Dijkstra usando vector<nodo>

```
#include <vector>
#include <utility>
#include <limits>
#include <iostream>

using namespace std;
typedef unsigned long long ull;

class nodo {
public:
     bool esta;
     ull peso;
     vector< pair<nodo*, ull> > ad;
     //constructor no necesario si se van a reiniciar los nodos
     nodo(){
          esta = true;
          peso = numeric_limits<ull>().max(); //pongo en infinito
     }
};
typedef pair<nodo*, ull> par;

ull dijs(vector<nodo> &v, int desde, int hasta){
     //si dijs() se llama varias veces con los mismos datos (mismo
     //grafo), reiniciar los nodos de la siguiente forma:
     //for (int i=0; i<v.size(); ++i){
          //v[i].peso = numeric_limits<ull>().max();
          //v[i].esta = true;
     //}

     //pongo 0 al inicio
     v[desde].peso = 0;

     while (v[hasta].esta){ //mientras hasta esté adentro
          //selecciono peso mínimo
          ull pesominimo = numeric_limits<ull>().max();
          int minimo;
          for (int i=0; i<v.size(); ++i){
               if (v[i].esta && v[i].peso < pesominimo){
                    pesominimo = v[i].peso;
                    minimo = i;
               }
          }
          //quito el mínimo
          v[minimo].esta = false;

          //para cada adyacente actualizo sus pesos
          for (int i=0; i<v[minimo].ad.size(); ++i){
               if (v[minimo].ad[i].first->esta)
                    v[minimo].ad[i].first->peso =
```

```
min( v[minimo].ad[i].first->peso, v[minimo].peso +
v[minimo].ad[i].second );
          } //del for
     } //del while
     return v[hasta].peso;
}

//uso:
int main(){
     int n;
     cin>>n;
     while (n!=0){
          nodo nuevo;
          vector<nodo> v(n, nuevo);
          ull a, b;
          for (int i=0; i<n-1; ++i){ //leo adyacencias
               cin>>a>>b;
               par p;
               p.first = &(v[a]);
               p.second = b;
               v[i+1].ad.push_back(p);

               p.first = &(v[i+1]); //p.second = b;
               v[a].ad.push_back(p);
          }
          cin>>a>>b;
          cout<<dijs(v, a, b)<<'\n';
     }
     return 0;
}
```