq.top().second;
        pq.pop();
        for(int i=0;i<Grafo[v].size();i++){
            if(dist[Grafo[v][i].first] > Grafo[v][i].second+d){
                pq.push(make pair(Grafo[v][i].second+d,Grafo[v][i].first));
                pai[Grafo[v][i].first] = v;
                dist[Grafo[v][i].first] = Grafo[v][i].second+d;
            }
        }
    }


}

main(){

    memset(pai,-1,sizeof pai);
    string o,des,e,aux;
    int d;
    cin >> n;
    cin.ignore();
    for(int i=0;i<n;i++){
        cin >> e;
        cidades[e] = i;
        vertices[i] = e;
    }
    cin >> m;
    for(int i=0;i<m;i++){
        cin.ignore();
        cin >> e >> aux >> d;
        Grafo[cidades[e]].push back(make pair(cidades[aux],d));
        Grafo[cidades[aux]].push back(make pair(cidades[e],d));
    }
    cin >> o >> des;

    dijkstra(o,des);
    vector<string> ans;
    while(pai[cidades[des]]!=-1){
        ans.push back(des);
        des = vertices[pai[cidades[des]]];
    }
    ans.push back(o);
    for(int i=ans.size()-1;i>=0;i--){
        if(i!=ans.size()-1)
            cout << "-";
        cout << ans[i];
    }
```

```
70      cout << endl;
71  }
72
```