1D- Dijkstra:

Given the description of some roads print the case number and the minimum distance Tanvir has to travel to reach Atiq's house. If it's impossible, then print **'Impossible'**.

Code:

```
#include<bits/stdc++.h>
using namespace std;
vector<long long >edge[1005],cost[1005];
#define inf 0x3f3f3f3f
long long  dis[1005];
class node
{
public:
   long long  city,costt;
   bool operator < (const node & b)const
   {
      return costt > b.costt;
   }
};
long long  dijkstra(long long  src,long long  des)
{
   for(long long  i = 0; i<=1004; i++)
      dis[i] = inf;
   priority_queue<node>pq;
   node u,v;
   u.city = src;
   u.costt = 0;
   pq.push(u);
   dis[src] = 0;
   while(!pq.empty())
   {
      u = pq.top();
      pq.pop();
      long long  ct = u.city;
      long long  cst = dis[u.city];
      for(long long  i=0; i<edge[ct].size(); i++)
      {
         v.city = edge[ct][i];
         v.costt= cost[ct][i] + cst;
         if(v.costt<dis[v.city])
         {
            dis[v.city] = v.costt;
            pq.push(v);
         }
      }
   }
   return dis[des];

}
int main()
{
   long long  t,w=0;
   scanf("%lld",&t);
   while(t--)
   {
      long long  nod,edg;
      scanf("%lld %lld",&nod,&edg);
      while(edg--)
      {
         long long f,t,c;
         scanf("%lld %lld %lld",&f,&t,&c);
         edge[f].push_back(t);
         edge[t].push_back(f);
         cost[f].push_back(c);
         cost[t].push_back(c);
      }
      long long f,t;
      //cin>>f>>t;
```

```
        printf("Case %lld: ",++w);
        long long  k = dijkstra(1,nod);
        if(k!=inf)
            cout<<k<<endl;
        else
            cout<<"Impossible"<<endl;
        for(long long  i = 0; i<=1004; i++)
        {
            cost[i].clear();
            edge[i].clear();
        }
    }
    return 0;
}
```