# Wangluoliu

1002

//bzoj1001

#include <iostream>

#include <cstring>

#include <cstdio>

#include <queue>

using namespace std;

const int N=2000006, INF=0x3fffffff, E=N\*3;

struct ARC {

int u, val, next;

inline void init(int a, int b, int c) {

u=a, val=b, next=c;

}

} arc[E];

int head[N], tot, S, T, n, m, dis[N];

bool vs[N];

struct data {

int u, dis;

data() {}

data(int a, int b) : u(a), dis(b) {}

bool operator < (const data &T) const {

return dis>T.dis;

}

};

inline void add\_arc(int s, int t, int val) {

arc[tot].init(t, val, head[s]);

head[s]=tot++;

}

priority\_queue <data> Q;

void Dijkstra() {

fill(dis, dis+T+1, INF);

fill(vs, vs+T+1, 0);

while(!Q.empty()) Q.pop();

dis[S]=0, Q.push(data(S, 0));

for(int u; !Q.empty(); ) {

u=Q.top().u, Q.pop();

if(vs[u]) continue;

if(u==T) {

printf("%d\n", dis[T]);

break;

}

vs[u]=1;

for(int e=head[u]; e!=-1; e=arc[e].next) {

int v=arc[e].u;

if(vs[v] || dis[u]+arc[e].val>=dis[v]) continue;

dis[v]=dis[u]+arc[e].val;

Q.push(data(v, dis[v]));

}

}

}

void Input() {

for(int i=0, id1, id2, a; i<=n-1; i++)

for(int j=1; j<=m-1; j++) {

read(a);

id1=((i-1)\*(m-1)+j)\*2-1;

id2=(i\*(m-1)+j)\*2;

if(i==0) id1=T;

else if(i==n-1) id2=S;

add\_arc(id1, id2, a);

add\_arc(id2, id1, a);

}

for(int i=1, id1, id2, a; i<=n-1; i++)

for(int j=0; j<m; j++) {

read(a);

id1=((i-1)\*(m-1)+j)\*2;

id2=((i-1)\*(m-1)+j+1)\*2-1;

if(j==0) id1=S;

else if(j==m-1) id2=T;

add\_arc(id1, id2, a);

add\_arc(id2, id1, a);

}

for(int i=1, id1, id2, a; i<=n-1; i++)

for(int j=1; j<=m-1; j++) {

read(a);

id1=((i-1)\*(m-1)+j)\*2;

id2=((i-1)\*(m-1)+j)\*2-1;

add\_arc(id1, id2, a);

add\_arc(id2, id1, a);

}

}

int main() {

read(n), read(m);

S=0, T=(n-1)\*(m-1)\*2+1;

fill(head, head+T+1, -1), tot=0;

if(n==1 || m==1) {

if(n>m) swap(n, m);

int ans=INF;

for(int i=1, a; i<m; i++) {

read(a);

if(ans>a) ans=a;

}

printf("%d\n", ans==INF?0:ans);

}

else Input(), Dijkstra();

return 0;

}

3894

#include<algorithm>

#define inf 1000000000

#define pa pair<int,int>

#define ll long long

#define mod 1000000007

using namespace std;

#define p(i,j) (i-1)\*m+j

int read()

{

int x=0,f=1;char ch=getchar();

while(ch<'0'||ch>'9'){if(ch=='-')f=-1;ch=getchar();}

while(ch>='0'&&ch<='9'){x=x\*10+ch-'0';ch=getchar();}

return x\*f;

}

int ans,tot;

int n,m,T,cnt=1;

int q[30005],last[30005],h[30005];

int xx[5]={0,0,1,-1,0},yy[5]={1,-1,0,0,0};

struct edge{

int to,next,v;

}e[1000005];

void insert(int u,int v,int w)

{

e[++cnt].to=v;e[cnt].next=last[u];last[u]=cnt;e[cnt].v=w;

e[++cnt].to=u;e[cnt].next=last[v];last[v]=cnt;e[cnt].v=0;

}

bool bfs()

{

int head=0,tail=1;

memset(h,-1,sizeof(h));

q[0]=0;h[0]=0;

while(head!=tail)

{

int now=q[head];head++;

for(int i=last[now];i;i=e[i].next)

if(e[i].v&&h[e[i].to]==-1)

{

h[e[i].to]=h[now]+1;

q[tail++]=e[i].to;

}

}

return h[T]!=-1;

}

int dfs(int x,int f)

{

if(x==T)return f;

int w,used=0;

for(int i=last[x];i;i=e[i].next)

if(e[i].v&&h[e[i].to]==h[x]+1)

{

w=dfs(e[i].to,min(e[i].v,f-used));

e[i].v-=w;e[i^1].v+=w;

used+=w;if(used==f)return f;

}

if(!used)h[x]=-1;

return used;

}

int main()

{

n=read();m=read();T=3\*n\*m+1;

for(int i=1;i<=n;i++)

for(int j=1;j<=m;j++)

{

int x=read();

insert(0,p(i,j),x);

tot+=x;

}

for(int i=1;i<=n;i++)

for(int j=1;j<=m;j++)

{

int x=read();

insert(p(i,j),T,x);

tot+=x;

}

for(int i=1;i<=n;i++)

for(int j=1;j<=m;j++)

{

int val=read();tot+=val;

for(int k=0;k<5;k++)

{

int x=xx[k]+i,y=yy[k]+j;

if(x>n||y>m||x<1||y<1)continue;

insert(p(i,j)+n\*m,p(x,y),inf);

}

insert(0,p(i,j)+n\*m,val);

}

for(int i=1;i<=n;i++)

for(int j=1;j<=m;j++)

{

int val=read();tot+=val;

for(int k=0;k<5;k++)

{

int x=xx[k]+i,y=yy[k]+j;

if(x>n||y>m||x<1||y<1)continue;

insert(p(x,y),p(i,j)+2\*n\*m,inf);

}

insert(p(i,j)+2\*n\*m,T,val);

}

while(bfs())ans+=dfs(0,inf);

printf("%d\n",tot-ans);

return 0;

}

6118

#include<cstdio>

#include<algorithm>

#include<iostream>

#include<cstring>

using namespace std;

const int N=6100,INF=0x3f3f3f3f;

const int BIG=12345678;

const int M=N\*10;

int pre[N],d[N],p[N],ans;

int cnt,head[N];

int q[M],l,r;

struct edge{

int u,v,w,c,next;

}e[M];

void init(){

memset(head,-1,sizeof(head));

ans=cnt=0;

}

void addedge(int u,int v,int c,int w){

e[cnt].u=u,e[cnt].v=v,e[cnt].w=w,e[cnt].c=c;

e[cnt].next=head[u],head[u]=cnt++;

e[cnt].u=v,e[cnt].v=u,e[cnt].w=-w,e[cnt].c=0;

e[cnt].next=head[v],head[v]=cnt++;

}

int updata(int s,int t){

int i,f=INF,sum=0;

for(i=t;i!=s;i=e[pre[i]].u)

f=min(f,e[pre[i]].c);

for(i=t;i!=s;i=e[pre[i]].u){

e[pre[i]].c-=f;

e[pre[i]^1].c+=f;

sum+=f\*e[pre[i]].w;

}

if(sum/f>=2000) return 1;

//printf("CCC%d %d\n",sum,f);

ans=ans-sum+f\*2000;

return 0;

}

int spfa(int s,int t){

int i,u,v,w;

memset(p,0,sizeof(p));

memset(pre,-1,sizeof(pre));

memset(d,0x3f,sizeof(d));

l=r=0;

q[++r]=s,p[s]=1,d[s]=0;

while(l<r){

p[u=q[++l]]=0;

for(i=head[u];i!=-1;i=e[i].next){

v=e[i].v,w=e[i].w;

if(e[i].c&&d[v]>d[u]+w){

d[v]=d[u]+w;

pre[v]=i;

if(!p[v]){

p[v]=1;

q[++r]=v;

}

}

}

}

if(pre[t]==-1)return 0;

return 1;

}

int MiCMaF(int s,int t){

ans=0;

while(spfa(s,t))

if(updata(s,t))

return 0;

}

int A,B,C,D;

int main(){

int n,m;

while(cin>>n>>m){

init();

for(int i=1;i<=n;i++){

scanf("%d%d%d%d",&A,&B,&C,&D);

addedge(0,i,B,A);

addedge(i,n+1,D,2000-C);

}

for(int i=1;i<=m;i++){

scanf("%d%d%d",&A,&B,&C);

addedge(A,B,BIG,C);

addedge(B,A,BIG,C);

}

MiCMaF(0,n+1);

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

}

return 0;

}

//7264

#include <iostream>

#include <cstring>

#include <cstdio>

#include <algorithm>

#include <queue>

using namespace std;

const int INF = 0x3f3f3f3f;

const int maxn = 1e3+5;

const int maxv = 1e5 + 5;

int head[maxn], cur[maxn], d[maxn], s, t, k, sum;

int n, m, g;

struct node

{

int v, w, next;

}edge[maxv];

void addEdge(int u, int v, int w)

{

edge[k].v = v;

edge[k].w = w;

edge[k].next = head[u];

head[u] = k++;

edge[k].v = u;

edge[k].w = 0;

edge[k].next = head[v];

head[v] = k++;

}

int bfs()

{

memset(d, 0, sizeof(d));

d[s] = 1;

queue<int> q;

q.push(s);

while(!q.empty())

{

int u = q.front();

if(u == t) return 1;

q.pop();

for(int i = head[u]; i != -1; i = edge[i].next)

{

int to = edge[i].v, w = edge[i].w;

if(w && d[to] == 0)

{

d[to] = d[u] + 1;

if(to == t) return 1;

q.push(to);

}

}

}

return 0;

}

int dfs(int u, int maxflow)

{

if(u == t) return maxflow;

int ret = 0;

for(int i = cur[u]; i != -1; i = edge[i].next)

{

int to = edge[i].v, w = edge[i].w;

if(w && d[to] == d[u]+1)

{

int f = dfs(to, min(maxflow-ret, w));

edge[i].w -= f;

edge[i^1].w += f;

ret += f;

if(ret == maxflow) return ret;

}

}

return ret;

}

int Dinic()

{

int ans = 0;

while(bfs() == 1)

{

memcpy(cur, head, sizeof(head));

ans += dfs(s, INF);

}

return ans;

}

int main()

{

int T;

scanf("%d", &T);

while(T--)

{

scanf("%d%d%d", &n, &m, &g);

s = 0, t = n\*2+1, k = 0;

memset(head, -1, sizeof(head));

int x, y, z;

for(int i = 1; i <= m; i++)

{

scanf("%d%d%d", &x, &y, &z);

addEdge(n+x, y, z);

}

for(int i = 1; i <= n; i++)

{

scanf("%d", &x);

addEdge(s, i, x);

}

for(int i = 1; i <= n; i++)

{

scanf("%d", &x);

addEdge(i, i+n, x);

}

addEdge(g+n, t, INF);

printf("%d\n", Dinic());

}

return 0;

}