[解题报告] POJ3345

[Source] http://poj.org/problem?id=3345

[Description]

为了能够拉到至少 m 张选票，现在需要用钱贿赂一些人。但是有些人是另一些人的上司，每个人至多有有 1 个上司，并且关系不会成环。只要贿赂了一个人，那么他的所有下属的票都会得到。贿赂每个人需要花费 Wi 金钱，问至少需要多少金钱能够达到满足要求。

[Solution]

树形 DP，树上的 0/1 背包。DP[i][j]表示在以结点 i 为根节点形成的子树上，贿赂 j 个人的最小开销，sum[i]表示节点 i 的下属个数，转移方程如下：

DP[i][j]=Min{DP[i][j-k]+DP[i_son][k]} (k from 0 to j);

DP[i][sum[i]]=W[i];

注意边界条件和初始化。由于可能是多棵树，可以虚拟一个超级 boss，向所有入度为 0 的点连边建树。

这道题我觉得十分蛋疼的地方就是读入，之前做的时候就纠结了好久。

[Code]

```
#include<iostream>
#include<cstdio>
#include<cstring>
#include<algorithm>
#define N 400
#define oo 1000000000
using namespace std;
char name[N][150];
int sum[N],cost[N],dp[N][N];
int cnt,edge_num;
int first[N],next[N],end[N],degree[N];

void dfs(int u)
{
	int i,j,k,v;
```

```
        sum[u]=0;
        dp[u][0]=0;

        for (i=first[u];i;i=next[i])
        {
                v=end[i];
                dfs(v);
                sum[u]+=sum[v];
                for (j=sum[u];j>=0;j--)
                        for (k=1;k<=j;k++)

        dp[u][j]=min(dp[u][j],dp[u][j-k]+dp[v][k]);
        }
        sum[u]++;
        dp[u][sum[u]]=cost[u];
}

int find(char str[])
{
        int res,i;
        res=0;

        for (i=1;i<=cnt;i++)
                if (!strcmp(str,name[i]))
                {
                        res=i;
                        break;
                }
        if (res>0)
                return res;
        else
        {
                strcpy(name[++cnt],str);
                return cnt;
        }
}
int main()
{
        int n,m,i,j,u,v,ans;
        char ch[100],c;
        char str[150];

        while (gets(ch)&&ch[0]!='#')
        {
```

```
        sscanf(ch,"%d%d",&n,&m);

        memset(first,0,sizeof(first));
        memset(degree,0,sizeof(degree));
        memset(cost,0,sizeof(cost));
        for (i=0;i<=n+1;i++)
                for (j=0;j<=n+1;j++)
                        dp[i][j]=oo;

cnt=0;
edge_num=1;
for (i=1;i<=n;i++)
{
        scanf("%s",str);
        u=find(str);
                scanf("%d",&cost[u]);
    while (getchar()!='\n')
    {
        scanf("%s",str);
        v=find(str);
                        end[edge_num]=v;
                        next[edge_num]=first[u];
                        first[u]=edge_num;
                        edge_num++;
                        degree[v]++;
                }
        }


        u=n+1;
        for (i=1;i<=n;i++)
                if (degree[i]==0)
                {
                        end[edge_num]=i;
                        next[edge_num]=first[u];
                        first[u]=edge_num;
                        edge_num++;
                }

dfs(u);
ans=oo;

for (i=m;i<=n;i++)
```

```
                    if (ans>dp[u][i])
                        ans=dp[u][i];

            printf("%d\n",ans);
        }

        return 0;
}
```