POJ 2057

不太聪明的版本

```
#include<iostream>
#include<cstdio>
#include<algorithm>
#include<cstring>
#include<cmath>
#include<cstdlib>
#define INF 0x3f3f3f3f
using namespace std;
int n,f[1005],sz[1005],head[1005],mxlen[1005],cnt;
bool worm[1005];
struct Node{int nxt,to;}Edges[1005*2];
void addedge(int u,int v)
{
    Edges[++cnt].nxt=head[u];
    head[u]=cnt,Edges[cnt].to=v;
}
int dfs(int u)
{
    if(head[u]==-1){sz[u]=1;return f[u];}
    if(f[u])return f[u];
    int son[8],top=0,p[8]={0,1,2,3,4,5,6,7};
    for(int i=head[u];~i;i=Edges[i].nxt)
    {
        int v=Edges[i].to;
        dfs(v);
        sz[u]+=sz[v],son[top++]=v;
    }
    f[u]=INF;
    do
    {
        int len=0,res=0;
        for(int i=0;i<top;i++)
        {
            int v=son[p[i]];
            len+=i?2:1;
            res+=dfs(v)+sz[v]*len;
            if(!worm[v])len+=mxlen[v];
        }
        len++;
        if(res<f[u]||(res==f[u]&&len<mxlen[u]))
        f[u]=res,mxlen[u]=len;
    }
    while(next_permutation(p,p+top));
    return f[u];
}
int main()
{
```

```
    while(~scanf("%d",&n))
    {
        if(!n)break;
        memset(f,0,sizeof(f));
        memset(sz,0,sizeof(sz));
        memset(head,-1,sizeof(head));
        memset(worm,0,sizeof(worm));
        memset(mxlen,0,sizeof(mxlen));
        cnt=0;
        for(int i=1;i<=n;i++)
        {
            int f;
            char c;
            scanf("%d %c",&f,&c);
            if(c=='Y')worm[i]=1;
            if(f>0)addedge(f,i);
        }
        printf("%.4lf\n",1.0*dfs(1)/sz[1]);
    }
    return 0;
}
```

贪心

```
#include<iostream>
#include<cstdio>
#include<algorithm>
#include<cstring>
#include<cmath>
#include<cstdlib>
#define INF 0x3f3f3f3f
using namespace std;
int n,f[1005],sz[1005],head[1005],mxlen[1005],cnt;
bool worm[1005];
struct Node{int nxt,to;}Edges[1005*2];
void addedge(int u,int v)
{
    Edges[++cnt].nxt=head[u];
    head[u]=cnt,Edges[cnt].to=v;
}
bool cmp(int x,int y){return (mxlen[x]+2)*sz[y]<(mxlen[y]+2)*sz[x];}
int dfs(int u)
{
    if(head[u]==-1){sz[u]=1;return f[u];}
    if(f[u])return f[u];
    int son[8],top=0;
    for(int i=head[u];~i;i=Edges[i].nxt)
    {
        int v=Edges[i].to;
        dfs(v);
```

```
        sz[u]+=sz[v],son[top++]=v;
        mxlen[u]+=mxlen[v]+2;
    }
    if(worm[u])mxlen[u]=0;
    f[u]=0;
    sort(son,son+top,cmp);
    for(int i=0;i<top;i++)
    {
        f[u]+=f[son[i]]+sz[son[i]];
        for(int j=0;j<i;j++)
        f[u]+=(mxlen[son[j]]+2)*sz[son[i]];
    }
    return f[u];
}
int main()
{
    while(~scanf("%d",&n))
    {
        if(!n)break;
        memset(f,0,sizeof(f));
        memset(sz,0,sizeof(sz));
        memset(head,-1,sizeof(head));
        memset(worm,0,sizeof(worm));
        memset(mxlen,0,sizeof(mxlen));
        cnt=0;
        for(int i=1;i<=n;i++)
        {
            int f;
            char c;
            scanf("%d %c",&f,&c);
            if(c=='Y')worm[i]=1;
            if(f>0)addedge(f,i);
        }
        printf("%.4lf\n",1.0*dfs(1)/sz[1]);
    }
    return 0;
}
```