

```
#include<iostream>
#include<climits>
using namespace std;

int findMinVertex(int* distance, bool* visited, int n){

    int minVertex = -1;
    for(int i = 0; i < n; i++){
        if(!visited[i] && (minVertex == -1 || distance[i] < distance[minVertex])){
            minVertex = i;
        }
    }
    return minVertex;
}

void dijkstra(int** edges, int n){
    int* distance = new int[n];
    bool* visited = new bool[n];

    for(int i = 0; i < n; i++){
        distance[i] = INT_MAX;
        visited[i] = false;
    }

    distance[0] = 0;

    for(int i = 0; i < n - 1; i++){
        int minVertex = findMinVertex(distance, visited, n);
        visited[minVertex] = true;
        for(int j = 0; j < n; j++){
            if(edges[minVertex][j] != 0 && !visited[j]){
                int dist = distance[minVertex] + edges[minVertex][j];
                if(dist < distance[j]){
                    distance[j] = dist;
                }
            }
        }
    }

    for(int i = 0; i < n; i++){
        cout << i << " " << distance[i] << endl;
    }
    delete [] visited;
    delete [] distance;
}
```

```

int main() {
    int n;
    int e;
    cin >> n >> e;
    int** edges = new int*[n];
    for (int i = 0; i < n; i++) {
        edges[i] = new int[n];
        for (int j = 0; j < n; j++) {
            edges[i][j] = 0;
        }
    }

    for (int i = 0; i < e; i++) {
        int f, s, weight;
        cin >> f >> s >> weight;
        edges[f][s] = weight;
        edges[s][f] = weight;
    }
    cout << endl;
    dijkstra(edges, n);

    for (int i = 0; i < n; i++) {
        delete [] edges[i];
    }
    delete [] edges;
}

```