```
//AC 自动机
struct trie{
	trie *next[4];
	trie *fail;
	bool isend;
};
void insert(char s[]) {
	trie *now=root;
	for (;;) {
		if (s[0]==0) {
			now->isend=1;
			return;
		}
		int tt=s[0]-'0';
		if (now->next[tt]==NULL) now->next[tt]=++head;
		now=now->next[tt];
		s++;
	}
}
void buildFaliure() {
	queue<trie*> q;
	for (int i=0;i<4;i++)
	if (root->next[i]) {
		root->next[i]->fail=root;
		q.push(root->next[i]);
	} else root->next[i]=root;
	while (!q.empty()) {
		trie *now=q.front(); q.pop();
		for (int i=0;i<4;i++) {
			trie *u=now->next[i];
			if (u) {
				q.push(u);
				trie *v=now->fail;
				while (v->next[i]==NULL)
					v=v->fail;
				u->fail=v->next[i];
			}
		}
		if (now->fail->isend) now->isend=1;
	}
}
trie* go(trie *now,char ch) {
	ch-='0';
	trie *ans=now;
	while (ans->next[ch]==NULL)
		ans=ans->fail;
	return ans->next[ch];
}
```

```
//nlogn Dijkstra
struct node{
    int dist,n;
    node(int x,int y){
        n=x; dist=y;
    }
    bool operator < (const node &t) const {
        return dist>t.dist;
    }
};

int a[1010][2000];
int b[1010][2000];
int dist[1010];

int main(){
    int n,m;
    scanf("%d%d",&m,&n);
    while (m--){
        int f,t,cost;
        scanf("%d%d%d",&f,&t,&cost);
        a[f][++a[f][0]]=t;
        b[f][++b[f][0]]=cost;
        a[t][++a[t][0]]=f;
        b[t][++b[t][0]]=cost;
    }
    memset(dist,63,sizeof(dist));
    priority_queue<node> q;
    dist[n]=0; q.push(node(n,0));
    while (!q.empty()&&!visit[1]){
        int v=q.top().n;
        int d=q.top().dist;
        q.pop();
        if (d<=dist[v]) {
            for (int i=1;i<=a[v][0];i++)
            if (dist[a[v][i]]>dist[v]+b[v][i]){
                dist[a[v][i]]=dist[v]+b[v][i];
                q.push(node(a[v][i],dist[a[v][i]]));
            }
        }
    }
    printf("%d\n",dist[1]);
    return 0;
}
```

```
//KMP
void init(char s[],int next[],int n)
{
	next[0]=next[1]=0;
	for (int i=2;i<=n;i++)
	{
		int j=next[i-1];
		while (j>0)
		{
			if (s[j]==s[i-1]) break;
			j=next[j];
		}
		if (s[j]==s[i-1]) j++;
		next[i]=j;
	}
}
int main()
{
	int l;
	while (scanf("%d",&l)!=EOF)
	{
		char *s=new char[l+10];
		scanf("%s",s);

		int *next=new int[l+10];
		init(s,next,l);
		getchar();
		int j=0; int pos=0;
		bool ans=0;
		for (;;)
		{
			char c=getchar();
			if (c==10) break;
			while (j>0 && s[j]!=c) j=next[j];
			if (s[j]==c) j++;
			if (j==l)
			{
				printf("%d\n",pos-l+1);
				ans=1;
				j=next[j];
			}
			pos++;
		}
		if (!ans) printf("\n");
		delete []s;
		delete []next;
	}
	return 0;
}
```

```
//SA
#define maxn 1000001
int wa[maxn],wb[maxn],wv[maxn],ws[maxn];
int cmp(int *r,int a,int b,int l)
{return r[a]==r[b]&&r[a+l]==r[b+l];}
void da(int *r,int *sa,int n,int m)//n+1,m:字符集大小，sa：[1,N]
{
     int i,j,p,*x=wa,*y=wb,*t;
     for(i=0;i<m;i++) ws[i]=0;
     for(i=0;i<n;i++) ws[x[i]=r[i]]++;
     for(i=1;i<m;i++) ws[i]+=ws[i-1];
     for(i=n-1;i>=0;i--) sa[--ws[x[i]]]=i;
     for(j=1,p=1;p<n;j*=2,m=p)
     {
       for(p=0,i=n-j;i<n;i++) y[p++]=i;
       for(i=0;i<n;i++) if(sa[i]>=j) y[p++]=sa[i]-j;
       for(i=0;i<n;i++) wv[i]=x[y[i]];
       for(i=0;i<m;i++) ws[i]=0;
       for(i=0;i<n;i++) ws[wv[i]]++;
       for(i=1;i<m;i++) ws[i]+=ws[i-1];
       for(i=n-1;i>=0;i--) sa[--ws[wv[i]]]=y[i];
       for(t=x,x=y,y=t,p=1,x[sa[0]]=0,i=1;i<n;i++)
       x[sa[i]]=cmp(y,sa[i-1],sa[i],j)?p-1:p++;
     }
     return;
}
int rank[maxn],height[maxn];
void calheight(int *r,int *sa,int n)
{
     int i,j,k=0;
     for(i=1;i<=n;i++) rank[sa[i]]=i;
     for(i=0;i<n;height[rank[i++]]=k)
     for(k?k--:0,j=sa[rank[i]-1];r[i+k]==r[j+k];k++);
     return;
}
```

```
int RMQ[maxn];
int mm[maxn];
int best[20][maxn];
void initRMQ(int n)
{
     int i,j,a,b;
     for(mm[0]=-1,i=1;i<=n;i++)
     mm[i]=((i&(i-1))==0)?mm[i-1]+1:mm[i-1];
     for(i=1;i<=n;i++) best[0][i]=i;
     for(i=1;i<=mm[n];i++)
     for(j=1;j<=n+1-(1<<i);j++)
     {
       a=best[i-1][j];
       b=best[i-1][j+(1<<(i-1))];
       if(RMQ[a]<RMQ[b]) best[i][j]=a;
       else best[i][j]=b;
     }
     return;
}
int askRMQ(int a,int b)
{
    int t;
    t=mm[b-a+1];b-=(1<<t)-1;
    a=best[t][a];b=best[t][b];
    return RMQ[a]<RMQ[b]?a:b;
}
int lcp(int a,int b)
{
    int t;
    a=rank[a];b=rank[b];
    if(a>b) {t=a;a=b;b=t;}
    return(height[askRMQ(a+1,b)]);
}
```

From：POJ1144，割点、割边

```
#include <cstdio>
#include <cstring>
#include <algorithm>

using namespace std;

bool a[110][110];
int visit[110];
int deep[110];
int back[110];
bool cut[110];
int n,ans;


void dfs(int k,int fa,int d)
{
    visit[k]=1;
    back[k]=deep[k]=d;
    int tot=0;
    for (int i=1;i<=n;i++)
    {
        if (a[k][i] && i!=fa && visit[i]==1)
            back[k]=min(back[k],deep[i]);
        if (a[k][i] && visit[i]==0)
        {
            dfs(i,k,d+1);
            tot++;
            back[k]=min(back[k],back[i]);
            if ((k==1 &&tot>1) || (k!=1 && back[i]>=deep[k]))
                if (!cut[k])
                {
                    cut[k]=1;
                    ans++;
                }
          //if back[i]>deep[k] k,i is bridge;
        }
    }
    visit[k]=2;
}
```

```
int main()
{
    while (1)
    {
        scanf("%d",&n);
        if (n==0)
            break;
        memset(a,0,sizeof(a));
        memset(back,0,sizeof(back));
        memset(cut,0,sizeof(cut));
        memset(deep,0,sizeof(deep));
        memset(visit,0,sizeof(visit));
        ans=0;
        int f;
        while (scanf("%d",&f) && f>0)
        {
            while (getchar()!=10)
            {
                int t;
                scanf("%d",&t);
                a[f][t]=a[t][f]=1;
            }
        }
        dfs(1,0,0);
        printf("%d\n",ans);
    }
    return 0;
}
```

From:POJ3041，二分图

```
#include <cstdio>
#include <cstring>

bool a[1010][1010];
bool visit[1010];
int match[1010];
int n;

bool dfs(int k)
{
      for (int i=1;i<=n+n;i++)
      if (!visit[i]&&a[k][i])
      {
            visit[i]=1;
            int tt=match[i];
            match[i]=k;
            if (tt==0||dfs(tt)) return 1;
            match[i]=tt;
      }
      return 0;
}

int main()
{
    int m;
    scanf("%d%d",&n,&m);
    while (m--)
    {
        int x,y;
        scanf("%d%d",&x,&y);
        a[x][y+n]=a[y+n][x]=1;
    }
    int ans=0;
    for (int i=1;i<=n;i++)
    {
        memset(visit,0,sizeof(visit));
        if (dfs(i))
            ans++;
    }
    printf("%d\n",ans);
    return 0;
}
```

From：POJ2299，逆序对

```
#include <cstdio>

int a[500010];
int t[500010];
long long ans;

void merge(int a[],int sizea,int b[],int sizeb)
{
    int nowa=0;
    int nowb=0;
    int s=0;
    while (nowa<sizea&&nowb<sizeb)
    {
        if (a[nowa]<=b[nowb])
            t[s++]=a[nowa++];
        else
        if (a[nowa]>b[nowb])
        {
            t[s++]=b[nowb++];
            ans+=sizea-nowa;
        }
    }
    while (nowa<sizea)
        t[s++]=a[nowa++];
    while (nowb<sizeb)
        t[s++]=b[nowb++];
}

void sort(int a[],int size)
{
    if (size<2)
        return;
    int lsize=size>>1;
    int rsize=size-lsize;
    sort(a,lsize);
    sort(a+lsize,rsize);
    merge(a,lsize,a+lsize,rsize);
    for (int i=0;i<size;i++)
        a[i]=t[i];
}
```

杂

```
void gcd(int a,int b, int &d, int &x, int &y)
{
      if (b==0)
      {
            x=1;
            y=0;
            d=a;
      }
      else
      {
            int x1,y1;
            gcd(b,a%b,d,x1,y1);
            x=y1;
            y=x1-(a/b)*y1;
      }
}

int elfhash(char *key)
{
    unsigned long h=0;
    while(*key)
    {
        h=(h<<4)+*key++;
        unsigned long g=h&0Xf0000000L;
        if(g) h^=g>>24;
        h&=~g;
    }
    return h%MOD;
}
```

BIT:

```
int sum(int k)
{
    int ans = 0;
    for (int i=k;i>0;i-=i&-i)
        ans += a[i];
    return ans;
}

void change(int k,int n,int delta)
{
    for (int i=k;i<=n;i+=i&-i)//小心i=0死循环
        a[i]+=delta;
}
```

```
//POJ2195 新最小费用流
int n,m,ans,t,f;
int maxf[210][210],flow[210][210],dist[210][210];
int fa[210],cost[210];
bool inque[210];
inline int abs(int a) {return a>0?a:-a;}
void init()
{
	int a[210][2]={0},b[210][2]={0},s=0,sa=0,sb=0;
	memset(maxf,0,sizeof(maxf));
	memset(flow,0,sizeof(flow));
	memset(dist,0,sizeof(dist));
	for (int i=1;i<=n;i++)
	for (int j=1;j<=m;j++)
	{
		char tt;
		cin>>tt;
		if (tt=='H')
		{
			a[++sa][0]=i;
			a[sa][1]=j;
		}
		if (tt=='m')
		{
			b[++sb][0]=i;
			b[sb][1]=j;
		}
	}
	s=sa;
	for (int i=1;i<=s;i++)
	for (int j=1;j<=s;j++)
	{
		dist[i][s+j]=abs(a[i][0]-b[j][0])+abs(a[i][1]-b[j][1]);
		dist[s+j][i]=dist[i][s+j];
		maxf[i][s+j]=1;
	}
	for (int i=1;i<=s;i++)
		maxf[0][i]=maxf[s+i][s+s+1]=1;
	t=s+s+1;
	f=0;
	ans=0;
}

inline int value(int i,int j){
	return flow[j][i]>0?-dist[i][j]:dist[i][j];
}
```

```
bool spfamark()
{
	memset(fa,0,sizeof(fa));
	memset(inque,0,sizeof(inque));
	for (int i=1;i<=t;i++)
		cost[i]=2000000000;
	queue<int> q;
	q.push(f); inque[f]=1; cost[f]=0;
	while (!q.empty())
	{
		int tt=q.front(); q.pop(); inque[tt]=0;
		for (int i=0;i<=t;i++)
			if ((maxf[tt][i]-flow[tt][i])&&cost[tt]
+value(tt,i)<cost[i])
			{
				cost[i]=cost[tt]+value(tt,i);
				fa[i]=tt;
				if (!inque[i])
				{
					inque[i]=1;
					q.push(i);
				}
			}
	}
	return cost[t]<2000000000;
}

void change(){
	for(int tt=t;tt!=f;tt=fa[tt]){
		ans+=value(fa[tt],tt);
		flow[fa[tt]][tt]++;
		flow[tt][fa[tt]]--;
	}
}

int main(){
	while (cin>>n>>m&&n&&m)  {
		init();
		while (spfamark())
			change();
		cout<<ans<<endl;
	}
	return 0;
}
```

```
//状态压缩之棋盘放车
long long dp[1<<20];
int line[20];
int pow[21];

inline int getbit(int x)
{
	int ans=0;
	while (x)
	{
		ans++; x-=(x&-x);
	}
	return ans;
}
int main()
{
	for (int i=0;i<=20;i++)
		pow[i]=1<<i;
	int nn;
	scanf("%d",&nn);
	while (nn--)
	{
		int n;
		scanf("%d",&n);
		memset(line,0,sizeof(line));
		for (int i=0;i<n;i++)
			for (int j=0;j<n;j++)
			{
				int t;
				scanf("%d",&t);
				if (t)
					line[i]+=pow[j];
			}
		dp[0]=1;
		for (int i=1;i<pow[n];i++)
		{
			dp[i]=0;
			int bit=getbit(i);
			for (int t=i&line[bit-1],j=t&-t;j>0;t-=j,j=t&-t)
				dp[i]+=dp[i^j];
		}
		printf("%lld\n",dp[pow[n]-1]);
	}
	return 0;
}
```

面积并：From：POI01 火星地图

```
class segment
{
public:
    int l,r,cover,length;
    segment *lc,*rc;
    segment(int L,int R)
    {
        l=L; r=R; cover=0; length=0;
        if (l<r)
        {
            int m=(L+R)>>1;
            lc=new segment(L,m);
            rc=new segment(m+1,R);
        }
    }
    void insert(int L,int R,int delta)
    {
        if (L<=l&&r<=R)
            cover+=delta;
        else
        {
            if (L<=lc->r)
                lc->insert(L,R,delta);
            if (R>=rc->l)
                rc->insert(L,R,delta);
        }
        if (cover)
            length=r-l+1;
        else
            if (l<r)
                length=lc->length+rc->length;
            else
                length=0;
    }
    int count()
    {
            return length;
    }
};
struct line
{
    int x,y1,y2;
    bool operator <(const line &b) const
    {
        return x<b.x;
    }
};
```

```
segment a(0,30000);
line st[10010],ed[10010];
int x[20010];
int n;

int main()
{
    scanf("%d",&n);
    for (int i=0;i<n;i++)
    {
        int x1,y1,x2,y2;
        scanf("%d%d%d%d",&x1,&y1,&x2,&y2);
        st[i].x=x1;st[i].y1=y1;st[i].y2=y2;
        ed[i].x=x2;ed[i].y1=y1;ed[i].y2=y2;
        x[i]=x1;
        x[i+n]=x2;
    }
    sort(x,x+n+n);
    sort(st,st+n);
    sort(ed,ed+n);
    long long ans=0;
    int ST=0,ED=0;
    for (int i=0;i<n+n;i++)
    {
        if (i)
        {
            if (x[i]==x[i-1])
                continue;
            ans+=a.count()*(x[i]-x[i-1]);
        }
        for (;ST<n&&st[ST].x==x[i];ST++)
            a.insert(st[ST].y1,st[ST].y2-1,1);
        for (;ED<n&&ed[ED].x==x[i];ED++)
            a.insert(ed[ED].y1,ed[ED].y2-1,-1);
    }
    printf("%d\n",ans);
    return 0;
}
```

周长并：POJ1177

```
line inX[10010]; line ouX[10010]; line inY[10010]; line ouY[10010];
int n; int ans=0;
void work(line in[],line ou[])
{
     int y=-10000; int i,j;
     for (i=0,j=0;i<n&&j<n;)
     {
         while (y<in[i].y && y<ou[j].y)  y++;
         for (;i<n && in[i].y==y;i++)
         {
               int last=root.length;
               root.insert(in[i].x1,in[i].x2-1,1);
               ans+=abs(root.length-last);
         }
         for (;j<n && ou[j].y==y;j++)
         {
               int last=root.length;
               root.insert(ou[j].x1,ou[j].x2-1,-1);
               ans+=abs(last-root.length);
         }
     }
     for (;j<n;j++)
     {
         int last=root.length;
         root.insert(ou[j].x1,ou[j].x2-1,-1);
         ans+=abs(last-root.length);
     }
}
int main()
{
    scanf("%d",&n);
    for (int i=0;i<n;i++)
    {
        int x1,y1,x2,y2;
        scanf("%d%d%d%d",&x1,&y1,&x2,&y2);
        //x2>x1,y2>y1
        inX[i].x1=x1; inX[i].x2=x2; inX[i].y=y1;
        ouX[i].x1=x1; ouX[i].x2=x2; ouX[i].y=y2;
        inY[i].x1=y1; inY[i].x2=y2; inY[i].y=x1;
        ouY[i].x1=y1; ouY[i].x2=y2; ouY[i].y=x2;
    }
    sort(inX,inX+n); sort(inY,inY+n);
    sort(ouX,ouX+n); sort(ouY,ouY+n);
    work(inX,ouX);
    work(inY,ouY);
    printf("%d\n",ans);
    return 0;
}
```

```
//source：POJ1273 预流推进，n^3
const int inf=2000000000;
int c[210][210];
int f[210][210];
int e[210];
int h[210];
int S,T;
queue<int> q;
bool inque[210];
void init(){
    h[S]=T;
    e[S]=inf;
    for (int i=S;i<=T;i++)
        if (c[S][i])
        {
            f[S][i]=c[S][i];
            f[i][S]=-c[S][i];
            e[i]+=c[S][i];
            e[S]-=c[S][i];
            if (i!=S&&i!=T)
            {
                q.push(i);
                inque[i]=1;
            }
        }
}
void push(int k){
    for (int i=S;i<=T&&e[k]>0;i++) {
        if (c[k][i]-f[k][i]>0&&h[k]==h[i]+1) {
            int delta=min(e[k],c[k][i]-f[k][i]);
            f[k][i]+=delta;
            f[i][k]-=delta;
            e[k]-=delta;
            e[i]+=delta;
            if (!inque[i]&&i!=S&&i!=T&&e[i]>0){
                inque[i]=1;
                q.push(i);
            }
        }
    }
}
void relable(int k)
{
     int tmp=inf;
     for(int i=S;i<=T;i++)
          if(i!=k&&h[i]<tmp&&c[k][i]-f[k][i]>0)
               tmp=h[i];
        h[k]=tmp+1;
}
```

```
int main()
{
    int n,m;
    while (scanf("%d%d",&m,&n)!=EOF)
    {
        memset(c,0,sizeof(c));
        memset(f,0,sizeof(f));
        memset(e,0,sizeof(e));
        memset(h,0,sizeof(h));
        memset(inque,0,sizeof(inque));
        S=1;T=n;
        while (m--)
        {
            int f,t,w;
            scanf("%d%d%d",&f,&t,&w);
            c[f][t]+=w;
        }
        init();
        while (!q.empty())
        {
            int tt=q.front(); q.pop();
            while (e[tt])
            {
                push(tt);
                if (e[tt])
                    relable(tt);
            }
            inque[tt]=0;
        }
        printf("%d\n",e[T]);
    }
    return 0;
}
```