

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
57     的距離平方
58     point<T> v=p2-p1,v1=p-p1;
59     if(is_segment){
60     point<T> v2=p-p2;
61     if(v.dot(v1)<=0)return v1.abs2();
62     if(v.dot(v2)>=0)return v2.abs2();
63 }
64     T tmp=v.cross(v1);
65     return tmp*tmp/v.abs2();
66 }
67     T seg_dis2(const line<T> &l)const{//兩
68     線段距離平方
69     return min({dis2(l.p1,1),dis2(l.p2,1)
70     ,l.dis2(p1,1),l.dis2(p2,1)});
71 }
72     point<T> projection(const point<T> &p)
73     const{//點對直線的投影
74     point<T> n=(p2-p1).normal();
75     return p-n*(p-p1).dot(n)/n.abs2();
76 }
77     point<T> mirror(const point<T> &p)const
78     {
79     //點對直線的鏡射, 要先呼叫pton轉成一
80     般式
81     point<T> R;
82     T d=a*b+b*b;
83     R.x=(b*b*p.x-a*a*p.x-2*a*b*p.y-2*a*c)/
84     d;
85     R.y=(a*a*p.y-b*b*p.y-2*a*b*p.x-2*b*c)/
86     d;
87     return R;
88 }
89     bool equal(const line &l1)const{//直線相
90     等
91     return ori(l1.p1)==0&&ori(l1.p2)==0;
92 }
93     bool parallel(const line &l1)const{
94     return (p1-p2).cross(l1.p1-l1.p2)==0;
95 }
96     bool cross_seg(const line &l1)const{
97     return (p2-p1).cross(l1.p1-p1)*(p2-p1)
98     .cross(l1.p2-p1)<=0; //直線是否交
99     線段
100 }
101     int line_intersect(const line &l1)const{
102     //直線相交情況, -1無限多點、1交於
103     一點、0不相交
104     return parallel(l1)?(ori(l1.p1)
105     ==0?-1:0):1;
106 }
107     int seg_intersect(const line &l1)const{
108     T c1=ori(l1.p1), c2=ori(l1.p2);
109     T c3=l1.ori(p1), c4=l1.ori(p2);
110     if(c1==0&&c2==0){ //共線
111     bool b1=btw(l1.p1)>=0,b2=btw(l1.p2)
112     >=0;
113     T a3=l1.btw(p1),a4=l1.btw(p2);
114     if(b1&&b2&&a3==0&&a4==0) return 2;
115     if(b1&&b2&&a3>=0&&a4==0) return 3;
116     if(b1&&b2&&a3>=0&&a4>=0) return 0;
117     return -1; //無限交點
118 }else if(c1*c2<=0&&c3*c4<=0) return 1;
119     return 0; //不相交
120 }
121     point<T> line_intersection(const line &
122     l1)const{//*直線交點*/
123     point<T> a=p2-p1,b=l1.p2-l1.p1,s=l1.p1-
124     p1;
125     //if(a.cross(b)==0)return INF;
126     return p1+a*(s.cross(b)/a.cross(b));
127 }
128     point<T> seg_intersection(const line &l
129     1)const{//線段交點
130     int res=seg_intersect(l1);
131     if(res<=0) assert(0);
132     if(res==2) return p1;
133     if(res==3) return p2;
134     return line_intersection(l1);
135 }
136 };
137 template<typename T>
138 struct polygon{
139     polygon(){}
140     vector<point<T> > p; //逆時針順序
141     T area()const{//面積
142     T ans=0;
143     for(int i=p.size()-1,j=0;j<(int)p.
144     size();i=j++){
145     ans+=p[i].cross(p[j]);
146     return ans/2;
147 }
148     point<T> center_of_mass()const{//重心
149     T cx=0,cy=0,w=0;
150     for(int i=p.size()-1,j=0;j<(int)p.
151     size();i=j++){
152     T a=p[i].cross(p[j]);
153     cx+=(p[i].x+p[j].x)*a;
154     cy+=(p[i].y+p[j].y)*a;
155     w+=a;
156 }
157     return point<T>(cx/3/w,cy/3/w);
158 }
159     char ahas(const point<T>& t)const{//點
160     是否在簡單多邊形內, 是的話回傳1、
161     在邊上回傳-1、否則回傳0
162     bool c=0;
163     for(int i=0,j=p.size()-1;i<p.size();j
164     =i++){
165     if(line<T>(p[i],p[j]).
166     point_on_segment(t))return -1;
167     else if((p[i].y>t.y)!=p[j].y>t.y)
168     &&
169     t.x<(p[j].x-p[i].x)*(t.y-p[i].y)/(p
170     [j].y-p[i].y)+p[i].x)
171     c=!c;
172     return c;
173 }
174     char point_in_convex(const point<T>&x)
175     const{
176     int l=1,r=(int)p.size()-2;
177 }
178     while(l<=r){ //點是否在凸多邊形內, 是
179     的話回傳1、在邊上回傳-1、否則回
180     傳0
181     int mid=(l+r)/2;
182     T a1=(p[mid]-p[0]).cross(x-p[0]);
183     T a2=(p[mid+1]-p[0]).cross(x-p[0]);
184     if(a1>=0&&a2<=0){
185     T res=(p[mid+1]-p[mid]).cross(x-p
186     [mid]);
187     return res>0?1:(res>=0?-1:0);
188     }else if(a1<0)r=mid-1;
189     else l=mid+1;
190 }
191     return 0;
192 }
193     vector<T> getA()const{//凸包邊對x軸的夾
194     角
195     vector<T> res; //一定是遞增的
196     for(size_t i=0;i<p.size();++i)
197     res.push_back((p[(i+1)%p.size()]-p[
198     i]).getA());
199     return res;
200 }
201     bool line_intersect(const vector<T>&A,
202     const line<T> &l1)const{//O(LogN)
203     int f1=upper_bound(A.begin(),A.end()
204     ,(l1.p1-l1.p2).getA())-A.begin();
205     int f2=upper_bound(A.begin(),A.end()
206     ,(l1.p2-l1.p1).getA())-A.begin();
207     return l1.cross_seg(line<T>(p[f1],p[f2
208     ]));
209 }
210     polygon cut(const line<T> &l1)const{//凸
211     包對直線切割, 得到直線L左側的凸包
212     polygon ans;
213     for(int n=p.size(),i=n-1,j=0;j<n;i=j
214     ++){
215     if(l1.ori(p[i])>=0){
216     ans.p.push_back(p[i]);
217     if(l1.ori(p[j])<0)
218     ans.p.push_back(l1.
219     line_intersection(line<T>(
220     p[i],p[j])));
221     }else if(l1.ori(p[j])>0)
222     ans.p.push_back(l1.
223     line_intersection(line<T>(p[
224     i],p[j])));
225 }
226     return ans;
227 }
228     static bool monotone_chain_cmp(const
229     point<T>&a,const point<T>&b){ //
230     凸包排序函數
231     return (a.x<b.x)|| (a.x==b.x&&a.y<b.y)
232     ;
233 }
234     void monotone_chain(vector<point<T> > &
235     s){ //凸包
236     sort(s.begin(),s.end(),
237     monotone_chain_cmp);
238     p.resize(s.size()+1);
239     int m=0;
240     for(size_t i=0;i<s.size();++i){
241     while(m>=2&&(p[m-1]-p[m-2]).cross(s
242     [i]-p[m-2])<=0)--m;
243     p[m++]=s[i];
244 }
245     for(int i=s.size()-2,t=m+1;i>=0;--i){
246     while(m>=2&&(p[m-1]-p[m-2]).cross(s
247     [i]-p[m-2])<=0)--m;
248     p[m++]=s[i];
249 }
250     if(s.size()>1)--m;
251     p.resize(m);
252 }
253     T diam()const{//直徑
254     int n=p.size(),t=1;
255     T ans=0;p.push_back(p[0]);
256     for(int i=0;i<n;i++){
257     point<T> now=p[i+1]-p[i];
258     while(now.cross(p[t+1]-p[i])>now.
259     cross(p[t]-p[i]))t=(t+1)%n;
260     ans=max(ans,(p[i]-p[t]).abs2());
261 }
262     return p.pop_back(),ans;
263 }
264     T min_cover_rectangle()const{//最小覆蓋矩形
265     int n=p.size(),t=1,r=1,l;
266     if(n<3)return 0; //也可以做最小周長矩
267     形
268     T ans=1e99;p.push_back(p[0]);
269     for(int i=0;i<n;i++){
270     point<T> now=p[i+1]-p[i];
271     while(now.cross(p[t+1]-p[i])>now.
272     cross(p[t]-p[i]))t=(t+1)%n;

```

```

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     bool point_in(const point3D<T> &p)const
470         {
471         return triangle3D<T>(a,b,c).point_in(
472             p)&&triangle3D<T>(c,d,a).
473             point_in(p);
474     }
475 };
476 template<typename T>
477 struct convexhull3D{
478     static const int MAXN=1005;
479     struct face{
480         int a,b,c;
481         face(int a,int b,int c):a(a),b(b),c(c)
482             {}
483     };
484     vector<point3D<T>> pt;
485     vector<face> ans;
486     int fid[MAXN][MAXN];
487     void build(){
488         int n=pt.size();
489         ans.clear();
490         memset(fid,0,sizeof(fid));
491         ans.emplace_back(0,1,2);//注意不能共
492         //線
493         ans.emplace_back(2,1,0);
494         int ftop=0;
495         for(int i=3,ftop=1;i<n;++i,++ftop)
496             {
497             vector<face> next;
498             for(auto &f:ans){
499                 T d=(pt[i]-pt[f.a]).dot((pt[f.b]-
500                     pt[f.a]).cross(pt[f.c]-pt[f.
501                         a]));
502                 if(d<0)next.push_back(f);
503                 int ff=0;
504                 if(d>0)ff=ftop;
505                 else if(d<0)ff=-ftop;
506                 fid[f.a][f.b]=fid[f.b][f.c]=fid[f.
507                     c][f.a]=ff;
508             }
509             for(auto &f:ans){
510                 if(fid[f.a][f.b]>0 && fid[f.a][f.
511                     b]!=fid[f.b][f.a])
512                     next.emplace_back(f.a,f.b,i);
513                 if(fid[f.b][f.c]>0 && fid[f.b][f.
514                     c]!=fid[f.c][f.b])
515                     next.emplace_back(f.b,f.c,i);
516                 if(fid[f.c][f.a]>0 && fid[f.c][f.
517                     a]!=fid[f.a][f.c])
518                     next.emplace_back(f.c,f.a,i);
519             }
520             ans=next;
521         }
522     }
523 };

```

```

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

```

```

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(sa, sa+
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(i = 1; i <= text1.size(); i++){
6     for(j = 1; j <= text2.size(); j++){
7         if(text1[i-1] == text2[j-1])
8             dp[i][j] = dp[i-1][j-1] + 1;
9         else
10             dp[i][j] = max(dp[i-1][j],
11                 dp[i][j-1]);
12     }
13 }
14 cout << dp[text1.size()][text2.size()]
15 ];

```

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;

```

```

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?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;

```

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 neighbor,weight of edge
10 LL dis[N],parent[N];
11 bool visit[N] = {false};
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     //[],vertex 技巧性讓最小值pop
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             if(w + dis[v] < dis[e]){
37                 dis[e] = w + dis[v];
38                 parent[e] = v;
39                 PQ.push({-dis[e],e});
40             }
41         }
42     }
43     LL maxd = -1,cnt = 0,far;
44     for(i = 0; i < n; i++){
45         if(dis[i] < oo){
46             if(dis[i] > maxd)
47                 maxd = dis[i],far = i;
48         }
49         else
50             cnt++;//for can't reach
51     }
52     cout << maxd << endl << cnt;
53     return 0;
54 }

```

3.4 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    }
25    if(tail < n) cout << "not a DAG" <<
26        endl;
27    else{
28        for(LL i = 0; i < n; i++)
29            cout << topo[i] << ' ';
30    }
31    return 0;
32 }

```

3.5 union and find

```

1 #include<bits/stdc++.h>
2 #define good ios_base::sync_with_stdio(0)
3 ;cin.tie(0)
4 #define RSIZE 101
5 typedef long long LL;
6 using namespace std;
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 LL sfind(LL dots){//find Leader,Leader's
12     parent = size of set
13     if(parent[dots] < 0)
14         return dots;
15     return parent[dots] = sfind(parent[
16         dots]);
17 }
18 LL BFS(LL now,LL root){//find root and
19     return size
20     parent[now] = root;
21     LL cnt = 1;
22     for(int k = 0; k < 4; k++){//4
23         directions
24         int u = now+dxy[k];
25         if(graph[u] == 1 && parent[u] ==
26             -1)//unvisited
27             cnt += BFS(u,root);
28     }

```



```

24     }
25     return cnt;
26 }
27 void combine(LL u, LL v) { //merge two sets
28     LL set1 = sfind(u), set2 = sfind(v);
29     if(set1 == set2) return; //same set
30
31     max_area = max(max_area, -parent[set1]
32                     ]-parent[set2]);
33     now_area--; //merge -> 2 pools become
34     1
35     if(parent[set1] < parent[set2]) { //1
36         is larger
37         parent[set1] += parent[set2];
38         parent[set2] = set1;
39     }
40     else {
41         parent[set2] += parent[set1];
42         parent[set1] = set2;
43     }
44     return;
45 }
46 int main() {
47     good;
48     //freopen("file name", "r", stdin);
49     //input redirection
50     LL i, j, m, n, k;
51     cin >> m >> n >> k;
52     memset(parent, -1, sizeof(parent));
53     for(i = 1; i <= m; i++) {
54         for(j = 1; j <= n; j++)
55             cin >> graph[i*(n+2)+j];
56     }
57     n += 2;
58     dxy[2] = n, dxy[3] = -n;
59     LL mn = (m+1)*n;
60     for(LL x = n; x < mn; x++) {
61         if(graph[x] == 1 && parent[x] ==
62            -1) { //unvisited
63             parent[x] = -BFS(x, x); //first
64             point consider as root
65             now_area++;
66             max_area = max(max_area, -
67                parent[x]);
68         }
69     }
70     LL ans = now_area, max_ans = max_area;
71     while(k--) {
72         LL x, y, temp;
73         cin >> x >> y;
74         temp = x*n+y;
75         if(graph[temp] == 1) continue;
76         graph[temp] = 1;
77         now_area++;
78         max_area = max(max_area, (LL)1);
79         for(i = 0; i < 4; i++) {
80             if(graph[temp+dxy[i]] == 0)
81                 continue;
82             combine(temp, temp+dxy[i]);
83         }
84         ans += now_area;
85         max_ans += max_area;
86     }
87     cout << max_ans << endl << ans;
88     return 0;
89 }

```

4 Number Theory

4.1 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  }
7  long long int phi[N+1];
8  void phiTable() {
9      for(int i=1; i<=N; i++) phi[i]=i;
10     for(int i=1; i<=N; i++) for(x=i*2; x<=N; x+=
11        i) phi[x]-=phi[i];
12 }
13 void all_divdown(const LL &n) { // all n/x
14     for(LL a=1; a<=n; a=n/(n/(a+1))) {
15         // dosomething;
16     }
17 }
18 const int MAXPRIME = 1000000;
19 int iscom[MAXPRIME], prime[MAXPRIME],
20     primecnt;
21 int phi[MAXPRIME], mu[MAXPRIME];

```

```

19 void sieve(void) {
20     memset(iscom, 0, sizeof(iscom));
21     primecnt = 0;
22     phi[1] = mu[1] = 1;
23     for(int i=2; i<MAXPRIME; ++i) {
24         if(!iscom[i]) {
25             prime[primecnt++] = i;
26             mu[i] = -1;
27             phi[i] = i-1;
28         }
29         for(int j=0; j<primecnt; ++j) {
30             int k = i * prime[j];
31             if(k>MAXPRIME) break;
32             iscom[k] = prime[j];
33             if(i%prime[j]==0) {
34                 mu[k] = 0;
35                 phi[k] = phi[i] * prime[j];
36                 break;
37             } else {
38                 mu[k] = -mu[i];
39                 phi[k] = phi[i] * (prime[j]-1);
40             }
41         }
42     }
43 }
44
45 bool g_test(const LL &g, const LL &p,
46             const vector<LL> &v) {
47     for(int i=0; i<v.size(); ++i)
48         if(modexp(g, (p-1)/v[i], p) != 1)
49             return false;
50     return true;
51 }
52 LL primitive_root(const LL &p) {
53     if(p==2) return 1;
54     vector<LL> v;
55     Factor(p-1, v);
56     v.erase(unique(v.begin(), v.end()), v.
57        end());
58     for(LL g=2; g<p; ++g)
59         if(g_test(g, p, v))
60             return g;
61     puts("primitive_root NOT FOUND");
62     return -1;
63 }
64 int Legendre(const LL &a, const LL &p) {
65     return modexp(a%p, (p-1)/2, p);
66 }
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
74 int inv[maxN];
75 LL invtable(int n, LL P) {
76     inv[1]=1;
77     for(int i=2; i<=n; ++i)
78         inv[i]=(P-(P/i))*inv[P%i]%P;
79 }
80
81 LL log_mod(const LL &a, const LL &b,
82            const LL &p) {
83     // a^x = b (mod p)
84     int m=sqrt(p+.5), e=1;
85     LL v=inv(modexp(a, m, p), p);
86     map<LL, int> x;
87     x[1]=0;
88     for(int i=1; i<=m; ++i) {
89         e = LLmul(e, a, p);
90         if(!x.count(e)) x[e] = i;
91     }
92     for(int i=0; i<=m; ++i) {
93         if(x.count(b)) return i*m + x[b];
94         b = LLmul(b, v, p);
95     }
96     return -1;
97 }
98
99 LL Tonelli_Shanks(const LL &n, const LL &
100 p) {
101     // x^2 = n (mod p)
102     if(n==0) return 0;
103     if(Legendre(n, p)!=1) while(1) { puts("
104         SQRT ROOT does not exist"); }
105     int S = 0;
106     LL Q = p-1;
107     while(! (Q&1)) { Q>>=1; ++S; }
108     if(S==1) return modexp(n%p, (p+1)/4, p);
109     LL z = 2;
110     for(; Legendre(z, p)!=-1; ++z)
111         LL c = modexp(z, Q, p);
112     LL R = modexp(n%p, (Q+1)/2, p), t =
113         modexp(n%p, Q, p);
114     int M = S;
115     while(1) {
116         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; 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         ];
147     for(int i=0; i<(int)a.size(); ++i) {
148         tM=M/m[i];
149         ans=(ans+(a[i]*tM%M)*pow_mod(tM, Euler
150             (m[i])-1, m[i])%M)%M;
151     }
152     /*如果m[i]是質數 · Euler(m[i])-1=m[i]
153        j-2 · 就不用算Euler了*/
154 }
155 return ans;
156 }
157
158 //java code
159 //求sqrt(N)的連分數
160 public static void Pell(int n) {
161     BigInteger N, p1, p2, q1, q2, a0, a1, a2, g1, g2
162         , h1, h2, p, q;
163     g1=q2=p1=BigInteger.ZERO;
164     h1=q1=p2=BigInteger.ONE;
165     a0=a1=BigInteger.valueOf((int)Math.sqrt
166         (1.0*n));
167     BigInteger ans=a0.multiply(a0);
168     if(ans.equals(BigInteger.valueOf(n))) {
169         System.out.println("No solution!");
170         return;
171     }
172     while(true) {
173         g2=a1.multiply(h1).subtract(g1);
174         h2=N.subtract(g2.pow(2)).divide(h1);
175         a2=g2.add(a0).divide(h2);
176         p=a1.multiply(p2).add(p1);
177         q=a1.multiply(q2).add(q1);
178         if(p.pow(2).subtract(N.multiply(q.
179             pow(2))).compareTo(BigInteger.
180                 ONE)==0) break;
181         g1=g2; h1=h2; a1=a2;
182         p1=p2; p2=p;
183         q1=q2; q2=q;
184     }
185     System.out.println(p+" "+q);
186 }

```

4.2 bit set

```

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

```

15 }

4.3 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            return true;
49        }
50        T gas(){
51            vector<T> lazy(r,1);
52            bool sign=false;
53            for(int i=0;i<r;++i){
54                if(m[i][i]==0){
55                    int j=i+1;
56                    while(j<r&&!m[j][i])j++;
57                    if(j==r)continue;
58                    m[i].swap(m[j]);
59                    sign=!sign;
60                }
61                for(int j=0;j<r;++j){
62                    if(i==j)continue;
63                    lazy[j]=lazy[j]*m[i][i];
64                    T mx=m[j][i];
65                    for(int k=0;k<c;++k)
66                        m[j][k]=m[j][k]*m[i][i]-m[i][k]*mx;
67                }
68            }
69            T det=sign?-1:1;
70            for(int i=0;i<r;++i){
71                det = det*m[i][i];
72                det = det/lazy[i];
73            }
74            return det;
75        }
76    }
77 };

```

4.4 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 << "900RC^2" << n+2 << "9μ-0";
20     exp(u,n);
21 }

```

4.5 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 }

```

4.6 外星模運算

```

1 //a[0]^a[1]^a[2]^...
2 #define maxn 1000000
3 int euler[maxn+5];
4 bool is_prime[maxn+5];
5 void init_euler(){
6     is_prime[1]=1; //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 }
43 LL a[1000005];
44 int t,mod;
45 int main(){
46     init_euler();
47     scanf("%d",&t);
48     #define n 4
49     while(t--){
50         for(int i=0;i<n;++i)scanf("%lld",&a[i]);
51         scanf("%d",&mod);
52         printf("%lld\n",high_pow(a,n,mod));
53     }
54     return 0;
55 }

```

4.7 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 calcmo(LL index,LL p){
11     if(index == 0) return base[index]- '0';
12     ;
13     LL single = calcmo(index-1,p)*10;
14     return (single%p + base[index]- '0')%p;
15 }

```

4.8 模逆元

```

1 #include<bits/stdc++.h>
2 #define good ios_base::sync_with_stdio(0)
3 ;cin.tie(0)
4 typedef long long LL;
5 using namespace std;
6 LL mod_inverse_by_speed_exp(LL x,LL y,LL
7     p){
8     if(y == 0) return 1;
9     if(y % 2) return (
10         mod_inverse_by_speed_exp(x,y-1,p)
11         )%x%p;
12     else{
13         LL temp =
14             mod_inverse_by_speed_exp(x,y
15             /2,p);
16         return (temp*temp)%p;
17     }
18 }
19 int main(){
20     good;
21     LL n,i,p,x;
22     cin >> n >> p;
23     for(i = 0; i < n; i++){
24         cin >> x;
25         cout << mod_inverse_by_speed_exp(
26             x,p-2,p) << ' ';
27     }
28     return 0;
29 }

```

4.9 質因數分解

```

1 LL func(const LL n,const LL mod,const int
2     c) {
3     return (LLmul(n,n,mod)+c+mod)%mod;
4 }
5 LL pollorrho(const LL n, const int c) { //
6     循環節長度
7     LL a=1, b=1;
8     a=func(a,n,c)%n;
9     b=func(b,n,c)%n; b=func(b,n,c)%n;
10    while(gcd(abs(a-b),n)!=1) {
11        a=func(a,n,c)%n;
12        b=func(b,n,c)%n; b=func(b,n,c)%n;
13    }
14    return gcd(abs(a-b),n);
15 }
16 void prefactor(LL &n, vector<LL> &v) {
17     for(int i=0;i<12;++i) {
18         while(n%prime[i]==0) {
19             v.push_back(prime[i]);
20             n/=prime[i];
21         }
22     }
23 }
24 void smallfactor(LL n, vector<LL> &v) {
25     if(n<MAXPRIME) {
26         while(isp[(int)n]) {
27             v.push_back(isp[(int)n]);
28             n/=isp[(int)n];
29         }
30         v.push_back(n);
31     } else {
32         for(int i=0;i<primecnt&&prime[i]*
33             prime[i]<=n;++i) {

```

```

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

```

5 String

5.1 manacher(最小回文字串)

```

1 //原字串: asdsasdsa
2 //要先把字串變成這樣: @#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*i-l],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 }

```

6 Tree Problem

6.1 kruskal(MST)

```

1 #include<bits/stdc++.h>
2 #define good ios_base::sync_with_stdio(0)
3 #define RSIZE 10002
4 #define pll pair<LL,LL>
5 #define ll long long LL;
6 using namespace std;
7 struct EDGE{
8     LL u,v,w;

```

```

10 };
11
12 vector<EDGE> adjacent;//out neighbor,
13     weight
14 LL fa[RSIZE];
15 bool cmp(EDGE &a,EDGE &b){//sort by
16     weight
17     return a.w < b.w;
18 }
19 LL sfind(LL now){//find root,root's
20     father=set size
21     if(fa[now] < 0)
22         return now;
23     return fa[now] = sfind(fa[now]);
24 }
25 bool merge(LL u,LL v){//find two root,
26     comparing size(by root's father)
27     LL set1 = sfind(u),set2 = sfind(v);
28     if(set1 == set2) return false;//same
29     root-> no need to merge
30     if(fa[set1] < fa[set2]){ //set1 is
31         larger
32         fa[set1] += fa[set2];
33         fa[set2] = set1;
34     }
35     else{
36         fa[set2] += fa[set1];
37         fa[set1] = set2;
38     }
39     return true;
40 }
41 int main(){
42     good;
43     //freopen("file name", "r", stdin);
44     //input redirection
45     LL i,n,m;
46     cin >> n >> m;
47     for(i = 0; i < m; i++){
48         LL x,y,weight;
49         cin >> x >> y >> weight;
50         adjacent.push_back({x,y,weight});
51     }
52     memset(fa,-1,sizeof(fa)); //unvisited
53     sort(adjacent.begin(),adjacent.end(),
54         cmp); //sort by weight
55     LL cost = 0,now_edge = 0;
56     for(EDGE e : adjacent){
57         if(merge(e.u,e.v)){//connect edge
58             cost += e.w;
59             now_edge++;
60         }
61         if(now_edge < n-1){ //not a MST
62             cout << -1 << endl;
63         }
64         else
65             cout << cost << endl;
66         return 0;
67 }

```

6.2 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    }
11    for(auto &i:G[x]){
12        if(i==p)continue;
13        dep[i]=dep[x]+1;
14        dfs(i,x);
15    }
16 }
17 inline int jump(int x,int d){
18     for(int i=0;i<MLG;i++){
19         if((d>>i)&1) x=pa[i][x];
20     }
21     return x;
22 }
23 inline int find_lca(int a,int b){
24     if(dep[a]>dep[b])swap(a,b);
25     b=jump(b,dep[b]-dep[a]);
26     if(a==b)return a;
27     for(int i=MLG;i>0;i--){
28         if(pa[i][a]!=pa[i][b]){
29             a=pa[i][a];
30             b=pa[i][b];
31         }
32     }
33     return pa[0][a];

```

6.3 Prim(MST)

```

1 #include<bits/stdc++.h>
2 #define good ios_base::sync_with_stdio(0)
3 #define RSIZE 502
4 #define oo 1000000001 //1e9+1
5 #define ll long long LL;
6 using namespace std;
7
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 }

```

6.4 segment tree

```

1 #include<bits/stdc++.h>
2 #define good ios_base::sync_with_stdio(0)
3 #define RSIZE 100000
4 #define pll pair<LL,LL>
5 #define ll 2*index //c means child
6 #define rc 2*index+1
7 #define ll long long LL;
8 using namespace std;
9
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)); //choose better one
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 }

```

7 default

7.1 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     }
6     do{
7         bool valid = true;
8         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;
12                 break;
13             }
14         }
15         if(valid) total++;
16         while (next_permutation(p,p+n));
17         return total;
18 }

```

7.2 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

```

7.3 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:102400000,102400000")
8 //Linux stack resize
9 #include<sys/resource.h>
10 void increase_stack(){
11     const rlim_t ks=64*1024*1024;
12     struct rlimit rl;
13     int res=getrlimit(RLIMIT_STACK,&rl);
14     if(!res&&rl.rlim_cur<ks){
15         rl.rlim_cur=ks;
16         res=setrlimit(RLIMIT_STACK,&rl);
17     }
18 }

```

7.4 input

```

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

```

7.5 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++;

```

7.6 sweepline

```

1 #include<bits/stdc++.h>
2
3 #define good ios_base::sync_with_stdio(0);cin.tie(0)
4 typedef long long LL;
5 using namespace std;
6
7 struct Seg{
8     LL left,right;
9 };
10 bool cmp(Seg &a,Seg &b){
11     return a.left < b.left;
12 }
13
14 int main(){
15     good;
16     //freopen("P_2_1_5.in", "r", stdin)
17     //input redirection
18     LL n;
19     cin >> n;
20     Seg line[n];
21     for(LL i = 0; i < n; i++)
22         cin >> line[i].left >> line[i].right;
23     sort(line,line+n,cmp);
24     Seg last = line[0];
25     LL total = 0;
26     for(LL i = 1; i < n; i++){
27         if(line[i].left > last.right){
28             total += last.right - last.left;
29             last = line[i];
30             continue;
31         }
32         last.right = max(last.right, line[i].right); //merge last and line[i]
33     }
34     total += last.right - last.left;
35     cout << total;
36     return 0;
37 }

```

7.7 模板

```

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

```

8 other

8.1 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==9&&d<3)
4         return (d+2*m+3*(m+1)/5+y+y/4+5)%7;
5     return (d+2*m+3*(m+1)/5+y+y/4-y/100+y/400)%7;
6 }

```


9 zformula

9.1 formula

9.1.1 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

9.1.2 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

9.1.3 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： $O\left(V^{1/2}E\right)$

9.1.4 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 }
```

9.1.5 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 $M(\theta) = \begin{pmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{pmatrix}$

9.1.6 基本數論

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

9.1.7 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 $D_0 : 1, 0, 1, 2, 9, 44, 265, 1854, 14833, 133496$

- Binomial Equality

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

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

9.1.8 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
- $B_2 = 1/6, B_4 = -1/30, B_6 = 1/42, B_8 = -1/30, B_{10} = 5/66, B_{12} = -691/2730, B_{14} = 7/6, B_{16} = -3617/510, B_{18} = 43867/798, B_{20} = -174611/330,$

9.1.9 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 $\frac{1}{24} (3^6 + 6 \times 3^3 + 3 \times 3^4 + 8 \times 3^2 + 6 \times 3^3) = 57$

9.1.10 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	3.2 DFS	4	7 default	8
1.1 Geometry	1	3.3 dijkstra	4	7.1 8 queen	8
1.2 SmallestCircle	3	3.4 topology sort	4	7.2 debug	8
1.3 最近點對	3	3.5 union and find	4	7.3 IncStack	8
2 Data Structure	3	4 Number Theory	5	7.4 input	8
2.1 01 背包	3	4.1 basic	5	7.5 randomize	8
2.2 binary search	3	4.2 bit set	5	7.6 sweepline	8
2.3 discretization	3	4.3 Matrix	6	7.7 模板	8
2.4 half enumeration	3	4.4 matrix exponential	6	8 other	8
2.5 LCS	3	4.5 SpeedExpo	6	8.1 WhatDay	8
2.6 LIS	3	4.6 外星模運算	6	9 zformula	9
2.7 skew heap	4	4.7 大數取模	6	9.1 formula	9
2.8 sliding window	4	4.8 模逆元	6	9.1.1 Pick 公式	9
2.9 undo disjoint set	4	4.9 質因數分解	6	9.1.2 圖論	9
3 Graph	4	5 String	7	9.1.3 dinic 特殊圖複雜度	9
3.1 BFS	4	5.1 manacher(最小回文字串)	7	9.1.4 0-1 分數規劃	9
		6 Tree Problem	7	9.1.5 學長公式	9
		6.1 kruskal(MST)	7	9.1.6 基本數論	9
		6.2 LCA	7	9.1.7 排組公式	9
		6.3 Prim(MST)	7	9.1.8 冪次, 冪次和	9
		6.4 segment tree	7	9.1.9 Burnside's lemma	9
				9.1.10 Count on a tree	9

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 }
```