目录

[二分图 匈牙利 1](#_Toc502932313)

[网络流 DINIC 3](#_Toc502932314)

[FFT 5](#_Toc502932315)

[某道数学题，add 1-x；del-y 7](#_Toc502932316)

[矩阵链乘 输出方案 8](#_Toc502932317)

[延迟操作的LIS 9](#_Toc502932318)

[msort逆序对 9](#_Toc502932319)

[qsort第k大 10](#_Toc502932320)

[凸包 10](#_Toc502932321)

[最远最近点对 12](#_Toc502932322)

二分图 匈牙利

int n,m;

int i,j,k,t;

vector<int>edge[N];

int used[N];

/\*注意数组的标号，必须满足二分图的条件

int matching[N];

bool dfs(int u){

int v,i;

REP(i,edge[u].size()){

v=edge[u][i];

if (!used[v]){

used[v]=1;

if (matching[v]==-1||dfs(matching[v])){

matching[v]=u;

matching[u]=v;

return 1;

}

}

}

return 0;

}

int DFS(){

int ans=0;

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

int u;

FOR(u,1,n){

if (matching[u]==-1){

memset(used,0,sizeof(used));

if (dfs(u)) ans++;

}

}

return ans;

}\*/

/\*注意数组的标号，必须满足二分图的条件

queue<int> Q;

int prev[N];//两格

int matching[N];//结果

int check[N];//matchright

int BFS(){

int ans=0;

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

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

FOR(i,1,n){

if (matching[i]==-1){

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

Q.push(i);

prev[i]=-1;

bool flag=false;

while (!Q.empty()&&!flag){

int u=Q.front();Q.pop();

for (j=0;!flag&&j<edge[u].size();j++){

int v=edge[u][j];

if (check[v]!=i){

check[v]=i;

Q.push(matching[v]);

if (matching[v]!=-1) prev[matching[v]]=u;

else{

flag=1;

int d=u,e=v;

while (d!=-1){

int t=matching[d];

matching[d]=e;

matching[e]=d;

d=prev[d];

e=t;

}

}

}

}

}

if (matching[i]!=-1) ans++;

}

}

return ans;

}\*/

int main(){

int T;

scanf("%d",&T);

while (T--){

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

FOR(i,1,n){

scanf("%d",&k);

edge[i].clear();

REP(j,k) scanf("%d",&t),edge[i].push\_back(t+n);

}

// printf("%d",BFS());

if (BFS()==n) puts("YES");

else puts("NO");

}

}

网络流 DINIC

struct node{

int to,cap,next;

node(int t=0,int c=0,int n=0):to(t),cap(c),next(n){}

}edge[maxn\*50];

int head[maxn];

int tot;

void addedge(int from,int to,int cap){

edge[tot].to=to;

edge[tot].next=head[from];

edge[tot].cap=cap;

head[from]=tot++;

edge[tot].to=from;

edge[tot].next=head[to];

edge[tot].cap=0;

head[to]=tot++;

}

queue<int> Q;

bool vis[maxn];

int d[maxn];

int cur[maxn];//当前弧优化

bool bfs(int s,int t){

memset(vis,0,sizeof(vis));

while (Q.size()) Q.pop();

Q.push(s);

d[s]=0;vis[s]=1;

int i;

while (!Q.empty()){

int x=Q.front();Q.pop();

for(i=head[x];i!=-1;i=edge[i].next){

if (!vis[edge[i].to]&&edge[i].cap){

vis[edge[i].to]=1;

d[edge[i].to]=d[x]+1;

Q.push(edge[i].to);

}

}

}

return vis[t];

}

int dfs(int x,int t,int flow){

if (x==t||flow==0) return flow;

int i,ret=0,f;

for (i=cur[x];i!=-1;i=edge[i].next){

if (d[x]+1==d[edge[i].to]&&((f=dfs(edge[i].to,t,min(flow,edge[i].cap)))>0)){

edge[i].cap-=f;

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

ret+=f;

flow-=f;

cur[x]=i;

if (flow==0) break;

}

}

return ret;

}

int n,m,i;

int u,v,len,ans;

int s,t;

int main(){

while (~scanf("%d%d",&n,&m)){

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

ans=0;tot=0;

s=n+1;t=n+2;

FOR(i,1,n){

int a,b;

scanf("%d%d",&a,&b);

addedge(s,i,a);

addedge(i,t,b);

}

FOR(i,1,m){

scanf("%d%d%d",&u,&v,&len);

addedge(u,v,len);

addedge(v,u,len);

}

while (bfs(s,t)){

int f;

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

while (f=dfs(s,t,INF)) ans+=f;

}

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

}

}

FFT

struct complex{

double a,b;

complex(double \_a=.0,double \_b=.0):a(\_a),b(\_b){}

complex operator+(const complex x)const{return complex(a+x.a,b+x.b);}

complex operator-(const complex x)const{return complex(a-x.a,b-x.b);}

complex operator\*(const complex x)const{return complex(a\*x.a-b\*x.b,a\*x.b+b\*x.a);}

};

void fft(complex \*A,int len,int inv){//抄的板子

int i,j,k;

for (i=1,j=len/2;i<len-1;i++){

if (i<j) swap(A[i],A[j]);

k=len/2;

while(j>=k){

j-=k;

k/=2;

}if (j<k) j+=k;

}

for(i=2;i<=len;i<<=1){

complex wn(cos(-inv\*2\*pi/i),sin(-inv\*2\*pi/i));

for (j=0;j<len;j+=i){

complex w(1.0,0.0);

for (k=j;k<(j+i/2);k++){

complex a=A[k],b=w\*A[k+i/2];

A[k]=a+b;

A[k+i/2]=a-b;

w=w\*wn;

}

}

}

if (inv==-1) REP(i,len) A[i].a/=len;

}

complex x1[maxn],x2[maxn];

char a[maxn],b[maxn];

int ans[maxn];

int main(){

int T;

int i,j,k;

// printf("%lf\n",pi);

scanf("%d",&T);

while (T--){

scanf("%s%s",a,b);

bool mark=0;;

int len1=strlen(a),len2=strlen(b),len=1;

if (a[0]=='-') {REP(i,len1) a[i]=a[i+1];len1--;mark^=1;}

if (b[0]=='-') {REP(i,len2) b[i]=b[i+1];len2--;mark^=1;}

while(len<=len1+len2+1) len<<=1;

REP(i,len1) x1[i]=complex(a[len1-i-1]-'0',0);

rep(i,len1,len) x1[i]=complex(0,0);

REP(i,len2) x2[i]=complex(b[len2-i-1]-'0',0);

rep(i,len2,len) x2[i]=complex(0,0);

fft(x1,len,1);fft(x2,len,1);

// REP(i,len) printf("%lf %lf\n",x1[i].a,x1[i].b);

// REP(i,len) printf("%lf %lf\n",x2[i].a,x2[i].b);

REP(i,len) x1[i]=x1[i]\*x2[i];

fft(x1,len,-1);

REP(i,len) ans[i]=x1[i].a+0.5;

REP(i,len) ans[i+1]+=ans[i]/10,ans[i]%=10;

while (ans[len-1]<=0&&len-1>0) len--;

if (mark) putchar('-');

rREP(i,len) putchar(ans[i]+'0');

puts("");

}

}

某道数学题，add 1-x；del-y

LL n,m,i,j;

LL cnt[maxn],sum[maxn];

bool mark[maxn];

LL ans;

const int MAX=1e6;

int main(){//(j+i-f)\*y>x ==> j+i-f>x/y 的用x //f<j+i-x/y

LL x,y;

while (~scanf("%lld%lld%lld",&n,&x,&y)){

FOR(i,1,MAX\*2) cnt[i]=sum[i]=0;

LL val;

FOR(i,1,n) scanf("%lld",&val),cnt[val]++,sum[val]+=val;

FOR(i,1,MAX\*2) cnt[i]+=cnt[i-1],sum[i]+=sum[i-1];

// FOR(i,1,20) printf("%3d ",i);puts("");

// FOR(i,1,20) printf("%3d ",cnt[i]);puts("");

// FOR(i,1,20) printf("%3d ",sum[i]);puts("");

ans=INFF;

FOR(i,2,MAX){

if (mark[i]) continue;

LL now=0;

LL t=max(0ll,i-1-x/y);

for (j=0;j<=MAX;j+=i){

mark[j]=1;

now+=((cnt[j+i-1]-cnt[j+t])\*(j+i)-(sum[j+i-1]-sum[j+t]))\*y+(cnt[j+t]-cnt[j])\*x;

// printf("now+==%d %d %d;%d\n",j,j+t,j+i-1,now);

}

// printf("i=%d %d t=%d\n",i,now,t);

mark[i]=0;

ans=min(ans,now);

}

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

}

}

矩阵链乘 输出方案

#include<iostream>

#include<cmath>

#include<cstdio>

using namespace std;

int a[302];

int b[302][302];

int c[302][302];

int bracket1[302];

int bracket2[302];

void out(int l,int r)

{

if (r-l<=1) return;

bracket1[l]++;

bracket2[r]++;

int k=c[l][r];

out(l,k);

out(k,r);

}

int main()

{

int n;

int i,j,k,t;

while (~scanf("%d",&n))

{

for (i=0;i<n+1;i++) scanf("%d",a+i);

for (i=0;i<=n;i++)

for (j=i;j<=n;j++) b[i][j]=c[i][j]=0;

for (t=2;t<=n;t++)

for (i=0;i<=n-t;i++)

{

j=i+t;

for (k=i+1;k<j;k++)

if (c[i][j]==0||b[i][j]>=b[i][k]+b[k][j]+a[i]\*a[j]\*a[k])

{

c[i][j]=k;

b[i][j]=b[i][k]+b[k][j]+a[i]\*a[j]\*a[k];

}

}

for (i=0;i<=n;i++) bracket1[i]=bracket2[i]=0;

out(0,n);

printf("%d\n",b[0][n]);

for (i=0;i<n;i++)

{

for (j=0;j<bracket2[i];j++) printf(")");

for (j=0;j<bracket1[i];j++) printf("(");

printf("A%d",i+1);

}

for (j=0;j<bracket2[n];j++) printf(")");

printf("\n");

}

}

延迟操作的LIS

//延迟修改

int n,k;

int a[maxn],b[maxn],tot;

int i,j;

int pos[maxn];

int main(){

int T;

scanf("%d%d",&n,&k);

tot=0;

FOR(i,0,n) b[i]=INF;

FOR(i,1,n) scanf("%d",&a[i]);

FOR(i,1,n){

if (i-k>=1){

b[pos[i-k]]=min(b[pos[i-k]],a[i-k]);

if (pos[i-k]==tot) tot++;

}pos[i]=lower\_bound(b,b+tot,a[i])-b;

}FOR(i,1,n) if (pos[i]==tot) tot++;

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

}

msort逆序对

void msort(int le,int ri)

{

if (le==ri) return;

int mid=(le+ri)>>1,i=le,j=mid+1,k=i;

msort(le,mid);msort(j,ri);

while (i<=mid||j<=ri)

{

if (i==mid+1) {b[k++]=a[j++]; ans+=mid-i+1;}

else if (j==ri+1) b[k++]=a[i++];

else if (a[i]<=a[j]) b[k++]=a[i++];

else {b[k++]=a[j++]; ans+=mid-i+1;}

}

for (i=le;i<=ri;i++) a[i]=b[i];

}

qsort第k大

void fqsort(int l,int r)

{

int le=l,ri=r,m;

m=a[le];

while (le<ri)

{

while (le<ri&&a[ri]<=m) ri--;

a[le]=a[ri];

while (le<ri&&a[le]>=m) le++;

a[ri]=a[le];

}

if (le==k) printf("%d\n",m);

else if (le>k) fqsort(l,le-1);

else fqsort(le+1,r);

}

凸包

inline int sgn(double x){

if (abs(x)<eps) return 0;

if (x<0) return -1;return 1;

}

struct point{

double x,y;

bool operator <(const point &A)const{

if (y<A.y) return 1;

if (y>A.y) return 0;

return x<A.x;

}

}P[maxn],p[maxn];

inline double X(point A,point B,point C){

return (B.x-A.x)\*(C.y-A.y)-(B.y-A.y)\*(C.x-A.x);

}

inline double len(point A,point B){

return sqrt((A.x-B.x)\*(A.x-B.x)+(A.y-B.y)\*(A.y-B.y));

}

bool cmp(point A,point B){

double cp=X(p[0],A,B);

if (sgn(cp)>0) return 1;

if (sgn(cp)<0) return 0;

return len(p[0],A)<len(p[0],B);

}

int solve(){

int tot=0,n,i;

scanf("%d",&n);

REP(i,n) scanf("%lf%lf",&p[i].x,&p[i].y);

if (n<6) return 0\*puts("No");

// assert(n>=6);

REP(i,n) if (p[i]<p[0]) swap(p[0],p[i]);

sort(p+1,p+n,cmp);

P[0]=p[0];

P[1]=p[1];

tot=1;

rep(i,2,n){

while (tot&&sgn(X(P[tot-1],P[tot],p[i])<0)) tot--;

P[++tot]=p[i];

}

point last=p[tot],pre=p[tot-1];

rREP(i,n-1)

if (sgn(X(last,p[i],p[0])==0)) P[++tot]=p[i];

else break;

// FOR(i,0,tot) printf(" %lf %lf\n",P[i].x,P[i].y);printf("%d ",tot);

P[++tot]=P[0];

P[++tot]=P[1];

FOR(i,1,tot-2)

if (sgn(X(P[i-1],P[i],P[i+1]))&&sgn(X(P[i],P[i+1],P[i+2])))

return 0\*puts("No");

puts("Yes");

}

int a[maxn];

int main(){

int T;

scanf("%d",&T);

while (T--) solve();

}

最远最近点对

inline int sgn(double x){

if (abs(x)<eps) return 0;

if (x<0) return -1;

return 1;

}

struct point{

LL x,y;

bool operator <(const point &a) const{

if (y<a.y) return 1;

if (y>a.y) return 0;

return x<a.x;

}

}p[maxn],P[maxn],p1[maxn];

inline LL X(point A,point B,point C){

return (B.x-A.x)\*(C.y-A.y)-(B.y-A.y)\*(C.x-A.x);

}

inline LL len(point A,point B){

return (A.x-B.x)\*(A.x-B.x)+(A.y-B.y)\*(A.y-B.y);

}

bool cmp(point A,point B){

LL cp=X(p[0],A,B);

if (cp>0) return 1;

if (cp<0) return 0;

return len(p[0],A)<len(p[0],B);

// return sgn(len(p[0],A)-len(p[0],B))<=0;

}

int n;

LL getMAX(){//求完凸包旋转卡壳

int tot,i,j,m;

LL ans=0;

if (n==1){

tot=0;

P[0]=p[0];

}else if (n==2){

tot=1;

P[0]=p[0];

P[1]=p[1];

}else{

REP(i,n) if (p[i]<p[0]) swap(p[0],p[i]);

sort(p+1,p+n,cmp);

P[0]=p[0];

P[1]=p[1];

tot=1;

rep(i,2,n){

while (tot&&X(P[tot-1],P[tot],p[i])<=0) tot--;

P[++tot]=p[i];

}

}m=tot;

FOR(i,0,tot) P[++m]=P[i];

j=0;ans=0;

FOR(i,0,m){

while (j<m&&len(P[i],P[j])<len(P[i],P[j+1])) j++;

ans=max(ans,len(P[i],P[j]));

}return ans;

}

inline int cmpx(point a,point b){return a.x<b.x;}

inline int cmpy(point a,point b){return a.y<b.y;}

LL getMIN(int l,int r){//分治求最近点对,nsqrtn

LL ans=0;

int i,j;

if(l>=r) return INFF;

if(l+1==r) return len(p[l],p[r]);

int mid=(l+r)>>1;

ans=min(getMIN(l,mid),getMIN(mid+1,r));

int cn=0;

FOR(i,l,r) if (p[i].x-p[mid].x<ans) p1[cn++]=p[i];

sort(p1,p1+cn,cmpy);

REP(i,cn){

rep(j,i+1,cn){

if (p1[j].y-p1[i].y>=ans) break;

ans=min(ans,len(p1[i],p1[j]));

}

}return ans;

}

int i,j,k;

LL ans;

int main(){//0->tot是凸包上的点

while (~scanf("%d",&n)){

REP(i,n) scanf("%lld%lld",&p[i].x,&p[i].y);

sort(p,p+n,cmpx);

printf("%lld %lld\n",getMIN(0,n-1),getMAX());

}

}

**概率DP：一定注意设的不变量是啥即可**

***Goodbye 2017的题***

int k,i,j;

LL pa,pb;

LL fa,fb;

LL f[1007][1007];//(ab)num,a\_num

LL ans;

int main(){

scanf("%d%d%d",&k,&pa,&pb);

fa=pa\*powMM(pa+pb,M-2)%M;

fb=pb\*powMM(pa+pb,M-2)%M;

LL P=pa\*powMM(pb,M-2)%M;

f[0][1]=1;

FOR(i,0,k-1){

FOR(j,1,k-1){

(f[i][j+1]+=fa\*f[i][j])%=M;

if (i+j<k) (f[i+j][j]+=fb\*f[i][j])%=M;

else (ans+=(i+j)\*fb%M\*f[i][j])%=M;

}(ans+=f[i][k]\*(i+k+P))%=M;

}//后方只满足第一种

printf("%I64d\n",ans);

}