

1 Computational Geometry

1.1 Geometry

```

1 const double PI=atan2(0.0,-1.0);
2 template<typename T>
3 struct point{
4     T x,y;
5     point(){}
6     point(const T&x,const T&y):x(x),y(y){}
7     point operator+(const point &b)const{
8         return point(x+b.x,y+b.y); }
9     point operator-(const point &b)const{
10        return point(x-b.x,y-b.y); }
11    point operator*(const T &b)const{
12        return point(x*b,y*b); }
13    point operator/(const T &b)const{
14        return point(x/b,y/b); }
15    bool operator==(const point &b)const{
16        return x==b.x&&y==b.y; }
17    T dot(const point &b)const{
18        return x*b.x+y*b.y; }
19    T cross(const point &b)const{
20        return x*b.y-y*b.x; }
21    point normal()const{//求法向量
22        return point(-y,x); }
23    T abs2()const{//向量長度的平方
24        return dot(*this); }
25    T rad(const point &b)const{//兩向量的弧
        度
26    return fabs(atan2(fabs(cross(b)),dot(b)))
        ; }
27    T getA()const{//對x軸的弧度
28    T A=atan2(y,x);//超過180度會變負的
29    if(A<=-PI/2)A+=PI*2;
30    return A;
31    }
32 };
33 template<typename T>
34 struct line{
35     line(){}
36     point<T> p1,p2;
37     T a,b,c;//ax+by+c=0
38     line(const point<T>&x,const point<T>&y)
39         :p1(x),p2(y){}
40     void pton()const{//轉成一般式
41         a=p1.y-p2.y;
42         b=p2.x-p1.x;
43         c=-a*p1.x-b*p1.y;
44     }
45     T ori(const point<T> &p)const{//點和有
        向直線的關係，>0左邊、=0在線上<0右
        邊
46     return (p2-p1).cross(p-p1);
47     }
48     T btw(const point<T> &p)const{//點投影
        落在線段上<=0
49     return (p1-p).dot(p2-p);
50     }
51     bool point_on_segment(const point<T>&p)
52         const{//點是否在線段上
53     return ori(p)==0&&btw(p)<=0;
54     }
55     T dis2(const point<T> &p,bool
56         is_segment=0)const{//點跟直線/線段
        的距離平方
57     point<T> v=p2-p1,v1=p-p1;
58     if(is_segment){
59         point<T> v2=p-p2;
60         if(v.dot(v1)<=0)return v1.abs2();
61         if(v.dot(v2)>=0)return v2.abs2();
62     }
63     T tmp=v.cross(v1);
64     return tmp*tmp/v.abs2();
65     }
66     T seg_dis2(const line<T> &l)const{//兩
        線段距離平方
67     return min({dis2(l.p1,1),dis2(l.p2,1)
68         ,l.dis2(p1,1),l.dis2(p2,1)});
69     }
70     point<T> projection(const point<T> &p)
71         const{//點對直線的投影
72     point<T> n=(p2-p1).normal();
73     return p-n*(p-p1).dot(n)/n.abs2();
74     }
75     point<T> mirror(const point<T> &p)const
76         {
77     //點對直線的鏡射，要先呼叫pton轉成一
78     般式
79     point<T> R;
80     T d=a*b+b*b;
81     R.x=(b*b*p.x-a*a*p.x-2*a*b*p.y-2*a*c)/d;
82     R.y=(a*a*p.y-b*b*p.y-2*a*b*p.x-2*b*c)/d;
83     return R;
84     }
85     bool equal(const line &l)const{//直線相
        等
86     return ori(l.p1)==0&&ori(l.p2)==0;
87     }
88     bool parallel(const line &l)const{
89         return (p1-p2).cross(l.p1-l.p2)==0;
90     }
91     bool cross_seg(const line &l)const{
92         return (p2-p1).cross(l.p1-p1)*(p2-p1)
93             .cross(l.p2-p1)<=0;//直線是否交
94             線段
95     }
96     int line_intersect(const line &l)const{
97         //直線相交情況，-1無限多點、1交於
98         一點、0不相交
99     return parallel(l)?(ori(l.p1)
100        ==0?-1:0):1;
101     }
102     int seg_intersect(const line &l)const{
103         T c1=ori(l.p1), c2=ori(l.p2);
104         T c3=l.ori(p1), c4=l.ori(p2);
105         if(c1==0&&c2==0){//共線
106             bool b1=btw(l.p1)>=0,b2=btw(l.p2)
107                 >=0;
108             T a3=l.btw(p1),a4=l.btw(p2);
109             if(b1&&b2&&a3==0&&a4==0) return 2;
110             if(b1&&b2&&a3>=0&&a4==0) return 3;
111             if(b1&&b2&&a3>=0&&a4>=0) return 0;
112             return -1;//無限交點
113         }else if(c1*c2<=0&&c3*c4<=0)return 1;
114         return 0;//不相交
115     }
116     point<T> line_intersection(const line &
117         l)const{//直線交點*/
118     point<T> a=p2-p1,b=l.p2-l.p1,s=l.p1-
119         p1;
120     //if(a.cross(b)==0)return INF;
121     return p1+a*(s.cross(b)/a.cross(b));
122     }
123     point<T> seg_intersection(const line &l
124         )const{//線段交點
125     int res=seg_intersect(l);
126     if(res<=0) assert(0);
127     if(res==2) return p1;
128     if(res==3) return p2;
129     return line_intersection(l);
130     }
131 };
132 template<typename T>
133 struct polygon{
134     polygon(){}
135     vector<point<T>> p;//逆時針順序
136     T area()const{//面積
137     T ans=0;
138     for(int i=p.size()-1,j=0;j<(int)p.
139         size();i=j++){
140         ans+=p[i].cross(p[j]);
141     }
142     return ans/2;
143     }
144     point<T> center_of_mass()const{//重心
145     T cx=0,cy=0,w=0;
146     for(int i=p.size()-1,j=0;j<(int)p.
147         size();i=j++){
148         T a=p[i].cross(p[j]);
149         cx+=(p[i].x+p[j].x)*a;
150         cy+=(p[i].y+p[j].y)*a;
151         w+=a;
152     }
153     return point<T>(cx/3/w,cy/3/w);
154     }
155     char ahas(const point<T>& t)const{//點
156         是否在簡單多邊形內，是的話回傳1、
157         在邊上回傳-1、否則回傳0
158     bool c=0;
159     for(int i=0,j=p.size()-1;i<p.size();j
160         =i++){
161         if(line<T>(p[i],p[j]).
162             point_on_segment(t))return -1;
163         else if((p[i].y>t.y)!=p[j].y>t.y)
164             &&
165             t.x<(p[j].x-p[i].x)*(t.y-p[i].y)/(p
166                 [j].y-p[i].y)+p[i].x)
167             c=!c;
168     }
169     return c;
170     }
171     char point_in_convex(const point<T>&x)
172         const{
173     int l=1,r=(int)p.size()-2;
174     while(l<=r){//點是否在凸多邊形內，是
175         的話回傳1、在邊上回傳-1、否則回
176         傳0
177         int mid=(l+r)/2;
178         T a1=(p[mid]-p[0]).cross(x-p[0]);
179         T a2=(p[mid+1]-p[0]).cross(x-p[0]);
180         if(a1>=0&&a2<=0){
181             T res=(p[mid+1]-p[mid]).cross(x-p
182                 [mid]);
183             return res>0?1:(res==0?-1:0);
184         }else if(a1<0)r=mid-1;
185         else l=mid+1;
186     }
187     return 0;
188     }
189     vector<T> getA()const{//凸包邊對x軸的夾
190         角
191     vector<T> res;//一定是遞增的
192     for(size_t i=0;i<p.size();++i)
193         res.push_back((p[(i+1)%p.size()]-p[
194             i]).getA());
195     return res;
196     }
197     bool line_intersect(const vector<T>&A,
198         const line<T> &l)const{//O(LogN)
199     int f1=upper_bound(A.begin(),A.end(),
200         ,(l.p1-l.p2).getA())-A.begin();
201     int f2=upper_bound(A.begin(),A.end(),
202         ,(l.p2-l.p1).getA())-A.begin();
203     return l.cross_seg(line<T>(p[f1],p[f2
204         ]));
205     }
206     polygon cut(const line<T> &l)const{//凸
207         包對直線切割，得到直線L左側的凸包
208     polygon ans;
209     for(int n=p.size(),i=n-1,j=0;j<n;i=j
210         ++){
211         if(l.ori(p[i])>=0){
212             ans.p.push_back(p[i]);
213             if(l.ori(p[j])<0)
214                 ans.p.push_back(l.
215                     line_intersection(line<T>(
216                         p[i],p[j])));
217         }else if(l.ori(p[j])>0)
218             ans.p.push_back(l.
219                 line_intersection(line<T>(p[
220                     i],p[j])));
221         }
222     }
223     return ans;
224     }
225     static bool monotone_chain_cmp(const
226         point<T>&a,const point<T>&b){//
227         凸包排序函數
228     return (a.x<b.x)||((a.x==b.x&&a.y<b.y)
229         );
230     }
231     void monotone_chain(vector<point<T>> &
232         s){//凸包
233     sort(s.begin(),s.end(),
234         monotone_chain_cmp);
235     p.resize(s.size()+1);
236     int m=0;
237     for(size_t i=0;i<s.size();++i){
238         while(m>2&&(p[m-1]-p[m-2]).cross(s
239             [i]-p[m-2])<=0)--m;
240         p[m++]=s[i];
241     }
242     for(int i=s.size()-2,t=m+1;i>0;--i){
243         while(m>2&&(p[m-1]-p[m-2]).cross(s
244             [i]-p[m-2])<=0)--m;
245         p[m++]=s[i];
246     }
247     if(s.size()>1)--m;
248     p.resize(m);
249     }
250     T diam()const{//直徑
251     int n=p.size(),t=1;
252     T ans=0;p.push_back(p[0]);
253     for(int i=0;i<n;i++){
254         point<T> now=p[i+1]-p[i];
255         while(now.cross(p[t+1]-p[i])>now.
256             cross(p[t]-p[i]))t=(t+1)%n;
257         ans=max(ans,(p[i]-p[t]).abs2());
258     }
259     return p.pop_back(),ans;
260     }
261     T min_cover_rectangle()const{//最小覆蓋矩形
262     int n=p.size(),t=1,r=1,l;
263     if(n<3)return 0;//也可以做最小周長矩
264     形
265     T ans=1e99;p.push_back(p[0]);
266     for(int i=0;i<n;i++){
267         point<T> now=p[i+1]-p[i];
268         while(now.cross(p[t+1]-p[i])>now.
269             cross(p[t]-p[i]))t=(t+1)%n;
270     }

```

```

217 while(now.dot(p[r+1]-p[i])>now.dot(
218     p[r]-p[i]))r=(r+1)%n;
219 if(!i)l=r;
220 while(now.dot(p[l+1]-p[i])<=now.dot(
221     (p[l]-p[i]))l=(l+1)%n;
222 T d=now.abs2();
223 T tmp=now.cross(p[t]-p[i])*(now.dot(
224     (p[r]-p[i])-now.dot(p[l]-p[i])
225     )/d;
226 ans=min(ans,tmp);
227 }
228 return p.pop_back(),ans;
229 }
230 T dis2(polygon &p1){//凸包最近距離平方
231 vector<point<T>> > &P=p,&Q=p1.p;
232 int n=P.size(),m=Q.size(),l=0,r=0;
233 for(int i=0;i<n;++i)if(P[i].y<P[l].y)l=
234     i;
235 for(int i=0;i<m;++i)if(Q[i].y<Q[r].y)r=
236     i;
237 P.push_back(P[0]),Q.push_back(Q[0]);
238 T ans=1e99;
239 for(int i=0;i<n;++i){
240     while((P[l]-P[l+1]).cross(Q[r+1]-Q[
241         r])<0)r=(r+1)%m;
242     ans=min(ans,line<T>(P[l],P[l+1]).
243         seg_dis2(line<T>(Q[r],Q[r+1]))
244         );
245     l=(l+1)%n;
246 }
247 return P.pop_back(),Q.pop_back(),ans;
248 }
249 static char sign(const point<T>&t){
250     return (t.y==0?t.x:t.y)<0;
251 }
252 static bool angle_cmp(const line<T>& A,
253     const line<T>& B){
254     point<T> a=A.p2-A.p1,b=B.p2-B.p1;
255     return sign(a)<sign(b)||sign(a)==
256         sign(b)&&a.cross(b)>0;
257 }
258 int halfplane_intersection(vector<line<
259     T>> &s){//半平面交
260     sort(s.begin(),s.end(),angle_cmp);
261     //線段左側為該線段半平面
262     int L,R,n=s.size();
263     vector<point<T>> > px(n);
264     vector<line<T>> > q(n);
265     q[L=R=0]=s[0];
266     for(int i=1;i<n;++i){
267         while(L<R&&s[i].ori(px[R-1])<=0)--R;
268         while(L<R&&s[i].ori(px[L])<=0)++L;
269         q[++R]=s[i];
270         if(q[R].parallel(q[R-1])){
271             --R;
272             if(q[R].ori(s[i].p1)>0)q[R]=s[i];
273         }
274         if(L<R)px[R-1]=q[R-1].
275             line_intersection(q[R]);
276     }
277 while(L<R&&q[L].ori(px[R-1])<=0)--R;
278 p.clear();
279 if(R-L==1)return 0;
280 px[R]=q[R].line_intersection(q[L]);
281 for(int i=L;i<R;++i)p.push_back(px[i
282     ]);
283 return R-L+1;
284 }
285 };
286 template<typename T>
287 struct triangle{
288     point<T> a,b,c;
289     triangle(){
290         triangle(const point<T> &a,const point<
291             T> &b,const point<T> &c):a(a),b(b)
292             ,c(c){}
293     }
294     T area(){const{
295         T t=(b-a).cross(c-a)/2;
296         return t>0?t:-t;
297     }
298     point<T> barycenter(){const{//重心
299         return (a+b+c)/3;
300     }
301     point<T> circumcenter(){const{//外心
302         static line<T> u,v;
303         u.p1=(a+b)/2;
304         u.p2=point<T>(u.p1.x-a.y+b.y,u.p1.y+a
305             .x-b.x);
306         v.p1=(a+c)/2;
307         v.p2=point<T>(v.p1.x-a.y+c.y,v.p1.y+a
308             .x-c.x);
309         return u.line_intersection(v);
310     }
311     point<T> incenter(){const{//內心
312         T A=sqrt((b-c).abs2()),B=sqrt((a-c).
313             abs2()),C=sqrt((a-b).abs2());
314         return point<T>(A*a.x+B*b.x+C*c.x,A*a
315             .y+B*b.y+C*c.y)/(A+B+C);
316     }
317     point<T> perpercenter(){const{//垂心
318         return barycenter()*3-circumcenter()
319             *2;
320     }
321     };
322 template<typename T>
323 struct point3D{
324     T x,y,z;
325     point3D(){
326         point3D(const T&x,const T&y,const T&z):
327             x(x),y(y),z(z){}
328     }
329     point3D operator+(const point3D &b)
330         const{
331         return point3D(x+b.x,y+b.y,z+b.z);
332     }
333     point3D operator-(const point3D &b)
334         const{
335         return point3D(x-b.x,y-b.y,z-b.z);
336     }
337     point3D operator*(const T &b)const{
338         return point3D(x*b,y*b,z*b);
339     }
340     point3D operator/(const T &b)const{
341         return point3D(x/b,y/b,z/b);
342     }
343     bool operator==(const point3D &b)const{
344         return x==b.x&&y==b.y&&z==b.z;
345     }
346     T dot(const point3D &b)const{
347         return x*b.x+y*b.y+z*b.z;
348     }
349     point3D cross(const point3D &b)const{
350         return point3D(y*b.z-z*b.y,z*b.x-x*b.
351             z,x*b.y-y*b.x);
352     }
353     T abs2(){const{//向量長度的平方
354         return dot(*this);
355     }
356     T area2(const point3D &b)const{//和b、
357         //原點圍成面積的平方
358         return cross(b).abs2()/4;
359     }
360     };
361 template<typename T>
362 struct line3D{
363     point3D<T> p1,p2;
364     line3D(){
365         line3D(const point3D<T> &p1,const
366             point3D<T> &p2):p1(p1),p2(p2){}
367     }
368     T dis2(const point3D<T> &p,bool
369         is_segment=0)const{//點跟直線/線段
370         //的距離平方
371         point3D<T> v=p2-p1,v1=p-p1;
372         if(is_segment){
373             point3D<T> v2=p-p2;
374             if(v.dot(v1)<=0)return v1.abs2();
375             if(v.dot(v2)>=0)return v2.abs2();
376         }
377         point3D<T> tmp=v.cross(v1);
378         return tmp.abs2()/v.abs2();
379     }
380     pair<point3D<T>,point3D<T>>
381         closest_pair(const line3D<T> &l)
382         const{
383         point3D<T> v1=(p1-p2),v2=(l.p1-l.p2);
384         point3D<T> N=v1.cross(v2),ab(l.p1-l.p1
385             );
386         //if(N.abs2()==0)return NULL;平行或重
387         //合
388         T tmp=N.dot(ab),ans=tmp*tmp/N.abs2();
389         //最近點對距離
390         point3D<T> d1=p2-p1,d2=l.p2-l.p1,D=d1
391             .cross(d2),G=l.p1-p1;
392         T t1=(G.cross(d2)).dot(D)/D.abs2();
393         T t2=(G.cross(d1)).dot(D)/D.abs2();
394         return make_pair(p1+d1*t1,l.p1+d2*t2)
395             ;
396     }
397     bool same_side(const point3D<T> &a,
398         const point3D<T> &b)const{
399         return (p2-p1).cross(a-p1).dot((p2-p1
400             ).cross(b-p1))>0;
401     }
402     };
403 template<typename T>
404 struct plane{
405     point3D<T> p0,n;//平面上的點和法向量
406     plane(){
407         plane(const point3D<T> &p0,const
408             point3D<T> &n):p0(p0),n(n){}
409     }
410     T dis2(const point3D<T> &p)const{//點到
411         //平面距離的平方
412         T tmp=(p-p0).dot(n);
413         return tmp*tmp/n.abs2();
414     }
415     point3D<T> projection(const point3D<T>
416         &p)const{
417         return p-n*(p-p0).dot(n)/n.abs2();
418     }
419     };
420 point3D<T> line_intersection(const
421     line3D<T> &l)const{
422     T tmp=n.dot(l.p2-l.p1);//等於0表示平
423     //行或重合該平面
424     return l.p1+(l.p2-l.p1)*(n.dot(p0-l.
425         p1)/tmp);
426 }
427 line3D<T> plane_intersection(const
428     plane &p1)const{
429     point3D<T> e=n.cross(p1.n),v=n.cross(
430         e);
431     T tmp=p1.n.dot(v);//等於0表示平行或重
432     //合該平面
433     point3D<T> q=p0+(v*(p1.n.dot(p1.p0-p0
434         ))/tmp);
435     return line3D<T>(q,q+e);
436 }
437 };
438 template<typename T>
439 struct triangle3D{
440     point3D<T> a,b,c;
441     triangle3D(){
442         triangle3D(const point3D<T> &a,const
443             point3D<T> &b,const point3D<T> &c):
444             a(a),b(b),c(c){}
445     }
446     bool point_in(const point3D<T> &p)const
447         {
448         //點在該平面上的投影在三角形中
449         return line3D<T>(b,c).same_side(p,a)
450             &&line3D<T>(a,c).same_side(p,b)
451             &&line3D<T>(a,b).same_side(p,c);
452     }
453     };
454 template<typename T>
455 struct tetrahedron{//四面體
456     point3D<T> a,b,c,d;
457     tetrahedron(){
458         tetrahedron(const point3D<T> &a,const
459             point3D<T> &b,const point3D<T> &c,
460             const point3D<T> &d):a(a),b(b),c(c)
461             ,d(d){}
462     }
463     T volume6(){const{//體積的六倍
464         return (d-a).dot((b-a).cross(c-a));
465     }
466     point3D<T> centroid(){const{
467         return (a+b+c+d)/4;
468     }
469     }
470     bool point_in(const point3D<T> &p)const
471         {
472         return triangle3D<T>(a,b,c).point_in(
473             p)&&triangle3D<T>(c,d,a).
474             point_in(p);
475     }
476     };
477 template<typename T>
478 struct convexhull3D{
479     static const int MAXN=1005;
480     struct face{
481         int a,b,c;
482         face(int a,int b,int c):a(a),b(b),c(c)
483             {}
484     };
485     vector<point3D<T>> pt;
486     vector<face> ans;
487     int fid[MAXN][MAXN];
488     void build(){
489         int n=pt.size();
490         ans.clear();
491         memset(fid,0,sizeof(fid));
492         ans.emplace_back(0,1,2);//注意不能共
493         //線
494         ans.emplace_back(2,1,0);
495         int ftop=0;
496         for(int i=3,ftop=1;i<n;++i,++ftop)
497             {
498             vector<face> next;
499             for(auto &f:ans){
500                 T d=(pt[i]-pt[f.a]).dot((pt[f.b]-
501                     pt[f.a]).cross(pt[f.c]-pt[f.
502                         a]));
503                 if(d<0)next.push_back(f);
504                 int ff=0;
505                 if(d>0)ff=ftop;
506                 else if(d<0)ff=-ftop;
507                 fid[f.a][f.b]=fid[f.b][f.c]=fid[f.
508                     c][f.a]=ff;
509             }
510             for(auto &f:ans){
511                 if(fid[f.a][f.b]>0 && fid[f.a][f.
512                     b]!=fid[f.b][f.a])
513                     next.emplace_back(f.a,f.b,i);
514                 if(fid[f.b][f.c]>0 && fid[f.b][f.
515                     c]!=fid[f.c][f.b])
516                     next.emplace_back(f.b,f.c,i);
517                 if(fid[f.c][f.a]>0 && fid[f.c][f.
518                     a]!=fid[f.a][f.c])
519                     next.emplace_back(f.c,f.a,i);
520             }
521             ans=next;
522         }
523     }
524     };

```

```

432     next.emplace_back(f.c,f.a,i);
433 }
434     ans=next;
435 }
436 }
437 point3D<T> centroid()const{
438     point3D<T> res(0,0,0);
439     T vol=0;
440     for(auto &f:ans){
441         T tmp=pt[f.a].dot(pt[f.b].cross(pt[
442             f.c]));
443         res=res+(pt[f.a]+pt[f.b]+pt[f.c])*
444             tmp;
445         vol+=tmp;
446     }
447     return res/(vol*4);
448 }
449 };

```

1.2 SmallestCircle

```

1 using PT=point<T>; using CPT=const PT;
2 PT circumcenter(CPT &a,CPT &b,CPT &c){
3     PT u=b-a, v=c-a;
4     T c1=u.abs2()/2, c2=v.abs2()/2;
5     T d=u.cross(v);
6     return PT(a.x+(v.y*c1-u.y*c2)/d, a.y+(u.
7         x*c2-v.x*c1)/d);
8 }
9 void solve(PT p[],int n,PT &c,T &r2){
10     random_shuffle(p,p+n);
11     c=p[0]; r2=0; // c,r2 = 圓心,半徑平方
12     for(int i=1;i<n;i++){
13         if((p[i]-c).abs2()>r2){
14             c=p[i]; r2=0;
15         }
16         for(int j=0;j<i;j++){
17             if((p[j]-c).abs2()>r2){
18                 c.x=(p[i].x+p[j].x)/2;
19                 c.y=(p[i].y+p[j].y)/2;
20                 r2=(p[j]-c).abs2();
21             }
22         }
23     }
24 }

```

1.3 最近點對

```

1 template<typename _IT=point<T>*>
2 T closest_pair(_IT L, _IT R){
3     if(R-L <= 1) return INF;
4     _IT mid = L+(R-L)/2;
5     T x = mid->x;
6     T d = min(closest_pair(L,mid),
7         closest_pair(mid,R));
8     inplace_merge(L, mid, R, ycmp);
9     static vector<point> b; b.clear();
10     for(auto u=L;u<R;u++){
11         if((u->x-x)*(u->x-x)>=d) continue;
12         for(auto v=b.rbegin();v!=b.rend();v++){
13             T dx=u->x-v->x, dy=u->y-v->y;
14             if(dy*dy>=d) break;
15             d=min(d,dx*dx+dy*dy);
16         }
17     }
18     b.push_back(*u);
19 }
20 T closest_pair(vector<point<T>> &v){
21     sort(v.begin(),v.end(),xcmp);
22     return closest_pair(v.begin(),v.end());
23 }

```

2 Data Structure

2.1 01 背包

```

1 LL dp[101][100001] = {0}; //前i個物品所湊
2 //出重量j的最大價值
3 int main(){
4     good;
5 }

```

```

4 LL j,i,n,w,svalue = 0,sweight = 0;
5 cin >> n >> w;
6 pair<LL,LL> item[n+1]; //weight,value;
7 for(i = 1; i <= n; i++){
8     cin >> item[i].first;
9     for(i = 1; i <= n; i++){
10         cin >> item[i].second;
11     }
12     for(i = 0; i <= n; i++){
13         dp[i][0] = dp[0][i] = 0;
14     }
15     for(i = 1; i <= n; i++){
16         for(j = 1; j <= w; j++){
17             if(item[i].first > j)
18                 dp[i][j] = dp[i-1][j];
19             else
20                 dp[i][j] = max(dp[i-1][j],
21                     item[i].second +
22                     dp[i-1][j-item[i].
23                         first]);
24         }
25     }
26     cout << dp[n][w];
27     return 0;
28 }

```

2.2 binary search

```

1 LL BS(LL left,LL right){
2     if(left+1 >= right) //break condition
3         return -1;
4     LL mid = (left+right)/2;
5     if(arr[mid] == target)
6         return mid;
7     else if(arr[mid] < target){
8         left = mid+1;
9         BS(left,right);
10    }
11    else if(arr[mid] > target){
12        right = mid;
13        BS(left,right);
14    }
15 }

```

2.3 discretization

```

1 map<LL,LL> S;
2 for (LL i=0;i<n;i++){
3     S[a[i]] = 0; // insert a[i] and
4     // set rank=0
5 }
6 LL r=0;
7 for (auto it=S.begin(); it!=S.end();
8     ++it) //traversal and set rank
9     it->second = r++;
10 // replace number with rank
11 for (LL i=0;i<n;i++){
12     a[i] = S.lower_bound(a[i]) ->
13         second;
14     // find() return the iterator,
15     // then take the rank
16     // or S.find(a[i]) -> second;
17 }

```

2.4 half enumeration

```

1 #include<bits/stdc++.h>
2
3 #define good ios_base::sync_with_stdio(0)
4 ;cin.tie(0)
5 typedef long long LL;
6 using namespace std;
7
8 LL sa[1<<18],sb[1<<18],no[1<<18]; //subset
9 //product of a and b
10
11 LL subset(LL num[],LL length,LL product
12     [],LL p){ //pass by pointer
13     LL k = 0,i,j; //count
14     for(i = 0; i < length; i++){
15         for(j = 0; j < k; j++){
16             product[k+j] = (product[j]*
17                 num[i]) % p; //old
18             //product times num[i]
19         }
20         product[k] = num[i]; //for num[i]
21         //itself
22         k += k+1;
23     }
24     return k; //return the size of subset
25 }

```

```

19 }
20
21 LL exp_modp(LL x,LL y,LL p){
22     if(y == 0) return 1;
23     if(y % 2) return (exp_modp(x,y-1,p)*x)
24         % p;
25     else{
26         LL temp = exp_modp(x,y/2,p);
27         return (temp*temp) % p;
28     }
29 }
30 int main(){
31     good;
32     //freopen("file name", "r", stdin);
33     //input redirection
34     LL i,n,p;
35     LL a[30],b[30];
36     cin >> n >> p;
37     int len_a = n/2, len_b = n - len_a;
38     for(i = 0; i < len_a; i++){
39         cin >> a[i];
40     }
41     for(i = 0; i < len_b; i++){
42         cin >> b[i];
43     }
44     LL len_sa = subset(a,len_a,sa,p);
45     LL len_sb = subset(b,len_b,sb,p);
46     sort(sa,sa+len_sa);
47     sort(sb,sb+len_sb);
48
49     LL len_sb2 = 1; //len_sb2 followed by
50     //i below
51     no[0] = 1; //assume not empty (check
52     //later)
53     for(i = 1; i < len_sb; i++){
54         if(sb[i] != sb[i-1]){ //new
55             //element
56             sb[len_sb2] = sb[i];
57             no[len_sb2] = 1;
58             len_sb2++;
59         }
60         else //old element
61             no[len_sb2-1]++;
62     }
63     LL ans = (sb[0] == 1) ? no[0] % p : 0;
64     for(i = 0; i < len_sa; i++){
65         if(sa[i] == 1) ans = (ans+1) % p;
66         LL y = exp_modp(sa[i],p-2,p); //
67         //module inverse
68         int it = lower_bound(sb,sb+
69             len_sb2,y) - sb;
70         if(it < len_sb2 && sb[it] == y){
71             ans = (ans + no[it]) % p;
72         }
73     }
74     cout << ans << '\n';
75     return 0;
76 }

```

2.5 LCS

```

1 int dp[1002][1002],i,j; //text1 前i個 &
2 //text2 前j個
3 for(i = 0; i < 1002; i++){
4     dp[i][0] = 0, dp[0][i] = 0;
5     for(j = 1; j <= text1.size(); j++){
6         //1 base <=
7         for(j = 1; j <= text2.size(); j
8             ++){
9             if(text1[i-1] == text2[j-1])
10                 dp[i][j] = dp[i-1][j-1] + 1;
11             else
12                 dp[i][j] = max(dp[i-1][j],
13                     dp[i][j-1]);
14         }
15     }
16     cout << dp[text1.size()][text2.size()]
17         ;
18 }

```

2.6 LIS

```

1 int main(){
2     good;
3     //freopen("file name", "r", stdin);
4     //input redirection
5     LL n,i,length = 0,num;
6     cin >> n;
7     LL last[RSIZE]; //長度為it的最小可能結
8     //尾
9     for(i = 0; i < n; i++){
10         cin >> num;
11     }
12 }

```

```

9         LL it = lower_bound(last,last+
10             length,num)-last;
11         last[it] = num;
12         if(it == length) length++;
13     }
14     cout << length;
15     return 0;

```

2.7 skew heap

```

1 node *merge(node *a,node *b){
2     if(!a||!b) return a?a:b;
3     if(b->data<a->data) swap(a,b);
4     swap(a->l,a->r);
5     a->l=merge(b,a->l);
6     return a;
7 }

```

2.8 sliding window

```

1 //same size
2 for(i = 0; i < m; i++){//making first
3     window
4     LL color = discret[a[right]];
5     cnt[color]++;
6     if(cnt[color] == 1) n_color++;
7     right++;
8 }
9 while(right < n){
10     if(n_color == m)
11         ans++;
12     LL l_remove = discret[a[left]];
13     cnt[l_remove]--;//remove left one
14     left++;
15     if(cnt[l_remove] == 0) n_color--;
16     LL add = discret[a[right]];
17     cnt[add]++;right++;//add next one
18     if(cnt[add] == 1) n_color++;
19 }

```

2.9 undo disjoint set

```

1 struct DisjointSet {
2     // save() is like recursive
3     // undo() is like return
4     int n, fa[MXN], sz[MXN];
5     vector<pair<int*,int*>> h;
6     vector<int> sp;
7     void init(int tn) {
8         n=tn;
9         for (int i=0; i<n; i++) sz[fa[i]=i]
10             =1;
11         sp.clear(); h.clear();
12     }
13     void assign(int *k, int v) {
14         h.PB({k, *k});
15         *k=v;
16     }
17     void save() { sp.PB(SZ(h)); }
18     void undo() {
19         assert(!sp.empty());
20         int last=sp.back(); sp.pop_back();
21         while (SZ(h)!=last) {
22             auto x=h.back(); h.pop_back();
23             *x.F=x.S;
24         }
25     }
26     int f(int x) {
27         while (fa[x]!=x) x=fa[x];
28         return x;
29     }
30     void uni(int x, int y) {
31         x=f(x); y=f(y);
32         if (x==y) return ;
33         if (sz[x]<sz[y]) swap(x, y);
34         assign(&sz[x], sz[x]+sz[y]);
35         assign(&fa[y], x);
36     }
37 }djs;

```

3 Graph

3.1 BFS

```

1 LL val;//unnecessary
2 bool visited[5000] = {false};
3 vector<LL> graph[5000];
4 void BFS(LL start) {
5     queue<LL> q;
6     q.push(start);
7     visited[start] = true;
8     while (!q.empty()){
9         LL curr = q.front();
10        q.pop();
11        for(auto it: graph[curr]){
12            if(!visited[it]){
13                q.push(it);
14                visited[it] = true;
15            }
16        }
17    }
18 }

```

3.2 DFS

```

1 #include<bits/stdc++.h>
2 #define good ios_base::sync_with_stdio(0)
3 ;cin.tie(0);cout.tie(0)
4 typedef long long LL;
5 using namespace std;
6 int fa[100000],d[100000] = {0};//
7 unnecessary
8 bool visit[100000] = {false};
9 vector<LL> v[100000];
10 void dfs(LL now,LL depth){
11     for(auto x:v[now]){
12         if(!visit[x]){
13             cout << x << ' ';
14             visit[x] = true;
15             d[x] = depth;
16             fa[x] = now;
17             dfs(x,depth+1);
18         }
19     }
20 }
21 int main(){
22     good;
23     LL i,n,a,b;
24     cin >> n;
25     for(i = 0; i < n; i++){
26         cin >> a >> b;
27         v[a].push_back(b);
28         v[b].push_back(a);
29     }
30     dfs(0,1);
31     return 0;
32 }

```

3.3 dijkstra

```

1 #include<bits/stdc++.h>
2 #define good ios_base::sync_with_stdio(0)
3 ;cin.tie(0)
4 #define N 10002
5 #define oo 1000000001//1e9+1
6 typedef long long LL;
7 using namespace std;
8 vector<pair<LL,LL>> adjacent[N];//out
9 LL dis[N],parent[N];
10 bool visit[N] = {false};
11
12 int main(){
13     LL i,n,m;
14     cin >> n >> m;
15     for(i = 0; i < m; i++){
16         LL x,y,w;
17         cin >> x >> y >> w;
18         adjacent[x].push_back({y,w});
19         adjacent[y].push_back({x,w});
20     }
21     //initial
22     LL source = 0;
23     memset(dis,oo,sizeof(dis));
24     memset(parent,-1,sizeof(parent));
25     priority_queue<pair<LL,LL>> PQ;//-dis
26     PQ.push({dis[source] = 0,source});
27     //dijkstra
28     while (!PQ.empty()){
29         auto p = PQ.top();
30         PQ.pop();
31         LL v = p.second;//vertex

```

```

32         if(visit[v]) continue;
33         visit[v] = true;
34         for(auto it : adjacent[v]){
35             LL e = it.first,w = it.second;
36             ;
37             if(w + dis[v] < dis[e]){
38                 dis[e] = w + dis[v];
39                 parent[e] = v;
40                 PQ.push({-dis[e],e});
41             }
42         }
43     }
44     LL maxd = -1,cnt = 0,far;
45     for(i = 0; i < n; i++){
46         if(dis[i] < oo){
47             if(dis[i] > maxd)
48                 maxd = dis[i],far = i;
49         }
50         else
51             cnt++;//for can't reach
52     }
53     cout << maxd << endl << cnt;
54     return 0;
55 }

```

3.4 graphISO

```

1 const int MAXN=1005,K=30;//K要夠大
2 const long long A=3,B=11,C=2,D=19,P=0
3 xdefaced;
4 long long f[K+1][MAXN];
5 vector<int> g[MAXN],rg[MAXN];
6 int n;
7 void init(){
8     for(int i=0;i<n;++i){
9         f[0][i]=1;
10        g[i].clear(), rg[i].clear();
11    }
12 }
13 void add_edge(int u,int v){
14     g[u].push_back(v), rg[v].push_back(u);
15 }
16 long long point_hash(int u){//O(N)
17     for(int t=1;t<=K;++t){
18         for(int i=0;i<n;++i){
19             f[t][i]=f[t-1][i]*A%P;
20             for(int j:g[i])f[t][i]=(f[t][i]+f[t-1][j]*B%P)%P;
21             for(int j:rg[i])f[t][i]=(f[t][i]+f[t-1][j]*C%P)%P;
22             if(i==u)f[t][i]=D;//如果圖太大的
23             //話，把這行刪掉，執行一次後f[K]
24             //就會是所有點的答案
25             f[t][i]=P;
26         }
27     }
28     return f[K][u];
29 }
30 vector<long long> graph_hash(){
31     vector<long long> ans;
32     for(int i=0;i<n;++i)ans.push_back(
33         point_hash(i));//O(N^2)
34     sort(ans.begin(),ans.end());
35     return ans;
36 }

```

3.5 MaximumClique

```

1 struct MaxClique{
2     static const int MAXN=105;
3     int N,ans;
4     int g[MAXN][MAXN],dp[MAXN],stk[MAXN][
5         MAXN];
6     int sol[MAXN],tmp[MAXN];//sol[0~ans-1]
7     //答案
8     void init(int n){
9         N=n;//0-base
10        memset(g,0,sizeof(g));
11    }
12    void add_edge(int u,int v){
13        g[u][v]=g[v][u]=1;
14    }
15    int dfs(int ns,int dep){
16        if(!ns){
17            if(dep>ans){
18                ans=dep;
19                memcpy(sol,tmp,sizeof tmp);
20                return 1;
21            }else return 0;
22        }
23        for(int i=0;i<ns;++i){

```



```

22     if(dep+ns-i<=ans) return 0;
23     int u=stk[dep][i],cnt=0;
24     if(dep+dp[u]<=ans) return 0;
25     for(int j=i+1;j<ns;++j){
26         int v=stk[dep][j];
27         if(g[u][v]) stk[dep+1][cnt++]=v;
28     }
29     tmp[dep]=u;
30     if(dfs(cnt,dep+1)) return 1;
31 }
32 return 0;
33 }
34 int clique(){
35     int u,v,ns;
36     for(ans=0,u=N-1;u>=0;--u){
37         for(ns=0,tmp[0]=u,v=u+1;v<N;++v)
38             if(g[u][v]) stk[1][ns++]=v;
39         dfs(ns,1),dp[u]=ans;
40     }
41     return ans;
42 }
43 };

```

3.6 MinimumMeanCycle

```

1 #include<cstdio> //for DBL_MAX
2 int dp[MAXN][MAXN]; // 1-base,0(NM)
3 vector<tuple<int,int,int>> edge;
4 double mmc(int n){//allow negative weight
5     const int INF=0x3f3f3f3f;
6     for(int t=0;t<n;++t){
7         memset(dp[t+1],0x3f,sizeof(dp[t+1]));
8         for(const auto &e:edge){
9             int u,v,w;
10            tie(u,v,w) = e;
11            dp[t+1][v]=min(dp[t+1][v],dp[t][u]+w);
12        }
13    }
14    double res = DBL_MAX;
15    for(int u=1;u<n;++u){
16        if(dp[n][u]==INF) continue;
17        double val = -DBL_MAX;
18        for(int t=0;t<n;++t)
19            val=max(val,(dp[n][u]-dp[t][u])
20                *1.0/(n-t));
21        res=min(res,val);
22    }
23    return res;

```

3.7 topology sort

```

1 int main(){
2     good;
3     LL indeg[1002] = {0};
4     vector<LL> graph[1002];
5     LL n,m,a,b;
6     cin >> n >> m;
7     for(LL i = 0; i < m; i++){
8         cin >> a >> b;
9         graph[a].push_back(b);
10        indeg[b]++;
11    }
12    LL topo[1002],head = 0,tail = 0;///?
13    queue
14    for(LL i = 0; i < n; i++)
15        if(indeg[i] == 0)
16            topo[tail++] = i;
17    while(head < tail){
18        LL v = topo[head++];//get data
19        and pop
20        for(LL u : graph[v]){
21            if(--indeg[u] == 0)
22                topo[tail++] = u;
23        }
24        if(tail < n) cout << "not a DAG" << endl;
25        else{
26            for(LL i = 0; i < n; i++)
27                cout << topo[i] << ' ';
28        }
29    }

```

3.8 union and find

```

1 #include<bits/stdc++.h>
2 #define good ios_base::sync_with_stdio(0)
3 #define cin.tie(0)
4 #define RSIZE 101
5 using namespace std;
6
7 LL parent[503*503];
8 int graph[503*503] = {0};
9 int dxy[4] = {1,-1};
10 LL now_area = 0,max_area = 0;
11
12 LL sfind(LL dots){//find Leader,Leader's
13     parent = size of set
14     if(parent[dots] < 0)
15         return dots;
16     return parent[dots] = sfind(parent[
17         dots]);
18 }
19 LL BFS(LL now,LL root){//find root and
20     return size
21     parent[now] = root;
22     LL cnt = 1;
23     for(int k = 0; k < 4; k++){//4
24         directions
25         int u = now+dxy[k];
26         if(graph[u] == 1 && parent[u] ==
27             -1){//unvisited
28             cnt += BFS(u,root);
29         }
30     }
31     return cnt;
32 }
33 void combine(LL u,LL v){//merge two sets
34     LL set1 = sfind(u),set2 = sfind(v);
35     if(set1 == set2) return;//same set
36
37     max_area = max(max_area,-parent[set1]
38         -parent[set2]);
39     now_area--;//merge -> 2 pools become
40     1
41     if(parent[set1] < parent[set2]){//1
42         is larger
43         parent[set1] += parent[set2];
44         parent[set2] = set1;
45     }
46     else{
47         parent[set2] += parent[set1];
48         parent[set1] = set2;
49     }
50     return;
51 }
52 int main(){
53     good;
54     //freopen("file name", "r", stdin);
55     //input redirection
56     LL i,j,m,n,k;
57     cin >> m >> n >> k;
58     memset(parent,-1,sizeof(parent));
59     for(i = 1; i <= m; i++){
60         for(j = 1; j <= n; j++){
61             cin >> graph[i*(n+2)+j];
62         }
63     }
64     n += 2;
65     dxy[2] = n,dxy[3] = -n;
66     LL mn = (m+1)*n;
67     for(LL x = n; x < mn; x++){
68         if(graph[x] == 1 && parent[x] ==
69             -1){//unvisited
70             parent[x] = -BFS(x,x);//first
71             point consider as root
72             now_area++;
73             max_area = max(max_area,-
74                 parent[x]);
75         }
76     }
77     LL ans = now_area,max_ans = max_area;
78     while(k--){
79         LL x,y,temp;
80         cin >> x >> y;
81         temp = x*n+y;
82         if(graph[temp] == 1) continue;
83         graph[temp] = 1;
84         now_area++;
85         max_area = max(max_area,(LL)1);
86         for(i = 0; i < 4; i++){
87             if(graph[temp+dxy[i]] == 0)
88                 continue;
89             combine(temp,temp+dxy[i]);
90         }
91         ans += now_area;
92         max_ans += max_area;
93     }
94     cout << max_ans << endl << ans;
95     return 0;
96 }

```

3.9 全局最小割

```

1 const int INF=0x3f3f3f3f;
2 template<typename T>
3 struct stoer_wagner{// 0-base
4     static const int MAXN=150;
5     T g[MAXN][MAXN],dis[MAXN];
6     int nd[MAXN],n,s,t;
7     void init(int _n){
8         n=_n;
9         for(int i=0;i<n;++i)
10             for(int j=0;j<n;++j)g[i][j]=0;
11     }
12     void add_edge(int u,int v,T w){
13         g[u][v]=g[v][u]+=w;
14     }
15     T min_cut(){
16         T ans=INF;
17         for(int i=0;i<n;++i)nd[i]=i;
18         for(int ind,tn=n;tn>1;--tn){
19             for(int i=1;i<tn;++i)dis[ind[i]]=0;
20             for(int i=1;i<tn;++i){
21                 ind=i;
22                 for(int j=i;j<tn;++j){
23                     dis[ind[j]]+=g[ind[i-1]][nd[j]];
24                     if(dis[ind[j]]<dis[ind[i]])ind=
25                         j;
26                 }
27                 swap(ind[ind],nd[i]);
28                 if(ans>dis[ind])ans=dis[t=nd[
29                     ind]],s=nd[ind-1];
30                 for(int i=0;i<tn;++i)
31                     g[ind[ind-1]][nd[i]]=g[ind[i]][nd[
32                         ind-1]]+=g[ind[i]][nd[ind]];
33             }
34             return ans;
35         }
36     };

```

3.10 最小樹形圖朱劉

```

1 template<typename T>
2 struct zhu_liu{
3     static const int MAXN=110,MAXM=10005;
4     struct node{
5         int u,v;
6         T w,tag;
7         node *l,*r;
8         node(int u=0,int v=0,T w=0):u(u),v(v)
9             ,w(w),tag(0),l(0),r(0){}
10        void down(){
11            w+=tag;
12            if(l)l->tag+=tag;
13            if(r)r->tag+=tag;
14            tag=0;
15        }
16    }mem[MAXN];//靜態記憶體
17    node *pq[MAXN*2],*E[MAXN*2];
18    int st[MAXN*2],id[MAXN*2],m;
19    void init(int n){
20        for(int i=1;i<=n;++i){
21            pq[i]=E[i]=0, st[i]=id[i]=i;
22            m=0;
23        }
24        node *merge(node *a,node *b){//skew
25            heap
26            if(!a||!b)return a?a:b;
27            a->down(),b->down();
28            if(b->w<a->w)return merge(b,a);
29            swap(a->l,a->r);
30            a->l=merge(b,a->l);
31            return a;
32        }
33        void add_edge(int u,int v,T w){
34            if(u!=v)pq[v]=merge(pq[v],&(mem[m++] =
35                node(u,v,w)));
36        }
37        int find(int x,int *st){
38            return st[x]==x?x:st[x]=find(st[x],st
39                );
40        }
41        T build(int root,int n){
42            T ans=0;int N=n,all=n;
43            for(int i=1;i<=N;++i){
44                if(i==root||!pq[i])continue;
45                while(pq[i]){
46                    pq[i]->down(),E[i]=pq[i];
47                    pq[i]=merge(pq[i]->l,pq[i]->r);
48                    if(find(E[i]->u,id)!=find(i,id))
49                        break;
50                }
51            }
52        }

```

```

46     if(find(E[i]->u,id)==find(i,id))
47         continue;
48     ans+=E[i]->w;
49     if(find(E[i]->u,st)==find(i,st)){
50         if(pq[i])pq[i]->tag-=E[i]->w;
51         pq[++N]=pq[i];id[N]=N;
52         for(int u=find(E[i]->u,id);u!=i;u
53             =find(E[u]->u,id)){
54             if(pq[u])pq[u]->tag-=E[u]->w;
55             id[find(u,id)]=N;
56             pq[N]=merge(pq[N],pq[u]);
57         }
58         st[N]=find(i,st);
59         id[find(i,id)]=N;
60     }else st[find(i,st)]=find(E[i]->u,
61         st),--all;
62 }
63 return all==1?ans:-INT_MAX;//圖不連通
64 }
65 }
66 };

```

4 Language

4.1 CNF

```

1 #define MAXN 55
2 struct CNF{
3     int s,x,y;//s->xy | s->x, if y==1
4     int cost;
5     CNF(){
6         CNF(int s,int x,int y,int c):s(s),x(x),
7             y(y),cost(c){
8     };
9     int state;//規則數量
10    map<char,int> rule;//每個字元對應到的規
11        則，小寫字母為終端字符
12    vector<CNF> cnf;
13    void init(){
14        state=0;
15        rule.clear();
16        cnf.clear();
17    }
18    void add_to_cnf(char s,const string &p,
19        int cost){
20        //加入一個s -> <p>的文法，代價為cost
21        if(rule.find(s)==rule.end())rule[s]=
22            state++;
23        for(auto c:p)if(rule.find(c)==rule.end
24            ())rule[c]=state++;
25        if(p.size()==1){
26            cnf.push_back(CNF(rule[s],rule[p
27                ][0],-1,cost));
28        }else{
29            int left=rule[s];
30            int sz=p.size();
31            for(int i=0;i<sz-2;++i){
32                cnf.push_back(CNF(left,rule[p[i]],
33                    state,0));
34                left=state++;
35            }
36            cnf.push_back(CNF(left,rule[p[sz-2]],
37                rule[p[sz-1]],cost));
38        }
39    }
40    vector<long long> dp[MAXN][MAXN];
41    vector<bool> neg_INF[MAXN][MAXN];//如果花
42        費是負的可能會有無限小的情形
43    void relax(int l,int r,const CNF &c,long
44        long cost,bool neg_c=0){
45        if(!neg_INF[l][r][c.s]&&(neg_INF[l][r][
46            c.x]||cost<dp[l][r][c.s])){
47            if(neg_c||neg_INF[l][r][c.x]){
48                dp[l][r][c.s]=0;
49                neg_INF[l][r][c.s]=true;
50            }else dp[l][r][c.s]=cost;
51        }
52    }
53    void bellman(int l,int r,int n){
54        for(int k=1;k<=state;++k)
55            for(auto c:cnf)
56                if(c.y==1)relax(l,r,c,dp[l][r][c.x
57                    ]+c.cost,k==n);
58    }
59    void cyk(const vector<int> &tok){
60        for(int i=0;i<(int)tok.size();++i){
61            for(int j=0;j<(int)tok.size();++j){
62                dp[i][j]=vector<long long>(state+1,
63                    INT_MAX);
64                neg_INF[i][j]=vector<bool>(state+1,
65                    false);
66            }
67        }
68    }
69 }

```

```

52     }
53     dp[i][j][tok[i]]=0;
54     bellman(i,i,tok.size());
55 }
56 for(int r=1;r<(int)tok.size();++r){
57     for(int l=r-1;l>=0;--l){
58         for(int k=l;k<r;++k)
59             for(auto c:cnf)
60                 if(~c.y)relax(l,r,c,dp[l][k][c.
61                     x]+dp[k+1][r][c.y]+c.cost)
62                     ;
63         bellman(l,r,tok.size());
64     }
65 }
66 }

```

5 Linear Programming

5.1 simplex

```

1 /*target:
2     max \sum_{j=1}^n A_{0,j}*x_j
3 condition:
4     \sum_{j=1}^n A_{i,j}*x_j <= A_{i,0} | i
5     =1~m
6     x_j >= 0 | j=1~n
7 VDB = vector<double>*/
8 template<class VDB>
9 VDB simplex(int m,int n,vector<VDB> a){
10     vector<int> left(m+1), up(n+1);
11     iota(left.begin(), left.end(), n);
12     iota(up.begin(), up.end(), 0);
13     auto pivot = [&](int x, int y){
14         swap(left[x], up[y]);
15         auto k = a[x][y]; a[x][y] = 1;
16         vector<int> pos;
17         for(int j = 0; j <= n; ++j){
18             a[x][j] /= k;
19             if(a[x][j] != 0) pos.push_back(j);
20         }
21         for(int i = 0; i <= m; ++i){
22             if(a[i][y]==0 || i == x) continue;
23             k = a[i][y], a[i][y] = 0;
24             for(int j : pos) a[i][j] -= k*a[x][j];
25         }
26     };
27     for(int x,y;;){
28         for(int i=x=1; i <= m; ++i)
29             if(a[i][0]<a[x][0]) x = i;
30         if(a[x][0]>=0) break;
31         for(int j=y=1; j <= n; ++j)
32             if(a[x][j]<a[x][y]) y = j;
33         if(a[x][y]>=0) return VDB();//
34             infeasible
35         pivot(x, y);
36     }
37     for(int x,y;;){
38         for(int j=y=1; j <= n; ++j)
39             if(a[0][j] > a[0][y]) y = j;
40         if(a[0][y]<=0) break;
41         x = -1;
42         for(int i=1; i<=m; ++i) if(a[i][y] >
43             0)
44             if(x == -1 || a[i][0]/a[i][y]
45                 < a[x][0]/a[x][y]) x = i;
46         if(x == -1) return VDB();//unbounded
47         pivot(x, y);
48     }
49     VDB ans(n + 1);
50     for(int i = 1; i <= m; ++i)
51         if(left[i] <= n) ans[left[i]] = a[i][0];
52     ans[0] = -a[0][0];
53     return ans;
54 }

```

6 Number Theory

6.1 basic

```

1 template<typename T>
2 void gcd(const T &a,const T &b,T &d,T &x,
3     T &y){
4     if(!b) d=a,x=1,y=0;
5     else gcd(b,a%b,d,y,x), y-=x*(a/b);
6 }

```

```

6 long long int phi[N+1];
7 void phiTable(){
8     for(int i=1;i<=N;i++)phi[i]=i;
9     for(int i=1;i<=N;i++)for(x=i*2;x<=N;x+=
10         i)phi[x]-=phi[i];
11 }
12 void all_divdown(const LL &n) { // all n/x
13     for(LL a=1;a<=n;a=n/(n/(a+1))) {
14         // dosomething;
15     }
16 }
17 const int MAXPRIME = 1000000;
18 int iscom[MAXPRIME], prime[MAXPRIME],
19     primecnt;
20 int phi[MAXPRIME], mu[MAXPRIME];
21 void sieve(void){
22     memset(iscom,0,sizeof(iscom));
23     primecnt = 0;
24     phi[1] = mu[1] = 1;
25     for(int i=2;i<MAXPRIME;++i) {
26         if(!iscom[i]) {
27             prime[primecnt++] = i;
28             mu[i] = -1;
29             phi[i] = i-1;
30         }
31         for(int j=0;j<primecnt;++j) {
32             int k = i * prime[j];
33             if(k>=MAXPRIME) break;
34             iscom[k] = prime[j];
35             if(i%prime[j]==0) {
36                 mu[k] = 0;
37                 phi[k] = phi[i] * prime[j];
38                 break;
39             } else {
40                 mu[k] = -mu[i];
41                 phi[k] = phi[i] * (prime[j]-1);
42             }
43         }
44     }
45 }
46 bool g_test(const LL &g, const LL &p,
47     const vector<LL> &v) {
48     for(int i=0;i<v.size();++i)
49         if(modexp(g,(p-1)/v[i],p)==1)
50             return false;
51     return true;
52 }
53 LL primitive_root(const LL &p) {
54     if(p==2) return 1;
55     vector<LL> v;
56     Factor(p-1,v);
57     v.erase(unique(v.begin(), v.end()), v.
58         end());
59     for(LL g=2;g<p;++g)
60         if(g_test(g,p,v))
61             return g;
62     puts("primitive_root NOT FOUND");
63     return -1;
64 }
65 int Legendre(const LL &a, const LL &p) {
66     return modexp(a%p,(p-1)/2,p);
67 }
68 LL inv(const LL &a, const LL &n) {
69     LL d,x,y;
70     gcd(a,n,d,x,y);
71     return d==1 ? (x+n)%n : -1;
72 }
73 int inv[maxn];
74 LL invtable(int n,LL P){
75     inv[1]=1;
76     for(int i=2;i<n;++i)
77         inv[i]=(P-(P/i))*inv[P%i]%P;
78 }
79 LL log_mod(const LL &a, const LL &b,
80     const LL &p) {
81     // a ^ x = b ( mod p )
82     int m=sqrt(p+.5), e=1;
83     LL v=inv(modexp(a,m,p), p);
84     map<LL,int> x;
85     x[1]=0;
86     for(int i=1;i<m;++i) {
87         e = Llmul(e,a,p);
88         if(!x.count(e)) x[e] = i;
89     }
90     for(int i=0;i<m;++i) {
91         if(x.count(b)) return i*m + x[b];
92         b = Llmul(b,v,p);
93     }
94     return -1;
95 }
96 LL Tonelli-Shanks(const LL &n, const LL &
97     p) {
98     // x^2 = n ( mod p )
99 }

```

```

96 if(n==0) return 0;
97 if(Legendre(n,p)!=1) while(1) { puts("
    Sqrt Root does not exist"); }
98 int S = 0;
99 LL Q = p-1;
100 while( !(Q&1) ) { Q>>=1; ++S; }
101 if(S==1) return modexp(n%p,(p+1)/4,p);
102 LL z = 2;
103 for(; Legendre(z,p)!=-1; ++z)
104 LL c = modexp(z,Q,p);
105 LL R = modexp(n%p,(Q+1)/2,p), t =
    modexp(n%p,Q,p);
106 int M = S;
107 while(1) {
108     if(t==1) return R;
109     LL b = modexp(c,1L<<(M-i-1),p);
110     R = LLMul(R,b,p);
111     t = LLMul( LLMul(b,b,p), t, p);
112     c = LLMul(b,b,p);
113     M = i;
114 }
115 return -1;
116 }
117
118 template<typename T>
119 T Euler(T n){
120     T ans=n;
121     for(T i=2;i*i<=n;++i){
122         if(n%i==0){
123             ans=ans/i*(i-1);
124             while(n%i==0)n/=i;
125         }
126     }
127     if(n>1)ans=ans/n*(n-1);
128     return ans;
129 }
130
131 //Chinese_remainder_theorem
132 template<typename T>
133 T pow_mod(T n,T k,T m){
134     T ans=1;
135     for(n=(n>m?n%m:n);k>>=1){
136         if(k&1)ans=ans*n%m;
137         n=n*n%m;
138     }
139     return ans;
140 }
141
142 template<typename T>
143 T crt(vector<T> &m,vector<T> &a){
144     T M=1,tM,ans=0;
145     for(int i=0;i<(int)m.size();++i)M*=m[i];
146     for(int i=0;i<(int)a.size();++i){
147         tM=M/m[i];
148         ans=(ans+(a[i]*tM%M)*pow_mod(tM,Euler(m[i])-1,m[i])%M)%M;
149         /*如果m[i]是質數 · Euler(m[i])-1=m[i]-2 · 就不用算Euler了*/
150     }
151     return ans;
152 }
153
154 //java code
155 //求sqrt(N)的連分數
156 public static void Pell(int n){
157     BigInteger N,p1,p2,q1,q2,a0,a1,a2,g1,g2,h1,h2,p,q;
158     g1=q2=p1=BigInteger.ZERO;
159     h1=q1=p2=BigInteger.ONE;
160     a0=a1=BigInteger.valueOf((int)Math.sqrt(1.0*n));
161     BigInteger ans=a0.multiply(a0);
162     if(ans.equals(BigInteger.valueOf(n))){
163         System.out.println("No solution!");
164         return;
165     }
166     while(true){
167         g2=a1.multiply(h1).subtract(g1);
168         h2=N.subtract(g2.pow(2)).divide(h1);
169         a2=g2.add(a0).divide(h2);
170         p=a1.multiply(p2).add(p1);
171         q=a1.multiply(q2).add(q1);
172         if(p.pow(2).subtract(N.multiply(q.pow(2))).compareTo(BigInteger.ONE)==0)break;
173         g1=g2;h1=h2;a1=a2;
174         p1=p2;p2=p;
175         q1=q2;q2=q;
176     }
177     System.out.println(p+" "+q);
178 }

```

6.2 bit set

```

1 void sub_set(int S){
2     int sub=S;
3     do{
4         //對某集合的子集合的處理
5         sub=(sub-1)&S;
6     }while(sub!=S);
7 }
8 void k_sub_set(int k,int n){
9     int comb=(1<<k)-1,S=1<<n;
10    while(comb<S){
11        //對大小為k的子集合的處理
12        int x=comb&-comb,y=comb+x;
13        comb=((comb&y)/x>>1)|y;
14    }
15 }

```

6.3 cantor expansion

```

1 int factorial[MAXN];
2 void init(){
3     factorial[0]=1;
4     for(int i=1;i<=MAXN;++i)factorial[i]=
        factorial[i-1]*i;
5 }
6 int encode(const vector<int> &s){
7     int n=s.size(),res=0;
8     for(int i=0;i<n;++i){
9         int t=0;
10        for(int j=i+1;j<n;++j)
11            if(s[j]<s[i])++t;
12        res+=t*factorial[n-i-1];
13    }
14    return res;
15 }
16 vector<int> decode(int a,int n){
17     vector<int> res;
18     vector<bool> vis(n,0);
19     for(int i=n-1;i>=0;--i){
20         int t=a/factorial[i],j;
21         for(j=0;j<n;++j)
22             if(!vis[j]){
23                 if(t==0)break;
24                 --t;
25             }
26         res.push_back(j);
27         vis[j]=1;
28         a%=factorial[i];
29     }
30     return res;
31 }

```

6.4 find real root

```

1 // an*x^n + ... + a1x + a0 = 0;
2 int sign(double x){
3     return x < -eps ? -1 : x > eps;
4 }
5
6 double get(const vector<double> &coef,
7            double x){
8     double e = 1, s = 0;
9     for(auto i : coef) s += i*e, e *= x;
10    return s;
11 }
12 double find(const vector<double> &coef,
13             int n, double lo, double hi){
14     double sign_lo, sign_hi;
15     if( !(sign_lo = sign(get(coef,lo))) )
16         return lo;
17     if( !(sign_hi = sign(get(coef,hi))) )
18         return hi;
19     if(sign_lo * sign_hi > 0) return INF;
20     for(int stp = 0; stp < 100 && hi - lo >
        eps; ++stp){
21         double m = (lo+hi)/2.0;
22         int sign_mid = sign(get(coef,m));
23         if(!sign_mid) return m;
24         if(sign_lo*sign_mid < 0) hi = m;
25         else lo = m;
26     }
27     return (lo+hi)/2.0;
28 }
29 vector<double> cal(vector<double> coef,
30                  int n){
31     vector<double> res;
32     if(n == 1){
33         if(sign(coef[1])) res.pb(-coef[0]/coef[1]);
34     }
35 }

```

```

31     return res;
32 }
33 vector<double> dcoef(n);
34 for(int i = 0; i < n; ++i) dcoef[i] =
    coef[i+1]*(i+1);
35 vector<double> droot = cal(dcoef, n-1);
36 droot.insert(droot.begin(), -INF);
37 droot.pb(INF);
38 for(int i = 0; i+1 < droot.size(); ++i)
39     {
40         double tmp = find(coef, n, droot[i],
41                           droot[i+1]);
42         if(tmp < INF) res.pb(tmp);
43     }
44 return res;
45 }
46
47 int main () {
48     vector<double> ve;
49     vector<double> ans = cal(ve, n);
50     // 視情況把答案 +eps · 避免 -0
51 }

```

6.5 LinearCongruence

```

1 pair<LL,LL> LinearCongruence(LL a[],LL b
2   [],LL m[],int n) {
3     // a[i]*x = b[i] ( mod m[i] )
4     for(int i=0;i<n;++i) {
5         LL x, y, d = extgcd(a[i],m[i],x,y);
6         if(b[i]%d!=0) return make_pair(-1LL,0);
7         LL;
8         m[i] /= d;
9         b[i] = LLMul(b[i]/d,x,m[i]);
10    }
11    LL lastb = b[0], lastm = m[0];
12    for(int i=1;i<n;++i) {
13        LL x, y, d = extgcd(m[i],lastm,x,y);
14        if((lastb-b[i])%d!=0) return
15            make_pair(-1LL,0LL);
16        lastb = LLMul((lastb-b[i])/d,x,(lastm
17            /d))*m[i];
18        lastm = (lastm/d)*m[i];
19        lastb = (lastb+b[i])%lastm;
20    }
21    return make_pair(lastb<0?lastb+lastm:
22        lastb,lastm);
23 }

```

6.6 Lucas

```

1 ll C(ll n, ll m, ll p){ // n!/m!/(n-m)!
2     if(n<m) return 0;
3     return f[n]*inv(f[m],p)%p*inv(f[n-m],p)%p;
4 }
5 ll L(ll n, ll m, ll p){
6     if(!m) return 1;
7     return C(n%p,m%p,p)*L(n/p,m/p,p)%p;
8 }
9 ll Wilson(ll n, ll p){ // n!%p
10    if(!n)return 1;
11    ll res=Wilson(n/p, p);
12    if((n/p)%2) return res*(p-f[n%p])%p;
13    return res*f[n%p]%p; // (p-1)!%p=-1
14 }

```

6.7 Matrix

```

1 template<typename T>
2 struct Matrix{
3     using rt = std::vector<T>;
4     using mt = std::vector<rt>;
5     using matrix = Matrix<T>;
6     int r,c;
7     mt m;
8     Matrix(int r,int c):r(r),c(c),m(r,rt(c)){}
9     rt& operator[](int i){return m[i];}
10    matrix operator+(const matrix &a){
11        matrix rev(r,c);
12        for(int i=0;i<r;++i)
13            for(int j=0;j<c;++j)
14                rev[i][j]=m[i][j]+a.m[i][j];
15        return rev;
16    }
17    matrix operator-(const matrix &a){

```

```

18 matrix rev(r,c);
19 for(int i=0;i<r;++i)
20     for(int j=0;j<c;++j)
21         rev[i][j]=m[i][j]-a.m[i][j];
22 return rev;
23 }
24 matrix operator*(const matrix &a){
25     matrix rev(r,a.c);
26     matrix tmp(a.c,a.r);
27     for(int i=0;i<a.r;++i)
28         for(int j=0;j<a.c;++j)
29             tmp[j][i]=a.m[i][j];
30     for(int i=0;i<r;++i)
31         for(int j=0;j<a.c;++j)
32             for(int k=0;k<c;++k)
33                 rev.m[i][j]+=m[i][k]*tmp[j][k];
34     return rev;
35 }
36 bool inverse(){
37     Matrix t(r,r+c);
38     for(int y=0;y<r;y++){
39         t.m[y][c+y] = 1;
40         for(int x=0;x<c;++x)
41             t.m[y][x]=m[y][x];
42     }
43     if( !t.gas() )
44         return false;
45     for(int y=0;y<r;y++){
46         for(int x=0;x<c;++x)
47             m[y][x]=t.m[y][c+x]/t.m[y][y];
48     }
49     return true;
50 }
51 T gas(){
52     vector<T> lazy(r,1);
53     bool sign=false;
54     for(int i=0;i<r;++i){
55         if( m[i][i]==0 ){
56             int j=i+1;
57             while(j<r&&!m[j][i])j++;
58             if(j==r)continue;
59             m[i].swap(m[j]);
60             sign=!sign;
61         }
62         for(int j=0;j<r;++j){
63             if(i==j)continue;
64             lazy[j]=lazy[j]*m[i][i];
65             T mx=m[j][i];
66             for(int k=0;k<c;++k)
67                 m[j][k]=m[j][k]*m[i][i]-m[i][k]*mx;
68         }
69     }
70     T det=sign?-1:1;
71     for(int i=0;i<r;++i){
72         det = det*m[i][i];
73         det = det/lazy[i];
74     }
75     return det;
76 }
77 };

```

6.8 matrix exponential

```

1 void exp(LL m[2][2], LL x){
2     LL c[2][2] = {{1,1},{1,0}},n[2][2];
3     n[0][0] = m[0][0]*c[0][0] + m[0][1]*c
4         [1][0];
5     n[0][1] = m[0][0]*c[0][1] + m[0][1]*c
6         [1][1];
7     n[1][0] = m[1][0]*c[0][0] + m[1][1]*c
8         [1][0];
9     n[1][1] = m[1][0]*c[0][1] + m[1][1]*c
10        [1][1];
11     if(x != 1)
12         exp(n,x-1);
13     else
14         cout << n[0][0];
15 }
16 int main(){
17     LL u[2][2] = {{1,1},{1,0}},n;
18     cin >> n;
19     cout << "gOHR C^2" << n+2 << "gμ-°";
20     exp(u,n);
21 }

```

6.9 MillerRobin

```

1 ULL Llmul(ULL a, ULL b, const ULL &mod) {
2     LL ans=0;

```

```

3     while(b) {
4         if(b&1) {
5             ans+=a;
6             if(ans>=mod) ans-=mod;
7         }
8         a<<=1, b>>=1;
9         if(a>=mod) a-=mod;
10    }
11    return ans;
12 }
13 ULL mod_mul(ULL a,ULL b,ULL m){
14     a%=m,b%=m; /* fast for m < 2^58 */
15     ULL y=(ULL)(((double)a*b/m+0.5);
16     ULL r=(a*b-y*m)%m;
17     return r<0?r+m:r;
18 }
19 template<typename T>
20 T pow(T a,T b,T mod){/*a^b%mod
21     T ans=1;
22     for(;b;a=mod_mul(a,a,mod),b>>=1)
23         if(b&1)ans=mod_mul(ans,a,mod);
24     return ans;
25 }
26 int sprp[3]={2,7,61}; //int範圍可解
27 int llsp[3]
28     [7]={2,325,9375,28178,450775,9780504,
29     1795265022}; //至少unsigned Long Long範圍
30 template<typename T>
31 bool isprime(T n,int *sprp,int num){
32     if(n==2)return 1;
33     if(n<2||n%2==0)return 0;
34     int t=0;
35     T u=n-1;
36     for(;u%2==0;++t)u>>=1;
37     for(int i=0;i<num;++i){
38         T a=sprp[i]%n;
39         if(a==0||a==1||a==n-1)continue;
40         T x=pow(a,u,n);
41         if(x==1||x==n-1)continue;
42         for(int j=0;j<t;++j){
43             x=mod_mul(x,x,n);
44             if(x==1)return 0;
45             if(x==n-1)break;
46         }
47         if(x==n-1)continue;
48         return 0;
49     }
50     return 1;
51 }

```

6.10 NTT

```

1 2615053605667*(2^18)+1,3
2 15*(2^27)+1,31
3 479*(2^21)+1,3
4 7*17*(2^23)+1,3
5 3*3*211*(2^19)+1,5
6 25*(2^22)+1,3
7 template<typename T,typename VT=vector<T>
8     >
9 struct NTT{
10     const T P,G;
11     NTT(T p=(1<<23)*7*17+1,T g=3):P(p),G(g)
12     {}
13     unsigned bit_reverse(unsigned a,int len
14     ){
15         //Look FFT.cpp
16     }
17     T pow_mod(T n,T k,T m){
18         T ans=1;
19         for(n=(n>=m?n%m:n);k;k>>=1){
20             if(k&1)ans=ans*n%m;
21             n=n*n%m;
22         }
23         return ans;
24     }
25     void ntt(bool is_inv,VT &in,VT &out,int
26         N){
27         int bitlen=__lg(N);
28         for(int i=0;i<N;++i)out[bit_reverse(i
29             ,bitlen)]=in[i];
30         for(int step=2,id=1;step<=N;step
31             <<=1,++id){
32             T wn=pow_mod(G,(P-1)>>id,P),wi=1,u,
33                 t;
34             const int mh=step>>1;
35             for(int i=0;i<mh;++i){
36                 for(int j=i;j<N;j+=step){
37                     u=out[j],t=wi*out[j+mh]%P;
38                     out[j]=u+t;
39                     out[j+mh]=u-t;
40                     if(out[j]>=P)out[j]-=P;

```

```

34         if(out[j+mh]<0)out[j+mh]+=P;
35     }
36     wi=wi*wn%P;
37 }
38 }
39 if(is_inv){
40     for(int i=1;i<N/2;++i)swap(out[i],
41         out[N-i]);
42     T invn=pow_mod(N,P-2,P);
43     for(int i=0;i<N;++i)out[i]=out[i]*
44         invn%P;
45 }
46 }
47 }
48 }
49 }

```

6.11 Simpson

```

1 double simpson(double a,double b){
2     double c=a+(b-a)/2;
3     return (F(a)+4*F(c)+F(b))*(b-a)/6;
4 }
5 double asr(double a,double b,double eps,
6     double A){
7     double c=a+(b-a)/2;
8     double L=simpson(a,c),R=simpson(c,b);
9     if( abs(L+R-A)<15*eps )
10         return L+R+(L+R-A)/15.0;
11     return asr(a,c,eps/2,L)+asr(c,b,eps/2,R
12         );
13 }
14 double asr(double a,double b,double eps){
15     return asr(a,b,eps,simpson(a,b));
16 }

```

6.12 SpeedExpo

```

1 LL expo(LL a,LL b,LL p){
2     if(b == 0) return 1;
3     if(b & 1) return (expo(a,b-1,p)*a)%p;
4     //b is odd
5     LL temp = expo(a,b/2,p);
6     return (temp*temp)%p;
7 }

```

6.13 外星模運算

```

1 //a[0]^(a[1]^a[2]^...)
2 #define maxn 100000
3 int euler[maxn+5];
4 bool is_prime[maxn+5];
5 void init_euler(){
6     is_prime[1]=1; //一不是質數
7     for(int i=1;i<=maxn;i++)euler[i]=i;
8     for(int i=2;i<=maxn;i++){
9         if(!is_prime[i]){ //是質數
10             euler[i]--;
11             for(int j=i<1;j<=maxn;j+=i){
12                 is_prime[j]=1;
13                 euler[j]=euler[j]/i*(i-1);
14             }
15         }
16     }
17 }
18 LL pow(LL a,LL b,LL mod){/*a^b%mod
19     LL ans=1;
20     for(;b;a=a*a%mod,b>>=1)
21         if(b&1)ans=ans*a%mod;
22     return ans;
23 }
24 bool isless(LL *a,int n,int k){
25     if(*a==1)return k>1;
26     if(--n==0)return *a<k;
27     int next=0;
28     for(LL b=1;b<k;++next)
29         b*=*a;
30     return isless(a+1,n,next);
31 }
32 LL high_pow(LL *a,int n,LL mod){
33     if(*a==1||--n==0)return *a%mod;
34     int k=0,r=euler[mod];
35     for(LL tma=1;tma!=pow(*a,k+r,mod);++k)
36         tma=tma*(a%mod);
37     if(isless(a+1,n,k))return pow(*a,
38         high_pow(a+1,n,k),mod);
39     int tmd=high_pow(a+1,n,r), t=(tmd-k+r)%
40         r;
41     return pow(*a,k+t,mod);
42 }

```



```

40 }
41 LL a[1000005];
42 int t, mod;
43 int main(){
44     init_euler();
45     scanf("%d",&t);
46     #define n 4
47     while(t--){
48         for(int i=0; i<n; ++i) scanf("%lld",&a[i]);
49         scanf("%d",&mod);
50         printf("%lld\n", high_pow(a, n, mod));
51     }
52     return 0;
53 }

```

6.14 大數取模

```

1 LL exp(LL x, LL y, LL p){
2     if(y == 0) return 1;
3     if(y & 1) return (exp(x, y-1, p)*x) % p;
4     //y is odd
5     else{
6         LL temp = exp(x, y/2, p);
7         return (temp*temp) % p;
8     }
9 }
10 LL calcmmod(LL index, LL p){
11     if(index == 0) return base[index] - '0';
12     LL single = calcmmod(index-1, p)*10;
13     return (single%p + base[index] - '0')%p;
14 }

```

6.15 數位統計

```

1 ll d[65], dp[65][2]; //up 區間是不是完整
2 ll dfs(int p, bool is8, bool up){
3     if(!p) return 1; // 回傳0是不是答案
4     if(!up && ~dp[p][is8]) return dp[p][is8];
5     int mx = up?d[p]:9; //可以用的有那些
6     ll ans=0;
7     for(int i=0; i<mx; ++i){
8         if(is8 && i==7) continue;
9         ans += dfs(p-1, i==8, up && i==mx);
10    }
11    if(!up) dp[p][is8] = ans;
12    return ans;
13 }
14 ll f(ll N){
15     int k=0;
16     while(N){ // 把數字先分解到陣列
17         d[++k] = N%10;
18         N/=10;
19     }
20     return dfs(k, false, true);
21 }

```

6.16 質因數分解

```

1 LL func(const LL n, const LL mod, const int c){
2     return (LLmul(n, n, mod) + c + mod) % mod;
3 }
4 LL pollorrho(const LL n, const int c){ // 循環節長度
5     LL a=1, b=1;
6     a=func(a, n, c)%n;
7     b=func(b, n, c)%n; b=func(b, n, c)%n;
8     while(gcd(abs(a-b), n)!=1){
9         a=func(a, n, c)%n;
10        b=func(b, n, c)%n; b=func(b, n, c)%n;
11    }
12    return gcd(abs(a-b), n);
13 }
14 void prefactor(LL &n, vector<LL> &v){
15     for(int i=0; i<12; ++i){
16         while(n%prime[i]==0){
17             v.push_back(prime[i]);
18             n/=prime[i];
19         }
20     }
21 }
22 }
23 }
24 }

```

```

25 void smallfactor(LL n, vector<LL> &v){
26     if(n<MAXPRIME){
27         while(isp[(int)n]){
28             v.push_back(isp[(int)n]);
29             n/=isp[(int)n];
30         }
31         v.push_back(n);
32     } else {
33         for(int i=0; i<primecnt && prime[i]*prime[i]<=n; ++i){
34             while(n%prime[i]==0){
35                 v.push_back(prime[i]);
36                 n/=prime[i];
37             }
38             if(n!=1) v.push_back(n);
39         }
40     }
41 }
42 void comfactor(const LL &n, vector<LL> &v){
43     if(n<1e9){
44         smallfactor(n, v);
45         return;
46     }
47     if(Isprime(n)){
48         v.push_back(n);
49         return;
50     }
51     LL d;
52     for(int c=3; ; ++c){
53         d = pollorrho(n, c);
54         if(d!=n) break;
55     }
56     comfactor(d, v);
57     comfactor(n/d, v);
58 }
59 void Factor(const LL &x, vector<LL> &v){
60     LL n = x;
61     if(n==1){ puts("Factor 1"); return; }
62     prefactor(n, v);
63     if(n==1) return;
64     comfactor(n, v);
65     sort(v.begin(), v.end());
66 }
67 void AllFactor(const LL &n, vector<LL> &v){
68     vector<LL> tmp;
69     Factor(n, tmp);
70     v.clear();
71     v.push_back(1);
72     int len;
73     LL now=1;
74     for(int i=0; i<tmp.size(); ++i){
75         if(i==0 || tmp[i]!=tmp[i-1]){
76             len = v.size();
77             now = 1;
78         }
79         now*=tmp[i];
80         for(int j=0; j<len; ++j){
81             v.push_back(v[j]*now);
82         }
83     }
84 }
85 }
86 }

```

7 String

7.1 AC 自動機

```

1 template<char L='a', char R='z'>
2 class ac_automaton{
3     struct joe{
4         int next[R-L+1], fail, efl, ed, cnt_dp, vis;
5         joe():ed(0), cnt_dp(0), vis(0){
6             for(int i=0; i<R-L; ++i) next[i]=0;
7         }
8     };
9     public:
10        std::vector<joe> S;
11        std::vector<int> q;
12        int qs, qe, vt;
13        ac_automaton():S(1), qs(0), qe(0), vt(0){
14            void clear(){
15                q.clear();
16                S.resize(1);
17                for(int i=0; i<R-L; ++i) S[0].next[i]=0;
18                S[0].cnt_dp=S[0].vis=qs=qe=vt=0;
19            }
20            void insert(const char *s){

```

```

21     int o=0;
22     for(int i=0, id; s[i]; ++i){
23         id=s[i]-L;
24         if(!S[o].next[id]){
25             S.push_back(joe());
26             S[o].next[id]=S.size()-1;
27         }
28         o=S[o].next[id];
29     }
30     ++S[o].ed;
31 }
32 void build_fail(){
33     S[0].fail=S[0].efl=-1;
34     q.clear();
35     q.push_back(0);
36     ++qe;
37     while(qs!=qe){
38         int pa=q[qs++], id, t;
39         for(int i=0; i<R-L; ++i){
40             t=S[pa].next[i];
41             if(!t) continue;
42             id=S[pa].fail;
43             while(~id && !S[id].next[i]) id=S[id].fail;
44             S[t].fail=~id?S[id].next[i]:0;
45             S[t].efl=S[S[t].fail].ed+S[t].fail?S[t].fail:S[S[t].fail].efl;
46             q.push_back(t);
47             ++qe;
48         }
49     }
50 }
51 /*DP出每個前綴在字串s出現的次數並傳回所有字串被s匹配成功的次數O(N*M)*/
52 int match_0(const char *s){
53     int ans=0, id, p=0, i;
54     for(i=0; s[i]; ++i){
55         id=s[i]-L;
56         while(!S[p].next[id] && p=S[p].fail);
57         if(!S[p].next[id]) continue;
58         p=S[p].next[id];
59         ++S[p].cnt_dp; /*匹配成功則它所有後綴都可以被匹配(DP計算)*/
60     }
61     for(i=qe-1; i>=0; --i){
62         ans+=S[q[i]].cnt_dp*S[q[i]].ed;
63         if(~S[q[i]].fail) S[S[q[i]].fail].cnt_dp+=S[q[i]].cnt_dp;
64     }
65     return ans;
66 }
67 /*多串匹配走efl邊並傳回所有字串被s匹配成功的次數O(N*M^1.5)*/
68 int match_1(const char *s) const{
69     int ans=0, id, p=0, t;
70     for(int i=0; s[i]; ++i){
71         id=s[i]-L;
72         while(!S[p].next[id] && p=S[p].fail);
73         if(!S[p].next[id]) continue;
74         p=S[p].next[id];
75         if(S[p].ed) ans+=S[p].ed;
76         for(t=S[p].efl; ~t; t=S[t].efl){
77             ans+=S[t].ed; /*因為都走efl邊所以保證匹配成功*/
78         }
79     }
80     return ans;
81 }
82 /*枚舉(s的子字串na)的所有相異字串各恰一次並傳回次數O(N*M^(1/3))*/
83 int match_2(const char *s){
84     int ans=0, id, p=0, t;
85     ++vt;
86     /*把黏計vt+=1 只要vt沒溢位 所有S[p].vis==vt就會變成false 這種利用vt的方法可以O(1)歸零vis陣列*/
87     for(int i=0; s[i]; ++i){
88         id=s[i]-L;
89         while(!S[p].next[id] && p=S[p].fail);
90         if(!S[p].next[id]) continue;
91         p=S[p].next[id];
92         if(S[p].ed && S[p].vis!=vt){
93             S[p].vis=vt;
94             ans+=S[p].ed;
95         }
96         for(t=S[p].efl; ~t && S[t].vis!=vt; t=S[t].efl){
97             S[t].vis=vt;
98             ans+=S[t].ed; /*因為都走efl邊所以保證匹配成功*/
99         }
100    }

```

```

101     }
102     return ans;
103 }
104 /*把AC自動機變成真的自動機*/
105 void evolution(){
106     for(qs=1;qs!=qe;){
107         int p=q[qs++];
108         for(int i=0;i<R-L;++i)
109             if(S[p].next[i]==0)S[p].next[i]=S
110                 [S[p].fail].next[i];
111     }
112 };

```

```

5     ++k;
6     if(t){
7         if(t>0)i+=k;
8         else j+=k;
9         if(i==j)++j;
10        k=0;
11    }
12 }
13 return min(i,j); //最小循環表示法起始位
14     置

```

```

1 void z_alg(char *s,int len,int *z){
2     int l=0,r=0;
3     z[0]=len;
4     for(int i=1;i<len;++i){
5         z[i]=i>r?0:(i-l+z[i-l]<z[l]?z[i-l]:r-
6             i+1);
7         while(i+z[i]<len&&s[i+z[i]]==s[z[i]])
8             ++z[i];
9         if(i+z[i]-1>r)r=i+z[i]-1,l=i;
10    }

```

7.2 hash

```

1 #define MAXN 1000000
2 #define mod 1073676287
3 /*mod 必須要是質數*/
4 typedef long long T;
5 char s[MAXN+5];
6 T h[MAXN+5]; /*hash陣列*/
7 T h_base[MAXN+5]; /*h_base[n]=(prime^n)%
8     mod*/
9 void hash_init(int len,T prime){
10     h_base[0]=1;
11     for(int i=1;i<len;++i){
12         h[i]=(h[i-1]*prime+s[i-1])%mod;
13         h_base[i]=(h_base[i-1]*prime)%mod;
14     }
15 T get_hash(int l,int r){ /*閉區間寫法 · 設
16     編號為0 ~ len-1*/
17     return (h[r+1]-(h[l]*h_base[r-l+1])%mod
18         +mod)%mod;

```

7.3 KMP

```

1 /*產生fail function*/
2 void kmp_fail(char *s,int len,int *fail){
3     int id=-1;
4     fail[0]=-1;
5     for(int i=1;i<len;++i){
6         while(~id&&s[id+1]!=s[i])id=fail[id];
7         if(s[id+1]==s[i])++id;
8         fail[i]=id;
9     }
10 }
11 /*以字串B匹配字串A · 傳回匹配成功的數量(用
12     B的fail)*/
13 int kmp_match(char *A,int lenA,char *B,
14     int lenB,int *fail){
15     int id=-1,ans=0;
16     for(int i=0;i<lenA;++i){
17         while(~id&&B[id+1]!=A[i])id=fail[id];
18         if(B[id+1]==A[i])++id;
19         if(id==lenB-1){ /*匹配成功*/
20             ++ans, id=fail[id];
21         }
22     }
23     return ans;

```

7.4 manacher

```

1 //原字串: asdsasdsa
2 //先把字串變成這樣: @#a#s#d#s#a#s#d#s#a#
3 void manacher(char *s,int len,int *z){
4     int l=0,r=0;
5     for(int i=1;i<len;++i){
6         z[i]=r>i?min(z[2*l-i],r-i):1;
7         while(s[i+z[i]]==s[i-z[i]])++z[i];
8         if(z[i]+i>r)r=z[i]+i,l=i;
9     } //ans = max(z)-1
10 }

```

7.5 minimal string rotation

```

1 int min_string_rotation(const string &s){
2     int n=s.size(),i=0,j=1,k=0;
3     while(i<n&&j<n&&k<n){
4         int t=s[(i+k)%n]-s[(j+k)%n];

```

7.6 reverseBWT

```

1 const int MAXN = 305, MAXC = 'Z';
2 int ranks[MAXN], tots[MAXC], first[MAXC];
3 void rankBWT(const string &bw){
4     memset(ranks,0,sizeof(int)*bw.size());
5     memset(tots,0,sizeof(tots));
6     for(size_t i=0;i<bw.size();++i)
7         ranks[i] = tots[ bw[i] ]++;
8 }
9 void firstCol(){
10     memset(first,0,sizeof(first));
11     int totc = 0;
12     for(int c='A';c<='Z';++c){
13         if(!tots[c]) continue;
14         first[c] = totc;
15         totc += tots[c];
16     }
17 }
18 string reverseBwt(string bw,int begin){
19     rankBWT(bw), firstCol();
20     int i = begin; //原字串最後一個元素的位
21     置
22     string res;
23     do{
24         char c = bw[i];
25         res = c + res;
26         i = first[ bw[i] ] + ranks[i];
27     }while( i != begin );
28     return res;

```

7.7 suffix array lcp

```

1 #define radix_sort(x,y){\
2     for(i=0;i<A;++i)c[i]=0;\
3     for(i=0;i<n;++i)c[x[y[i]]]++;\
4     for(i=1;i<A;++i)c[i]+=c[i-1];\
5     for(i=n-1;~i;--i)sa[--c[x[y[i]]]]=y[i
6     ];\
7 }
8 #define AC(r,a,b)\
9     r[a]=r[b]|a>k>n||r[a+k]!=r[b+k]
10 void suffix_array(const char *s,int n,int
11     *sa,int *rank,int *tmp,int *c){
12     int A='z'+1,i,k,id=0;
13     for(i=0;i<n;++i)rank[tmp[i]=i]=s[i];
14     radix_sort(rank,tmp);
15     for(k=1;id<n-1;k<=1){
16         for(id=0,i=n-k;i<n;++i)tmp[id++]=i;
17         for(i=0;i<n;++i)
18             if(sa[i]>=k)tmp[id++]=sa[i]-k;
19         radix_sort(rank,tmp);
20         swap(rank,tmp);
21         for(rank[sa[0]]=id=0,i=1;i<n;++i)
22             rank[sa[i]]=id+=AC(tmp,sa[i-1],sa[i
23             ]);
24         A=id+1;
25     }
26 }
27 //h: 高度數組 sa: 後綴數組 rank: 排名
28 void suffix_array_lcp(const char *s,int
29     len,int *h,int *sa,int *rank){
30     for(int i=0;i<len;++i)rank[sa[i]]=i;
31     for(int i=0,k=0;i<len;++i){
32         if(rank[i]==0)continue;
33         if(k--<0)
34             while(s[i+k]==s[sa[rank[i]-1]+k])++k;
35         h[rank[i]]=k;
36     }
37     h[0]=0; // h[k]=Lcp(sa[k],sa[k-1]);
38 }

```

7.8 Z

8 Tarjan

8.1 tnfsb017 2 sat

```

1 #include<bits/stdc++.h>
2 using namespace std;
3 #define MAXN 8001
4 #define MAXN2 MAXN*4
5 #define n(X) ((X)+2*N)
6 vector<int> v[MAXN2], rv[MAXN2], vis_t;
7 int N,M;
8 void addedge(int s,int e){
9     v[s].push_back(e);
10    rv[e].push_back(s);
11 }
12 int scc[MAXN2];
13 bool vis[MAXN2]={false};
14 void dfs(vector<int> *uv,int n,int k=-1){
15     vis[n]=true;
16     for(int i=0;i<uv[n].size();++i)
17         if(!vis[uv[n][i]])
18             dfs(uv,uv[n][i],k);
19     if(uv==v)vis_t.push_back(n);
20     scc[n]=k;
21 }
22 void solve(){
23     for(int i=1;i<=N;++i){
24         if(!vis[i])dfs(v,i);
25         if(!vis[n(i)])dfs(v,n(i));
26     }
27     memset(vis,0,sizeof(vis));
28     int c=0;
29     for(int i=vis_t.size()-1;i>=0;--i)
30         if(!vis[vis_t[i]])
31             dfs(rv,vis_t[i],c++);
32 }
33 int main(){
34     int a,b;
35     scanf("%d%d",&N,&M);
36     for(int i=1;i<=N;++i){
37         // (A or B)&(!A & !B) A^B
38         a=i*2-1;
39         b=i*2;
40         addedge(n(a),b);
41         addedge(n(b),a);
42         addedge(a,n(b));
43         addedge(b,n(a));
44     }
45     while(M--){
46         scanf("%d%d",&a,&b);
47         a = a>0?a*2-1:-a*2;
48         b = b>0?b*2-1:-b*2;
49         // A or B
50         addedge(n(a),b);
51         addedge(n(b),a);
52     }
53     solve();
54     bool check=true;
55     for(int i=1;i<=2*N;++i)
56         if(scc[i]==scc[n(i)])
57             check=false;
58     if(check){
59         printf("%d\n",N);
60         for(int i=1;i<=2*N;i+=2){
61             if(scc[i]>scc[i+2*N]) putchar('+');
62             else putchar('-');
63         }
64         puts("");
65     }else puts("0");
66     return 0;
67 }

```

8.2 雙連通分量 & 割點

```

1 #define N 1005
2 vector<int> G[N]; // 1-base
3 vector<int> bcc[N]; // 存每塊雙連通分量的點
4 int low[N], vis[N], Time;
5 int bcc_id[N], bcc_cnt; // 1-base
6 bool is_cut[N]; // 是否為割點
7 int st[N], top;
8 void dfs(int u, int pa=-1) { // u當前點 · pa父
    親
9     int t, child=0;
10    low[u]=vis[u]=++Time;
11    st[top++]=u;
12    for(int v:G[u]){
13        if(!vis[v]){
14            dfs(v,u), ++child;
15            low[u]=min(low[u], low[v]);
16            if(vis[u]<=low[v]){
17                is_cut[u]=1;
18                bcc[++bcc_cnt].clear();
19                do{
20                    bcc_id[t=st[--top]]=bcc_cnt;
21                    bcc[bcc_cnt].push_back(t);
22                }while(t!=v);
23                bcc_id[u]=bcc_cnt;
24                bcc[bcc_cnt].push_back(u);
25            }
26        } else if(vis[v]<vis[u]&&v!=pa) // 反向
            邊
27            low[u] = min(low[u], vis[v]);
28        // u是dfs樹的根要特判
29        if(pa==--1&&child<2) is_cut[u]=0;
30    }
31    void bcc_init(int n){
32        Time=bcc_cnt=top=0;
33        for(int i=1; i<=n; ++i){
34            G[i].clear();
35            is_cut[i]=vis[i]=bcc_id[i]=0;
36        }
37    }

```

9 Tree Problem

9.1 HeavyLight

```

1 #include<vector>
2 #define MAXN 100005
3 int siz[MAXN], max_son[MAXN], pa[MAXN], dep[
    MAXN];
4 int link_top[MAXN], link[MAXN], cnt;
5 vector<int> G[MAXN];
6 void find_max_son(int u){
7     siz[u]=1;
8     max_son[u]=-1;
9     for(auto v:G[u]){
10         if(v==pa[u]) continue;
11         pa[v]=u;
12         dep[v]=dep[u]+1;
13         find_max_son(v);
14         if(max_son[u]==-1 || siz[v]>siz[max_son
            [u]]) max_son[u]=v;
15         siz[u]+=siz[v];
16     }
17 }
18 void build_link(int u, int top){
19     link[u]=++cnt;
20     link_top[u]=top;
21     if(max_son[u]==-1) return;
22     build_link(max_son[u], top);
23     for(auto v:G[u]){
24         if(v==max_son[u] || v==pa[u]) continue;
25         build_link(v, v);
26     }
27 }
28 int find_lca(int a, int b){
29     // 求LCA · 可以在過程中對區間進行處理
30     int ta=link_top[a], tb=link_top[b];
31     while(ta!=tb){
32         if(dep[ta]<dep[tb]){
33             swap(ta, tb);
34             swap(a, b);
35         }
36         // 這裡可以對a所在的鏈做區間處理
37         // 區間為(Link[ta], Link[a])
38         ta=link_top[a=pa[ta]];
39     }
40     // 最後a,b會在同一條鏈 · 若a!=b還要在進行
    一次區間處理

```

```

41     return dep[a]<dep[b]?a:b;
42 }

```

9.2 kruskal(MST)

```

1 #include<bits/stdc++.h>
2 #define good ios_base::sync_with_stdio(0)
    ; cin.tie(0)
3 #define RSIZE 10002
4 #define pll pair<LL, LL>
5 typedef long long LL;
6 using namespace std;
7
8 struct EDGE{
9     LL u, v, w;
10 };
11
12 vector<EDGE> adjacent; // out neighbor,
    weight
13 LL fa[RSIZE];
14
15 bool cmp(EDGE &a, EDGE &b) { // sort by
    weight
16     return a.w < b.w;
17 }
18 LL sfind(LL now) { // find root, root's
    father=set size
19     if(fa[now] < 0)
20         return now;
21     return fa[now] = sfind(fa[now]);
22 }
23 bool merge(LL u, LL v) { // find two root,
    comparing size(by root's father)
24     LL set1 = sfind(u), set2 = sfind(v);
25     if(set1 == set2) return false; // same
    root-> no need to merge
26     if(fa[set1] < fa[set2]) { // set1 is
        larger
27         fa[set1] += fa[set2];
28         fa[set2] = set1;
29     }
30     else{
31         fa[set2] += fa[set1];
32         fa[set1] = set2;
33     }
34     return true;
35 }
36
37 int main(){
38     good;
39     // freopen("file name", "r", stdin);
    // input redirection
40     LL i, n, m;
41     cin >> n >> m;
42     for(i = 0; i < m; i++){
43         LL x, y, weight;
44         cin >> x >> y >> weight;
45         adjacent.push_back({x, y, weight});
46     }
47     memset(fa, -1, sizeof(fa)); // unvisited
48     sort(adjacent.begin(), adjacent.end(),
        cmp); // sort by weight
49     LL cost = 0, now_edge = 0;
50     for(EDGE e : adjacent){
51         if(merge(e.u, e.v)) { // connect edge
52             cost += e.w;
53             now_edge++;
54         }
55     }
56     if(now_edge < n-1) // not a MST
57         cout << -1 << endl;
58     else
59         cout << cost << endl;
60     return 0;
61 }

```

9.3 LCA

```

1 const int MAXN=100000; // 1-base
2 const int MLG=17; // log2(MAXN)+1;
3 int pa[MLG+2][MAXN+5];
4 int dep[MAXN+5];
5 vector<int> G[MAXN+5];
6 void dfs(int x, int p=0) { // dfs(root);
7     pa[0][x]=p;
8     for(int i=0; i<=MLG; ++i)
9         pa[i+1][x]=pa[i][pa[i][x]];
10    for(auto &i:G[x]){
11        if(i==p) continue;
12        dep[i]=dep[x]+1;
13        dfs(i, x);

```

```

14    }
15 }
16 inline int jump(int x, int d){
17     for(int i=0; i<=MLG; ++i)
18         if((d>>i)&1) x=pa[i][x];
19     return x;
20 }
21 inline int find_lca(int a, int b){
22     if(dep[a]>dep[b]) swap(a, b);
23     b=jump(b, dep[b]-dep[a]);
24     if(a==b) return a;
25     for(int i=MLG; i>=0; --i){
26         if(pa[i][a]!=pa[i][b]){
27             a=pa[i][a];
28             b=pa[i][b];
29         }
30     }
31     return pa[0][a];
32 }

```

9.4 link cut tree

```

1 struct splay_tree{
2     int ch[2], pa; // 子節點跟父母
3     bool rev; // 反轉的懶惰標記
4     splay_tree(): pa(0), rev(0) { ch[0]=ch
        [1]=0; }
5 };
6 vector<splay_tree> nd;
7 // 有的時候用vector會TLE · 要注意
8 // 這邊以node[0]作為null節點
9 bool isroot(int x) { // 判斷是否為這棵splay
    tree的根
10     return nd[nd[x].pa].ch[0]!=x&&nd[nd[x].
        pa].ch[1]!=x;
11 }
12 void down(int x) { // 懶惰標記下推
13     if(nd[x].rev){
14         if(nd[x].ch[0]) nd[nd[x].ch[0]].rev
            ^=1;
15         if(nd[x].ch[1]) nd[nd[x].ch[1]].rev
            ^=1;
16         swap(nd[x].ch[0], nd[x].ch[1]);
17         nd[x].rev=0;
18     }
19 }
20 void push_down(int x) { // 所有祖先懶惰標記
    下推
21     if(!isroot(x)) push_down(nd[x].pa);
22     down(x);
23 }
24 void up(int x) { // 將子節點的資訊向上更新
25 void rotate(int x) { // 旋轉 · 會自行判斷轉的
    方向
26     int y=nd[x].pa, z=nd[y].pa, d=(nd[y].ch
        [1]==x);
27     nd[x].pa=z;
28     if(!isroot(y)) nd[z].ch[nd[z].ch[1]==y]=
        x;
29     nd[y].ch[d]=nd[x].ch[d^1];
30     nd[nd[y].ch[d]].pa=y;
31     nd[y].pa=x, nd[x].ch[d^1]=y;
32     up(y), up(x);
33 }
34 void splay(int x) { // 將x伸展到splay tree的
    根
35     push_down(x);
36     while(!isroot(x)){
37         int y=nd[x].pa;
38         if(!isroot(y)){
39             int z=nd[y].pa;
40             if((nd[z].ch[0]==y)^(nd[y].ch[0]==x))
                rotate(y);
41             else rotate(x);
42         }
43         rotate(x);
44     }
45 }
46 int access(int x){
47     int last=0;
48     while(x){
49         splay(x);
50         nd[x].ch[1]=last;
51         up(x);
52         last=x;
53         x=nd[x].pa;
54     }
55     return last; // access後splay tree的根
56 }
57 void access(int x, bool is=0) { // is=0就是一
    般的access

```

```

58 int last=0;
59 while(x){
60     splay(x);
61     if(is&&!nd[x].pa){
62         //printf("%d\n",max(nd[last].ma,nd[
63             nd[x].ch[1]].ma));
64     }
65     nd[x].ch[1]=last;
66     up(x);
67     last=x;
68     x=nd[x].pa;
69 }
70 void query_edge(int u,int v){
71     access(u);
72     access(v,1);
73 }
74 void make_root(int x){
75     access(x),splay(x);
76     nd[x].rev^=1;
77 }
78 void make_root(int x){
79     nd[access(x)].rev^=1;
80     splay(x);
81 }
82 void cut(int x,int y){
83     make_root(x);
84     access(y);
85     splay(y);
86     nd[y].ch[0]=0;
87     nd[x].pa=0;
88 }
89 void cut_parents(int x){
90     access(x);
91     splay(x);
92     nd[nd[x].ch[0]].pa=0;
93     nd[x].ch[0]=0;
94 }
95 void link(int x,int y){
96     make_root(x);
97     nd[x].pa=y;
98 }
99 int find_root(int x){
100     x=access(x);
101     while(nd[x].ch[0]x=nd[x].ch[0];
102     splay(x);
103     return x;
104 }
105 int query(int u,int v){
106     //傳回uv路徑splay tree的根結點
107     //這種寫法無法求LCA
108     make_root(u);
109     return access(v);
110 }
111 int query_lca(int u,int v){
112     //假設求鏈上點權的總和，sum是子樹的權重
113     //和，data是節點的權重
114     access(u);
115     int lca=access(v);
116     splay(u);
117     if(u==lca){
118         //return nd[lca].data+nd[nd[lca].ch
119             [1]].sum
120     }else{
121         //return nd[lca].data+nd[nd[lca].ch
122             [1]].sum+nd[u].sum
123     }
124 }
125 struct EDGE{
126     int a,b,w;
127 }e[10005];
128 int n;
129 vector<pair<int,int>> G[10005];
130 //first表示子節點，second表示邊的編號
131 int pa[10005],edge_node[10005];
132 //pa是父母節點，暫存用的，edge_node是每個
133 //編被存在哪個點裡面的陣列
134 void bfs(int root){
135     //在建構的時候把每個點都設成一個splay
136     tree
137     queue<int> q;
138     for(int i=1;i<=n;++i)pa[i]=0;
139     q.push(root);
140     while(q.size()){
141         int u=q.front();
142         q.pop();
143         for(auto P:G[u]){
144             int v=P.first;
145             if(v!=pa[u]){
146                 pa[v]=u;
147                 nd[v].pa=u;
148                 nd[v].data=e[P.second].w;
149                 edge_node[P.second]=v;
150                 up(v);
151                 q.push(v);

```

9.5 POJ tree

```

1 #include<bits/stdc++.h>
2 using namespace std;
3 #define MAXN 10005
4 int n,k;
5 vector<pair<int,int>> g[MAXN];
6 int size[MAXN];
7 bool vis[MAXN];
8 inline void init(){
9     for(int i=0;i<=n;++i){
10         g[i].clear();
11         vis[i]=0;
12     }
13 }
14 void get_dis(vector<int> &dis,int u,int
15     pa,int d){
16     dis.push_back(d);
17     for(size_t i=0;i<g[u].size();++i){
18         int v=g[u][i].first,w=g[u][i].second;
19         if(v!=pa&&!vis[v])get_dis(dis,v,u,d+w
20             );
21     }
22 }
23 vector<int> dis;//這東西如果放在函數裡會
24 TLE
25 int cal(int u,int d){
26     dis.clear();
27     get_dis(dis,u,-1,d);
28     sort(dis.begin(),dis.end());
29     int l=0,r=dis.size()-1,res=0;
30     while(l<r){
31         while(l<r&&dis[l]+dis[r]>k)--r;
32         res+=r-(l++);
33     }
34     return res;
35 }
36 pair<int,int> tree_centroid(int u,int pa,
37     const int sz){
38     size[u]=1;//找樹重心，second是重心
39     pair<int,int> res(INT_MAX,-1);
40     int ma=0;
41     for(size_t i=0;i<g[u].size();++i){
42         int v=g[u][i].first;
43         if(v==pa||vis[v])continue;
44         res=min(res,tree_centroid(v,u,sz));
45         size[u]+=size[v];
46         ma=max(ma,size[v]);
47     }
48     ma=max(ma,sz-size[u]);
49     return min(res,make_pair(ma,u));
50 }
51 int tree_DC(int u,int sz){
52     int center=tree_centroid(u,-1,sz).
53         second;
54     int ans=cal(center,0);
55     vis[center]=1;
56     for(size_t i=0;i<g[center].size();++i){
57         int v=g[center][i].first,w=g[center][
58             i].second;
59         if(vis[v])continue;
60         ans-=cal(v,w);
61         ans+=tree_DC(v,size[v]);
62     }
63     return ans;
64 }
65 int main(){
66     while(scanf("%d",&n,&k),n||k){
67         init();
68         for(int i=1;i<=n;++i){
69             int u,v,w;
70             scanf("%d%d%d",&u,&v,&w);
71             g[u].push_back(make_pair(v,w));
72             g[v].push_back(make_pair(u,w));
73         }
74         printf("%d\n",tree_DC(1,n));
75     }
76     return 0;
77 }

```

9.6 Prim(MST)

```

1 #include<bits/stdc++.h>
2 #define good ios_base::sync_with_stdio(0)
3 ;cin.tie(0)
4 #define RSIZE 502
5 #define oo 1000000001 //1e9+1
6 typedef long long LL;
7 using namespace std;
8 vector<pair<LL,LL>> adjacent[RSIZE];//out
9 neighbor, weight of edge
10 LL dis[RSIZE],fa[RSIZE];//dis for weight
11 of two vertexes
12 bool visit[RSIZE] = {false};
13
14 int main(){
15     good;
16     //freopen("file name", "r", stdin);
17     //input redirection
18     LL i,n,m;
19     cin >> n >> m;
20     for(i = 0; i < m; i++){
21         LL x,y,w;
22         cin >> x >> y >> w;
23         adjacent[x].push_back({y,w});
24         adjacent[y].push_back({x,w});
25     }
26     //initial
27     LL start = 0;
28     memset(dis,oo,sizeof(dis));
29     memset(fa,-1,sizeof(fa));
30     priority_queue<pair<LL,LL>> PQ;//-dis
31     [,vertex
32     PQ.push({dis[start] = 0,start});
33     //prim
34     while (!PQ.empty()){
35         auto pt = PQ.top();
36         PQ.pop();
37         LL v = pt.second;
38         if(visit[v]) continue;
39         visit[v] = true;
40         for(auto it : adjacent[v]){
41             LL neighbor = it.first,w = it.
42                 second;
43             if(visit[neighbor]) continue;
44             if(w < dis[neighbor]){ //new
45                 edge is shorter
46                 dis[neighbor] = w;
47                 fa[neighbor] = v;
48                 PQ.push({-dis[neighbor],
49                     neighbor});
50             }
51         }
52     }
53     LL cost = 0,cnt = 0;
54     //count cost and check if MST exists
55     for(i = 0; i < n; i++){
56         if(dis[i] < oo)
57             cost += dis[i];
58         else
59             cnt++;
60     }
61     if(cnt)
62         cout << -1 << endl;
63     else
64         cout << cost << endl;
65     return 0;
66 }

```

9.7 segment tree

```

1 #include<bits/stdc++.h>
2 #define good ios_base::sync_with_stdio(0)
3 ;cin.tie(0)
4 #define RSIZE 100000
5 #define pll pair<LL,LL>
6 #define lc 2*index //c means child
7 #define rc 2*index+1
8 typedef long long LL;
9 using namespace std;
10 LL tree[4*RSIZE];//saving range maximum
11 LL lazy[4*RSIZE] = {0};
12 LL num[RSIZE],cnt = 1;
13
14 //using range maximum as example
15 void build(LL L,LL R,LL index){
16     LL temp = index;
17     if(L == R){
18         tree[index] = num[cnt];
19         cnt++;
20         return;

```



```

21     }
22     LL M = (L+R)/2;
23     build(L,M,lc);
24     build(M+1,R,rc);
25     tree[index] = max(tree[lc],tree[rc]);
26 }
27 //single point modify
28 void modify(LL x,LL v,LL L,LL R,LL index)
29 {
30     if(L == R){
31         tree[index] = v;
32         return;
33     }
34     LL M = (L+R)/2;
35     if(x <= M)//left side
36         modify(x,v,L,M,lc);
37     else
38         modify(x,v,M+1,R,rc);
39     tree[index] = max(tree[lc],tree[rc]);
40 }
41 //a range including index has to add tag
42 void addtag(LL tag,LL index){
43     tree[index] += tag;
44     lazy[index] += tag;
45 }
46 //transferring tag to child
47 void push(LL index){
48     addtag(lazy[index],lc);
49     addtag(lazy[index],rc);
50     lazy[index] = 0;//tag is transfered to child
51 }
52 //Lower variables are queried range,UPPER ones are full range
53 LL query(LL l,LL r,LL L,LL R,LL index){
54     if(l <= L && R <= r) return tree[index];
55     push(index);//if use single point modify,no need
56     LL M = (L+R)/2;
57     if(r <= M) //answer in the left side, don't need to query right side
58         return query(l,r,L,M,lc);
59     else if(l > M) //in right side
60         return query(l,r,M+1,R,rc);
61     else //answer cross both side
62         return max(query(l,r,L,M,lc), query(l,r,M+1,R,rc));
63 }
64 void multi_modify(LL l,LL r,LL v,LL L,LL R,LL index){
65     if(l <= L && R <= r){
66         addtag(v,index);
67         return;
68     }
69     push(index);
70     LL M = (L+R)/2;
71     if(r <= M) multi_modify(l,r,v,L,M,lc);
72     else if(l > M) multi_modify(l,r,v,M+1,R,rc);
73     else{
74         multi_modify(l,r,v,L,M,lc);
75         multi_modify(l,r,v,M+1,R,rc);
76     }
77     tree[index] = max(tree[lc],tree[rc]);
78 }
79 int main(){
80     good;
81     LL k,n;
82     //build(1,n,1)at first,can use query(L,r,1,n,1).
83     return 0;
84 }

```

10 default

10.1 8 queen

```

1 LL nqueen(LL n){
2     int p[17],total = 0;
3     for(int i = 0; i < n; i++){
4         p[i] = i;
5         do{
6             bool valid = true;
7             for(int i = 0; i < n; i++){
8                 for(int j = i+1; j < n; j++){
9                     if(abs(p[i]-p[j]) == j-i)
10                        //same diagonal
11                        valid = false;

```

```

11         break;
12     }
13     }
14     }
15     if(valid) total++;
16     while (next_permutation(p,p+n));
17     return total;
18 }

```

10.2 debug

```

1 #ifdef DEBUG
2 #define dbg(...) {\
3     fprintf(stderr,"%s - %d : (%s) = ",
4         __PRETTY_FUNCTION__,__LINE__,#
5         __VA_ARGS__);\
6     _DO(__VA_ARGS__);\
7 }
8 template<typename I> void _DO(I&x){cerr
9     <<x<<endl;}
10 template<typename I,typename...T> void
11     _DO(I&x,T&...tail){cerr<<x<<" ";
12     _DO(tail...);}
13 #else
14 #define dbg(...)
15 #endif

```

10.3 IncStack

```

1 //Magic
2 #pragma GCC optimize "Ofast"
3 //stack resize,change esp to rsp if 64-bit system
4 asm("mov %0,%esp\n" :: "g"(mem+10000000))
5 ;
6 -Wl,--stack,214748364 -trigraphs
7 #pragma comment(linker, "/STACK
8 :1024000000,1024000000")
9 //Linux stack resize
10 #include<sys/resource.h>
11 void increase_stack(){
12     const rlim_t ks=64*1024*1024;
13     struct rlimit rl;
14     int res=getrlimit(RLIMIT_STACK,&rl);
15     if(!res&&rl.rlim_cur<ks){
16         rl.rlim_cur=ks;
17         res=setrlimit(RLIMIT_STACK,&rl);
18     }
19 }

```

10.4 input

```

1 inline int read(){
2     int x=0; bool f=0; char c=getchar();
3     while(ch<'0' || '9'<ch)f|=ch=='-',ch=
4         getchar();
5     while('0'<=ch&&ch<='9')x=x*10-'0'+ch,ch=
6         getchar();
7     return f?-x:x;
8 }
9 // #!/bin/bash
10 // g++ -std=c++11 -O2 -Wall -Wextra -Wno-
11 // unused-result -DDEBUG $1 && ./a.out
12 // -fsanitize=address -fsanitize=
13 // undefined -fsanitize=return

```

10.5 randomize

```

1 map<LL,LL> discret;
2 for(i = 0; i < n; i++){
3     cin >> a[i];
4     discret[a[i]] = 0;
5 }
6 LL index = 0;
7 for(auto &it : discret)
8     it.second = index++;

```

10.6 模板

```

1 #include<bits/stdc++.h>
2 #define good ios_base::sync_with_stdio(0)
3 ;cin.tie(0)
4 #define RSIZE 101
5 #define pll pair<LL,LL>
6 #define lc 2*index
7 #define rc 2*index+1
8 typedef long long LL;
9 using namespace std;
10
11 int main(){
12     good;
13     //freopen("file name", "r", stdin);
14     //input redirection
15     return 0;
16 }

```

11 other

11.1 WhatDay

```

1 int whatday(int y,int m,int d){
2     if(m<=2)m+=12,-y;
3     if(y<1752||y==1752&&m<9||y==1752&&m
4         ==9&&d<3)
5         return (d+2*m+3*(m+1)/5+y+y/4+5)%7;
6     return (d+2*m+3*(m+1)/5+y+y/4-y/100+y
7         /400)%7;
8 }

```

11.2 上下最大正方形

```

1 void solve(int n,int a[],int b[]){// 1-
2     base
3     int ans=0;
4     deque<int>da,db;
5     for(int l=1,r=1;r<=n;++r){
6         while(da.size()&&a[da.back()]>a[r]){
7             da.pop_back();
8         }
9         da.push_back(r);
10        while(db.size()&&b[db.back()]>b[r]){
11            db.pop_back();
12        }
13        db.push_back(r);
14        for(int d=a[da.front()]+b[db.front()]
15            ;r-l+1>d;++l){
16            if(da.front()==l)da.pop_front();
17            if(db.front()==l)db.pop_front();
18            if(da.size()&&db.size()){
19                d=a[da.front()]+b[db.front()];
20            }
21        }
22        ans=max(ans,r-l+1);
23    }
24    printf("%d\n",ans);
25 }

```

11.3 最大矩形

```

1 LL max_rectangle(vector<int> s){
2     stack<pair<int,int> > st;
3     st.push(make_pair(-1,0));
4     s.push_back(0);
5     LL ans=0;
6     for(size_t i=0;i<s.size();++i){
7         int h=s[i];
8         pair<int,int> now=make_pair(h,i);
9         while(h<st.top().first){
10             now=st.top();
11             st.pop();
12             ans=max(ans,(LL)(i-now.second)*now.
13                 first);
14         }
15         if(h>st.top().first){
16             st.push(make_pair(h,now.second));
17         }
18     }
19     return ans;
20 }

```

12 zformula

12.1 formula

12.1.1 Pick 公式

給定頂點坐標均是整點的簡單多邊形，面積 = 內部格點數 + 邊上格點數/2-1

12.1.2 圖論

- 對於平面圖 $F = E - V + C + 1$ ， C 是連通分量數
- 對於平面圖 $E \leq 3V - 6$
- 對於連通圖 G ，最大獨立點集的大小設為 $I(G)$ ，最大匹配大小設為 $M(G)$ ，最小點覆蓋設為 $Cv(G)$ ，最小邊覆蓋設為 $Ce(G)$ 。對於任意連通圖：

- $I(G) + Cv(G) = |V|$
- $M(G) + Ce(G) = |V|$

- 對於連通二分圖：

- $I(G) = Cv(G)$
- $M(G) = Ce(G)$

- 最大權閉合圖：

- $C(u, v) = \infty, (u, v) \in E$
- $C(S, v) = W_v, W_v > 0$
- $C(v, T) = -W_v, W_v < 0$
- $ans = \sum_{W_v > 0} W_v - flow(S, T)$

- 最大密度子圖：

- 求 $\max \left(\frac{W_e + W_v}{|V|} \right), e \in E', v \in V'$
- $U = \sum_{v \in V} 2W_v + \sum_{e \in E} W_e$
- $C(u, v) = W_{(u,v)}, (u, v) \in E$ ，雙向邊
- $C(S, v) = U, v \in V$
- $D_u = \sum_{(u,v) \in E} W_{(u,v)}$
- $C(v, T) = U + 2g - D_v - 2W_v, v \in V$
- 二分搜 g ：
 $l = 0, r = U, eps = 1/n^2$
 if $((U \times |V| - flow(S, T))/2 > 0)$ $l = mid$
 else $r = mid$
- $ans = min_cut(S, T)$
- $|E| = 0$ 要特殊判斷

- 弦圖：

- 點數大於 3 的環都要有一條弦
- 完美消除序列從後往前依次給每個點染色，給每個點染上可以染的最小顏色
- 最大團大小 = 色數
- 最大獨立集：完美消除序列從前往後能選就選
- 最小團覆蓋：最大獨立集的點和他延伸的邊構成
- 區間圖是弦圖
- 區間圖的完美消除序列：將區間按造又端點由小到大排序
- 區間圖染色：用線段樹做

12.1.3 dinic 特殊圖複雜度

- 單位流： $O\left(\min\left(V^{3/2}, E^{1/2}\right)E\right)$
- 二分圖： $O\left(V^{1/2}E\right)$

12.1.4 0-1 分數規劃

$x_i = \{0, 1\}$ ， x_i 可能會有其他限制，求 $\max \left(\frac{\sum B_i x_i}{\sum C_i x_i} \right)$

- $D(i, g) = B_i - g \times C_i$
- $f(g) = \sum D(i, g) x_i$
- $f(g) = 0$ 時 g 為最佳解， $f(g) < 0$ 沒有意義
- 因為 $f(g)$ 單調可以二分搜 g
- 或用 Dinkelbach 通常比較快

```
1 binary_search(){
2   while(r-l>eps){
3     g=(l+r)/2;
4     for(i:所有元素)D[i]=B[i]-g*C[i]; //D(i, g)
5     找出一組合法x[i]使f(g)最大;
6     if(f(g)>0) l=g;
7     else r=g;
8   }
9   Ans = r;
10 }
11 Dinkelbach(){
12   g=任意狀態(通常設為0);
13   do{
14     Ans=g;
15     for(i:所有元素)D[i]=B[i]-g*C[i]; //D(i, g)
16     找出一組合法x[i]使f(g)最大;
17     p=0, q=0;
18     for(i:所有元素)
19       if(x[i])p+=B[i], q+=C[i];
20     g=p/q; //更新解，注意q=0的情況
21   }while(abs(Ans-g)>EPS);
22   return Ans;
23 }
```

12.1.5 學長公式

- $\sum_{d|n} \phi(n) = n$
- $g(n) = \sum_{d|n} f(d) \Rightarrow f(n) = \sum_{d|n} \mu(d) \times g(n/d)$
- Harmonic series $H_n = \ln(n) + \gamma + 1/(2n) - 1/(12n^2) + 1/(120n^4)$
- $\gamma = 0.57721566490153286060651209008240243104215$
- 格雷碼 $n \oplus (n >> 1)$
- $SG(A+B) = SG(A) \oplus SG(B)$
- 選轉矩陣 $M(\theta) = \begin{pmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{pmatrix}$

12.1.6 基本數論

- $\sum_{d|n} \mu(n) = [n == 1]$
- $g(m) = \sum_{d|m} f(d) \Leftrightarrow f(m) = \sum_{d|m} \mu(d) \times g(m/d)$
- $\sum_{i=1}^n \sum_{j=1}^m$ 互質數量 $= \sum \mu(d) \lfloor \frac{n}{d} \rfloor \lfloor \frac{m}{d} \rfloor$
- $\sum_{i=1}^n \sum_{j=1}^n lcm(i, j) = n \sum_{d|n} d \times \phi(d)$

12.1.7 排組公式

- k 卡特蘭 $\frac{C_n^{kn}}{n(k-1)+1} \cdot C_m^n = \frac{n!}{m!(n-m)!}$
- $H(n, m) \cong x_1 + x_2 + \dots + x_n = k, num = C_k^{n+k-1}$
- Stirling number of 2^{nd} , n 人分 k 組方法數目
 - $S(0, 0) = S(n, n) = 1$
 - $S(n, 0) = 0$
 - $S(n, k) = kS(n-1, k) + S(n-1, k-1)$
- Bell number, n 人分任意多組方法數目
 - $B_0 = 1$
 - $B_n = \sum_{i=0}^n S(n, i)$
 - $B_{n+1} = \sum_{k=0}^n C_k^n B_k$
 - $B_{p+n} \equiv B_n + B_{n+1} \pmod{p}$, p is prime
 - $B_{p^m+n} \equiv mB_n + B_{n+1} \pmod{p}$, p is prime
 - From $B_0 : 1, 1, 2, 5, 15, 52, 203, 877, 4140, 21147, 115975$
- Derangement, 錯排，沒有人在自己位置上
 - $D_n = n!(1 - \frac{1}{1!} + \frac{1}{2!} - \frac{1}{3!} + \dots + (-1)^n \frac{1}{n!})$
 - $D_n = (n-1)(D_{n-1} + D_{n-2}), D_0 = 1, D_1 = 0$
 - From $D_0 : 1, 0, 1, 2, 9, 44, 265, 1854, 14833, 133496$

6. Binomial Equality

- $\sum_k \binom{r}{m+k} \binom{s}{n-k} = \binom{r+s}{m+n}$
- $\sum_k \binom{r}{m+k} \binom{s}{n+k} = \binom{l+s}{l-m+n}$
- $\sum_k \binom{r}{m+k} \binom{s+k}{n} (-1)^k = (-1)^{l+m} \binom{s-m}{n-l}$
- $\sum_{k \leq l} \binom{l-k}{m} \binom{s}{k-n} (-1)^k = (-1)^{l+m} \binom{s-m-1}{l-n-m}$

- $\sum_{0 \leq k \leq l} \binom{l-k}{m} \binom{q+k}{n} = \binom{l+q+1}{m+n+1}$
- $\binom{r}{k} = (-1)^k \binom{k-r-1}{k}$
- $\binom{r}{m} \binom{m}{k} = \binom{r}{k} \binom{r-k}{m-k}$
- $\sum_{k \leq n} \binom{r+k}{k} = \binom{r+n+1}{n}$
- $\sum_{0 \leq k \leq n} \binom{k}{m} = \binom{n+1}{m+1}$
- $\sum_{k \leq m} \binom{m+r}{k} x^k y^k = \sum_{k \leq m} \binom{-r}{k} (-x)^k (x+y)^{m-k}$

12.1.8 冪次，冪次和

- $a^{b\%P} = a^{b\% \varphi(P) + \varphi(P)}, b \geq \varphi(P)$
- $1^3 + 2^3 + 3^3 + \dots + n^3 = \frac{n^4}{4} + \frac{n^3}{2} + \frac{n^2}{4}$
- $1^4 + 2^4 + 3^4 + \dots + n^4 = \frac{n^5}{5} + \frac{n^4}{2} + \frac{n^3}{3} - \frac{n}{30}$
- $1^5 + 2^5 + 3^5 + \dots + n^5 = \frac{n^6}{6} + \frac{n^5}{2} + \frac{5n^4}{12} - \frac{n^2}{12}$
- $0^k + 1^k + 2^k + \dots + n^k = P(k), P(k) = \frac{(n+1)^{k+1} - \sum_{i=0}^{k-1} C_k^{i+1} P(i)}{k+1}, P(0) = n+1$
- $\sum_{k=0}^{m-1} k^n = \frac{1}{n+1} \sum_{k=0}^n C_k^{n+1} B_k m^{n+1-k}$
- $\sum_{j=0}^m C_j^{m+1} B_j = 0, B_0 = 1$
- 除了 $B_1 = -1/2$ ，剩下的奇數項都是 0
- $B_2 = 1/6, B_4 = -1/30, B_6 = 1/42, B_8 = -1/30, B_{10} = 5/66, B_{12} = -691/2730, B_{14} = 7/6, B_{16} = -3617/510, B_{18} = 43867/798, B_{20} = -174611/330,$

12.1.9 Burnside's lemma

- $|X/G| = \frac{1}{|G|} \sum_{g \in G} |X^g|$
- $X^g = t^{c(g)}$
- G 表示有幾種轉法， X^g 表示在那種轉法下，有幾種是會保持對稱的， t 是顏色數， $c(g)$ 是循環節不動的面數。
- 正立方體塗三顏色，轉 0 有 3^6 個元素不變，轉 90 有 6 種，每種有 3^3 不變，180 有 3×3^4 ，120(角)有 8×3^2 ，180(邊)有 6×3^3 ，全部 $\frac{1}{24} (3^6 + 6 \times 3^3 + 3 \times 3^4 + 8 \times 3^2 + 6 \times 3^3) = 57$

12.1.10 Count on a tree

- Rooted tree: $s_{n+1} = \frac{1}{n} \sum_{i=1}^n (i \times a_i \times \sum_{j=1}^{\lfloor n/i \rfloor} a_{n+1-i \times j})$
- Unrooted tree:
 - Odd: $a_n - \sum_{i=1}^{n/2} a_i a_{n-i}$
 - Even: $Odd + \frac{1}{2} a_{n/2} (a_{n/2} + 1)$
- Spanning Tree
 - 完全圖 $n^n - 2$
 - 一般圖 (Kirchhoff's theorem) $M[i][i] = degree(V_i), M[i][j] = -1, \text{if have } E(i, j), 0 \text{ if no edge. delete any one row and col in } A, ans = det(A)$

Codebook - ss

Contents

1 Computational Geometry	1	4 Language	6	9 Tree Problem	11
1.1 Geometry	1	4.1 CNF	6	9.1 HeavyLight	11
1.2 SmallestCircle	3	5 Linear Programming	6	9.2 kruskal(MST)	11
1.3 最近點對	3	5.1 simplex	6	9.3 LCA	11
2 Data Structure	3	6 Number Theory	6	9.4 link cut tree	11
2.1 01 背包	3	6.1 basic	6	9.5 POJ tree	12
2.2 binary search	3	6.2 bit set	7	9.6 Prim(MST)	12
2.3 discretization	3	6.3 cantor expansion	7	9.7 segment tree	12
2.4 half enumeration	3	6.4 find real root	7	10 default	13
2.5 LCS	3	6.5 LinearCongruence	7	10.1 8 queen	13
2.6 LIS	3	6.6 Lucas	7	10.2 debug	13
2.7 skew heap	4	6.7 Matrix	7	10.3 IncStack	13
2.8 sliding window	4	6.8 matrix exponential	8	10.4 input	13
2.9 undo disjoint set	4	6.9 MillerRobin	8	10.5 randomize	13
3 Graph	4	6.10 NTT	8	10.6 模板	13
3.1 BFS	4	6.11 Simpson	8	11 other	13
3.2 DFS	4	6.12 SpeedExpo	8	11.1 WhatDay	13
3.3 dijkstra	4	6.13 外星模運算	8	11.2 上下最大正方形	13
3.4 graphISO	4	6.14 大數取模	9	11.3 最大矩形	13
3.5 MaximumClique	4	6.15 數位統計	9	12 zformula	14
3.6 MinimumMeanCycle	5	6.16 質因數分解	9	12.1 formula	14
3.7 topology sort	5	7 String	9	12.1.1 Pick 公式	14
3.8 union and find	5	7.1 AC 自動機	9	12.1.2 圖論	14
3.9 全局最小割	5	7.2 hash	10	12.1.3 dinic 特殊圖複雜度	14
3.10 最小樹形圖朱劉	5	7.3 KMP	10	12.1.4 0-1 分數規劃	14
		7.4 manacher	10	12.1.5 學長公式	14
		7.5 minimal string rotation	10	12.1.6 基本數論	14
		7.6 reverseBWT	10	12.1.7 排組公式	14
		7.7 suffix array lcp	10	12.1.8 冪次, 冪次和	14
		7.8 Z	10	12.1.9 Burnside's lemma	14
		8 Tarjan	10	12.1.10 Count on a tree	14
		8.1 tnfsb017 2 sat	10		
		8.2 雙連通分量 & 割點	11		

Codebook - ss

C++ Resource Test

```
1 #include <bits/stdc++.h>
2 using namespace std;
3
4 namespace system_test {
5
6     const size_t KB = 1024;
7     const size_t MB = KB * 1024;
8     const size_t GB = MB * 1024;
9
10    size_t block_size, bound;
11    void stack_size_dfs(size_t depth = 1) {
12        if (depth >= bound)
13            return;
14        int8_t ptr[block_size]; // 若無法編譯將
15                                // block_size 改成常數
16        memset(ptr, 'a', block_size);
17        cout << depth << endl;
18        stack_size_dfs(depth + 1);
19    }
20
21    void stack_size_and_runtime_error(size_t
22        block_size, size_t bound = 1024) {
23        system_test::block_size = block_size;
24        system_test::bound = bound;
25        stack_size_dfs();
26    }
27 }
```

```
24 }
25
26 double speed(int iter_num) {
27     const int block_size = 1024;
28     volatile int A[block_size];
29     auto begin = chrono::
30         high_resolution_clock::now();
31     while (iter_num--)
32         for (int j = 0; j < block_size; ++j)
33             A[j] += j;
34     auto end = chrono::
35         high_resolution_clock::now();
36     chrono::duration<double> diff = end -
37         begin;
38     return diff.count();
39 }
40
41 void runtime_error_1() {
42     // Segmentation fault
43     int *ptr = nullptr;
44     *(ptr + 7122) = 7122;
45 }
46
47 void runtime_error_2() {
48     // Segmentation fault
49     int *ptr = (int *)memset;
50     *ptr = 7122;
51 }
52
53 void runtime_error_3() {
54     // munmap_chunk(): invalid pointer
55     int *ptr = (int *)memset;
56     delete ptr;
57 }
```

```
56 void runtime_error_4() {
57     // free(): invalid pointer
58     int *ptr = new int[7122];
59     ptr += 1;
60     delete[] ptr;
61 }
62
63 void runtime_error_5() {
64     // maybe illegal instruction
65     int a = 7122, b = 0;
66     cout << (a / b) << endl;
67 }
68
69 void runtime_error_6() {
70     // floating point exception
71     volatile int a = 7122, b = 0;
72     cout << (a / b) << endl;
73 }
74
75 void runtime_error_7() {
76     // call to abort.
77     assert(false);
78 }
79
80 } // namespace system_test
81
82 #include <sys/resource.h>
83 void print_stack_limit() { // only work
84     in Linux
85     struct rlimit l;
86     getrlimit(RLIMIT_STACK, &l);
87     cout << "stack_size = " << l.rlim_cur
88         << " byte" << endl;
89 }
```