

```
1.  /*****      dijkstra special by Shadman      *****/
2.  #include<cstdio>
3.  #include<cstring>
4.  #include<algorithm>
5.  #include<queue>
6.  #define ll long long
7.  using namespace std;
8.  const int MAXN=110;
9.  const int MAXE=2010;
10. const int INF=999999999;
11. int n,m,l;
12. struct EDGE
13. {
14.     int v,next;
15.     int dis;
16. }edge[MAXE];
17. struct HeadNode
18. {
19.     int d,u;
20.     bool operator < (const HeadNode& rhs) const
21.     {
22.         return d>rhs.d;
23.     }
24. };
25. int head[MAXN],size,pre[MAXN];
26. ll sum[MAXN],ans0,ans,tmp;
27. bool bel[MAXN][MAXE],del[MAXE];
28. void init()
29. {
30.     memset(head,-1,sizeof(head));
31.     size=0;
32. }
33. void add_edge(int u,int v,int dis)
34. {
35.     edge[size].v=v;
36.     edge[size].dis=dis;
37.     edge[size].next=head[u];
38.     head[u]=size++;
39. }
40. int dist[MAXN];
41. bool vis[MAXN];
42. void dijkstra(int s)
43. {
44.     memset(pre,-1,sizeof(pre));
45.     memset(vis,0,sizeof(vis));
46.     priority_queue<HeadNode> q;
47.     for(int i=1;i<=n;i++)
48.         dist[i]=INF;
49.     dist[s]=0;
50.     sum[s]=0;
51.     q.push(HeadNode{0,s});
52.     while(!q.empty())
53.     {
54.         HeadNode x=q.top();
55.         q.pop();
56.         int u=x.u;
57.         if(vis[u])
58.             continue;
```

```
59.     vis[u]=1;
60.     for(int i=head[u];i!=-1;i=edge[i].next)
61.     {
62.         int v=edge[i].v;
63.         if(dist[v]>dist[u]+edge[i].dis)
64.         {
65.             dist[v]=dist[u]+edge[i].dis;
66.             pre[v]=i;
67.             q.push(HeadNode{dist[v],v});
68.         }
69.     }
70. }
71. for(int i=1;i<=n;i++)
72. {
73.     if(i!=s)
74.     {
75.         if(dist[i]<INF)
76.             bel[s][pre[i]]=bel[s][pre[i]^1]=1;
77.         else
78.             dist[i]=1;
79.         ans0+=dist[i];
80.         sum[s]+=dist[i];
81.     }
82. }
83. }
84. void dijkstra1(int s)
85. {
86.     memset(vis,0,sizeof(vis));
87.     priority_queue<HeadNode> q;
88.     for(int i=1;i<=n;i++)
89.         dist[i]=INF;
90.     dist[s]=0;
91.     q.push(HeadNode{0,s});
92.     while(!q.empty())
93.     {
94.         HeadNode x=q.top();
95.         q.pop();
96.         int u=x.u;
97.         if(vis[u])
98.             continue;
99.         vis[u]=1;
100.        for(int i=head[u];i!=-1;i=edge[i].next)
101.        {
102.            if(del[i])
103.                continue;
104.            int v=edge[i].v;
105.            if(dist[v]>dist[u]+edge[i].dis)
106.            {
107.                dist[v]=dist[u]+edge[i].dis;
108.                q.push(HeadNode{dist[v],v});
109.            }
110.        }
111.    }
112.    ll k=0;
113.    for(int i=1;i<=n;i++)
114.    {
115.        if(i!=s)
116.        {
```

```
117.         if(dist[i]==INF)
118.             dist[i]=1;
119.             k+=dist[i];
120.         }
121.     }
122.     tmp=tmp-sum[s]+k;
123. }
124. int main()
125. {
126.     int i;
127.     while(scanf("%d%d%d",&n,&m,&l)!=EOF)
128.     {
129.         int u,v,d;
130.         init();
131.         while(m--)
132.         {
133.             scanf("%d%d%d",&u,&v,&d);
134.             add_edge(u,v,d);
135.             add_edge(v,u,d);
136.         }
137.         ans0=0;
138.         memset(bel,0,sizeof(bel));
139.         for(i=1;i<=n;i++)
140.             dijkstra(i);
141.         printf("%lld ",ans0);
142.         ans=ans0;
143.         memset(del,0,sizeof(del));
144.         for(int i=0;i<size;i+=2)
145.         {
146.             del[i]=del[i^1]=1;
147.             tmp=ans0;
148.             for(int j=1;j<=n;j++)
149.                 if(bel[j][i])
150.                     dijkstra1(j);
151.             del[i]=del[i^1]=0;
152.             ans=max(ans,tmp);
153.         }
154.         printf("%lld\n",ans);
155.     }
156.     return 0;
157. }
```