POJ 3281 Dining
ID：ChenJiayang

【简述】
使用拆点网络流，一头牛要保证只有一份食物和一份饮料，而我的建图为源点->食物->牛->饮料->汇点，食物和牛相连能保证是一份，但牛与饮料相连不能保证是一份，因此通过拆分牛的节点，从而保证牛与牛之间的流量是一，这就控制了仅有一份饮料流向牛。

Code（C++）Memory:1160K Time:250ms

```
#include<stdio.h>
#include<iostream>
#include<string.h>
#include<algorithm>
#include<queue>
using namespace std;
const int MAXN=500;
const int INF=0x3fffffff;
int g[MAXN][MAXN];
 int path[MAXN],flow[MAXN],start,end;
int n;
queue<int>q;
int bfs()
{
	int i,t;
	while(!q.empty())q.pop();
	memset(path,-1,sizeof(path));//路径初始化成-1
	path[start]=0;
	flow[start]=INF;
	q.push(start);
	while(!q.empty())
	{
		t=q.front();
		q.pop();
		if(t==end)break;
		for(i=0;i<=n;i++)
		{
			if(i!=start&&path[i]==-1&&g[t][i])
			{
				flow[i]=flow[t]<g[t][i]?flow[t]:g[t][i];
				q.push(i);
				path[i]=t;
			}
		}
	}
```

```
    if(path[end]==-1){return -1;}
    return flow[end];
}
int Edmonds_Karp()
{
    int max_flow=0;
    int step,now,pre;
    while((step=bfs())!=-1)
    {
        max_flow+=step;
        now=end;
        while(now!=start)
        {
            pre=path[now];
            g[pre][now]-=step;
            g[now][pre]+=step;
            now=pre;
        }
    }
    return max_flow;
}
int main()
{
    int N,F,D;
    while(scanf("%d%d%d",&N,&F,&D)!=EOF)
    {
        memset(g,0,sizeof(g));
        n=F+D+2*N+1;
        start=0;
        end=n;
        for(int i=1;i<=F;i++)g[0][i]=1;
        for(int i=F+2*N+1;i<=F+2*N+D;i++)g[i][n]=1;
        for(int i=1;i<=N;i++)g[F+2*i-1][F+2*i]=1;
        int k1,k2;
        int u;
        for(int i=1;i<=N;i++)
        {
            scanf("%d%d",&k1,&k2);
            while(k1--)
            {
                scanf("%d",&u);
                g[u][F+2*i-1]=1;
            }
            while(k2--)
```

```
                {
                    scanf("%d",&u);
                    g[F+2*i][F+2*N+u]=1;
                }
            }
            printf("%d\n",Edmonds_Karp());
        }
        return 0;
}
```