**Dijstra:**

int dijktra(){

int i,j,v,ans,min;

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

final[i]=0;

d[i]=map[1][i];

}

final[1]=1;

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

min=MAX;

for(j=1;j<=n;j++){

if(!final[j]){

if(min>d[j]){

v=j;

min=d[j];

}

}

}

final[v]=1;

for(j=1;j<=n;j++){

if(!final[j]&&(min+map[v][j])<d[j]){

d[j]=min+map[v][j];

}

}

}

ans=0;

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

if(ans<d[i])ans=d[i];

}

return ans;

}

**Bfs:**

void bfs(){

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

queue<node> q;

node nod={sx,sy,0};

q.push(nod);

bool mark=false;

while(!q.empty()){

node u=q.front();

q.pop();

if(u.x==ex&&u.y==ey){

if(u.dis==0)break;

printf("%d\n",u.dis);

mark=true;

break;

}

node v;

for(int i=0;i<8;i++){

v.x=u.x+dir[i][0];

v.y=u.y+dir[i][1];

int nx=u.x+dir[i][0]/2;//马当前方向上前面的一个坐标

int ny=u.y+dir[i][1]/2;

//限制条件

if(map[nx][ny]=='#'||vis[v.x][v.y]==1||map[v.x][v.y]=='#'||v.x<0||v.y<0||v.x>=n||v.y>=m)

continue;

else{

v.dis=u.dis+1;

vis[v.x][v.y]=1;

q.push(v);

}

}

}

if(!mark)printf("-1\n");

return;

}

**Dfs(从s到指定位置T步):**

void dfs(int x,int y,int step){

int i,mx,my;

if(x==bx&&y==by){

if(step==t)

flag=1;

return;

}

if(step>=t)return;

for(i=0;i<4;i++){

mx=x+dir[i][0];

my=y+dir[i][1];

if(maze[mx][my]!='X'&&mx>=0&&my>=0&&mx<n&&my<m&&!vis[mx][my]){

vis[mx][my]=1;

dfs(mx,my,step+1);

vis[mx][my]=0;

if(flag)return;

}

}

}

**Floyd:**

void shortPath\_Floyd(){

int i,j,k;

for(k=1;k<=n;k++)

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

for(j=1;j<=n;j++){

if(map[i][j]>map[i][k]+map[k][j])map[i][j]=map[i][k]+map[k][j];

}

}

**并查集：**

void makeset(){

int i;

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

pre[i]=i;

cnt=n;

}int find(int x){

if(x!=pre[x])

pre[x]=find(pre[x]);

return pre[x];

}void uniun(int a,int b){

int fx,fy;

fx=find(a);

fy=find(b);

if(fx!=fy){

pre[fy]=fx;

cnt--;

}

return;

}

int main(){

int i,a,b;

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

if(m==0){

printf("%d\n",n-1);

continue;

}

makeset();

for(i=0;i<m;i++){

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

uniun(a,b);

}

printf("%d\n",cnt-1);

}

}

**[dp 母函数入门](http://blog.csdn.net/ydm1234/article/details/51339154) ：**

while(scanf("%d",&n)!=EOF){

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

c1[i]=1;

c2[i]=0;

}

for(i=2;i<=n;i++){

for(j=0;j<=n;j++){

for(k=0;k+j<=n;k+=i){

c2[j+k]+=c1[j];

}

}

for(j=0;j<=n;j++){

c1[j]=c2[j];

c2[j]=0;

}

}

printf("%d\n",c1[n]);

}

**Java大数：**

Scanner sca=new Scanner(System.in);

int t=sca.nextInt();

while(t>0){

t--;

BigInteger m=sca.nextBigInteger();

int n=sca.nextInt();

m=m.pow(n-1);

int gc=m.gcd(BigInteger.valueOf(n)).intValue();

System.out.println(n/gc+"/"+m.divide(BigInteger.valueOf(gc)));

}

**Bellman-Ford(单源最短路径判断是否有负权环路)：**

bool Bellman\_Ford()

{

for(int i = 1; i <= nodenum; ++i) //初始化

dis[i] = (i == original ? 0 : MAX);

for(int i = 1; i <= nodenum - 1; ++i)

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

if(dis[edge[j].v] > dis[edge[j].u] + edge[j].cost) //松弛（顺序一定不能反~）

{

dis[edge[j].v] = dis[edge[j].u] + edge[j].cost;

pre[edge[j].v] = edge[j].u;

}

bool flag = 1; //判断是否含有负权回路

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

if(dis[edge[i].v] > dis[edge[i].u] + edge[i].cost)

{

flag = 0;

break;

}

return flag;

}

**[使用next\_permutation求第m小的全排列：](http://blog.csdn.net/ydm1234/article/details/50985342)**

[#include<algorithm>](http://blog.csdn.net/ydm1234/article/details/50985342)

[int main(){](http://blog.csdn.net/ydm1234/article/details/50985342)

[int ans[1002],i;](http://blog.csdn.net/ydm1234/article/details/50985342)

[int n,m,k;](http://blog.csdn.net/ydm1234/article/details/50985342)

[while(scanf("%d%d",&n,&m)!=EOF){](http://blog.csdn.net/ydm1234/article/details/50985342)

[for(i=0;i<1002;i++){](http://blog.csdn.net/ydm1234/article/details/50985342)

[ans[i]=i+1;](http://blog.csdn.net/ydm1234/article/details/50985342)

[}](http://blog.csdn.net/ydm1234/article/details/50985342)

[k=2;](http://blog.csdn.net/ydm1234/article/details/50985342)

[if(m==1){](http://blog.csdn.net/ydm1234/article/details/50985342)

[for(i=0;i<n-1;i++)printf("%d ",ans[i]);](http://blog.csdn.net/ydm1234/article/details/50985342)

[printf("%d\n",ans[n-1]);](http://blog.csdn.net/ydm1234/article/details/50985342)

[}else](http://blog.csdn.net/ydm1234/article/details/50985342)

[while(next\_permutation(ans,ans+n)){](http://blog.csdn.net/ydm1234/article/details/50985342)

[if(k==m){](http://blog.csdn.net/ydm1234/article/details/50985342)

[for(i=0;i<n-1;i++)printf("%d ",ans[i]);](http://blog.csdn.net/ydm1234/article/details/50985342)

[printf("%d\n",ans[n-1]);](http://blog.csdn.net/ydm1234/article/details/50985342)

[break;](http://blog.csdn.net/ydm1234/article/details/50985342)

[}](http://blog.csdn.net/ydm1234/article/details/50985342)

[k++;](http://blog.csdn.net/ydm1234/article/details/50985342)

[}](http://blog.csdn.net/ydm1234/article/details/50985342)

[}](http://blog.csdn.net/ydm1234/article/details/50985342)

[return 0;](http://blog.csdn.net/ydm1234/article/details/50985342)

[}](http://blog.csdn.net/ydm1234/article/details/50985342)

# **[容斥原理统计区间内与n互质的个数：](http://blog.csdn.net/ydm1234/article/details/51193065)**

ll count\_prime(ll x,ll n){

pme.clear();

ll i,j;

for(i=2;i\*i<=n;i++)

if(n%i==0){

pme.push\_back(i);

while(n%i==0)n/=i;

}

if(n>1)pme.push\_back(n);

ll sum=0,value,cnt;

for(i=1;i<(1<<pme.size());i++){

value=1;

cnt=0;

for(j=0;j<pme.size();j++){

if(i&(1<<j)){

value\*=pme[j];

cnt++;

}

}

if(cnt&1)

sum+=x/value;

else sum-=x/value;

}

return x-sum;

}

# **[前缀判断](http://blog.csdn.net/ydm1234/article/details/50931277)：**

char\* prefix(char\* haystack\_start, char\* needle\_start)

{

char\* haystack = haystack\_start;

char\* needle = needle\_start;

while(\*haystack && \*needle){

if(\*(haystack++)!=\*(needle++)) return NULL; //填空位置

}

if(\*needle) return NULL;

return haystack\_start;

}int main(){

cout<<prefix("abcd1234","abcd");

return 0;

}

**从n\*n矩形中求和最大的矩形：**

int num[110][110],dp[110][110];

int main(){

int n,i,j;

while(scanf("%d",&n)!=EOF){

memset(dp,0,sizeof(dp));

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

{

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

{

scanf("%d",&num[i][j]);

}

}

int max=0,sum;

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

{

for ( j = i; j < n; ++j) //从i行到j行

{

sum=0;

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

{

int temp=0;

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

{

temp+=num[k][m]; //竖着加某一列

}

sum+=temp;

if(sum<0)sum=0; //当加到小于的时候置为0，重新开始加。

else if(sum>max)max=sum;

}

}

}

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

}

return 0;

}

### **[求数组的最长递减子序列](http://nanti.jisuanke.com/t/142)：**

int num[1005];

int dp[1005];

int pre[1005];

int main(){

int n,i,j;

while(scanf("%d",&n)!=EOF){

memset(dp,1,sizeof(dp));

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

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

{

scanf("%d",&num[i]);

}

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

{

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

{

if(num[i]<num[j]&&dp[j]+1>dp[i]){

dp[i]=dp[j]+1;

pre[i]=j;

}

}

}

int maxlen=0;

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

{

if (maxlen<dp[i])

{

maxlen=dp[i];

j=i;

}

}

stack<int> v;

while(j!=-1){

v.push(num[j]);

j=pre[j];

}

printf("%d", v.top());

v.pop();

while(!v.empty()){

printf(" %d", v.top());

v.pop();

}

printf("\n");

}

return 0;

}

**01背包：**

int vol[1010],val[1010],dp[1010];

int main(){

int t,n,v,i,j;

while(scanf("%d",&t)!=EOF){

while(t--){

memset(dp,0,sizeof(dp));

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

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

{

scanf("%d",&val[i]);

}

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

{

scanf("%d",&vol[i]);

}

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

{

for ( j = v; j >=vol[i]; j--)

{

dp[j]=max(dp[j],dp[j-vol[i]]+val[i]);

}

}

printf("%d\n", dp[v]);

}

}

return 0;

}