1 Computational Geometry

1.1 SmallestCircle

```
using PT=point<T>; using CPT=const PT;
PT circumcenter(CPT &a,CPT &b,CPT &c){
  PT u=b-a, v=c-a;
  T c1=u.abs2()/2,c2=v.abs2()/2;
  T d=u.cross(v);
  return PT(a.x+(v.y*c1-u.y*c2)/d,a.y+(u.
      x*c2-v.x*c1)/d);
}
void solve(PT p[],int n,PT &c,T &r2){
  random_shuffle(p,p+n);
  c=p[0]; r2=0; // c,r2 = 圓心,半徑平方
for(int i=1;i<n;i++)if((p[i]-c).abs2()>r2
    ){
    c=p[i]; r2=0;
for(int j=0;j<i;j++)if((p[j]-c).abs2()>r2
    ){
      c.x=(p[i].x+p[j].x)/2;
      c.y=(p[i].y+p[j].y)/2;
      r2=(p[j]-c).abs2();
for(int k=0;k<j;k++)if((p[k]-c).abs2()>r2
    ){
        c=circumcenter(p[i],p[j],p[k]);
        r2=(p[i]-c).abs2();
      }
    }
  }
}
```

1.2 最近點對

```
template<typename _IT=point<T>* >
T cloest_pair(_IT L, _IT R){
  if(R-L <= 1) return INF;
  _IT mid = L+(R-L)/2;
  T x = mid->x;
  T d = min(cloest_pair(L,mid),
      cloest_pair(mid,R));
  inplace_merge(L, mid, R, ycmp);
  static vector<point> b; b.clear();
  for(auto u=L;u<R;++u){
    if((u->x-x)*(u->x-x)>=d) continue;
    for(auto v=b.rbegin();v!=b.rend();++v
        ){
      T dx=u->x-v->x, dy=u->y-v->y;
      if(dy*dy>=d) break;
      d=min(d,dx*dx+dy*dy);
    }
    b.push_back(*u);
  }
  return d;
}
T closest_pair(vector<point<T>> &v){
  sort(v.begin(),v.end(),xcmp);
  return closest_pair(v.begin(),v.end());
}
```

2 Data Structure

2.1 01 背包

```
LL dp[101][100001] = {0};//前i個物品所湊
    出重量j的最大價值
int main(){
    good;
    LL j,i,n,w,svalue = 0,sweight = 0;
    cin >> n >> w;
    pair<LL,LL> item[n+1];//weight,value;
    for(i = 1; i <= n; i++)
        cin >> item[i].first;
    for(i = 1; i <= n; i++)
        cin >> item[i].second;
    for(i = 0; i <= n; i++)
        dp[i][0] = dp[0][i] = 0;
    for(i = 1; i <= n; i++){
        for(j = 1; j <= w; j++){
            if(item[i].first > j)
                dp[i][j] = dp[i-1][j];
            else
                dp[i][j] = max(dp[i-1][j
                    ],item[i].second +
                    dp[i-1][j-item[i].
                    first]);
        }
    }
    cout << dp[n][w];
    return 0;
}
```

2.2 array simulate linked list

```
#include<bits/stdc++.h>
#define pll pair<LL,LL>
#define RSIZE 100002
#define oo 1000000001
#define good ios_base::sync_with_stdio(0)
    ;cin.tie(0)
typedef long long LL;
using namespace std;

struct {
    LL center,height;
    LL pre,next;
    bool alive;
}tree[RSIZE];

queue<LL> Q;//check removed tree
void removable(LL index){
    if(!tree[index].alive) return;
    LL s = tree[index].pre,t = tree[index
        ].next;
    if(tree[index].center-tree[index].
        height >= tree[s].center ||
       tree[index].center+tree[index].
           height <= tree[t].center){
        tree[index].alive = false;
        Q.push(index);
        tree[s].next = t;
        tree[t].pre = s;
    }
}

int main(){
    good;
    LL bound,n,total = 0,high = 0;
    cin >> n >> bound;
    for(LL i = 1; i <= n; i++)
        cin >> tree[i].center;
    for(LL i = 1; i <= n; i++)
        cin >> tree[i].height;
    for(LL i = 1; i <= n; i++){
        tree[i].pre = i-1;
        tree[i].next = i+1;
        tree[i].alive = true;
    }
    tree[0].center = 0,tree[0].height =
        oo;
    tree[n+1].center = bound,tree[n+1].
        height = oo;
    for(LL i = 1; i <= n; i++)
        removable(i);
        while (!Q.empty()){
            LL v = Q.front();
            Q.pop();
            total++;
            high = max(high,tree[v].
                height);
            removable(tree[v].pre);
            removable(tree[v].next);
        }
        cout << total << endl << high;
    return 0;
}
```

2.3 binary search

```
LL BS(LL left,LL right){
    if(left+1 >= right)//break condition
        return -1;
    LL mid = (left+right)/2;
    if(arr[mid] == target)
        return mid;
    else if(arr[mid] < target){
        left = mid+1;
        BS(left,right);
    }
    else if(arr[mid] > target){
        right = mid;
        BS(left,right);
    }
}
```

2.4 discretization

```
map<LL,LL> S;
    for (LL i=0;i<n;i++)
        S[a[i]] = 0; // insert a[i] and
            set rank=0
    LL r=0;
    for (auto it=S.begin(); it!=S.end();
        ++it)//traversal and set rank
        it->second = r++;
    // replace number with rank
    for (LL i=0;i<n;i++) {
        a[i] = S.lower_bound(a[i]) ->
            second;
        // find() return the iterator,
            then take the rank
        // or S.find(a[i]) -> second;
    }
```

2.5 half enumeration

```
#include<bits/stdc++.h>

#define good ios_base::sync_with_stdio(0)
    ;cin.tie(0)
typedef long long LL;
using namespace std;

LL sa[1<<18],sb[1<<18],no[1<<18];//subset
     product of a and b

LL subset(LL num[],LL length,LL product
    [],LL p){//pass by pointer
    LL k = 0,i,j;//count
    for(i = 0; i < length; i++){
        for(j = 0; j < k; j++){
            product[k+j] = (product[j]*
                num[i]) % p;//old
                product times num[i]
        }
        product[k] = num[i];//for num[i]
            itself
        k += k+1;
    }
    return k;//return the size of subset
}

LL exp_modp(LL x,LL y,LL p){
    if(y == 0) return 1;
    if(y % 2) return (exp_modp(x,y-1,p)*x
        ) % p;
    else{
        LL temp = exp_modp(x,y/2,p);
        return (temp*temp) % p;
    }
}
int main(){
    good;
    LL i,n,p;
    LL a[30],b[30];
    cin >> n >> p;
    int len_a = n/2,len_b = n - len_a;
    for(i = 0; i < len_a; i++)
        cin >> a[i];
    for(i = 0; i < len_b; i++)
        cin >> b[i];
    LL len_sa = subset(a,len_a,sa,p);
    LL len_sb = subset(b,len_b,sb,p);
    sort(sb,sb+len_sb);

    LL len_sb2 = 1;//len_sb2 followed by
        i below
    no[0] = 1;//assume not empty(check
        later)
    for(i = 1; i < len_sb; i++){
        if(sb[i] != sb[i-1]){//new
            element
            sb[len_sb2] = sb[i];
            no[len_sb2] = 1;
            len_sb2++;
        }
        else//old element
            no[len_sb2-1]++;
    }
    LL ans = (sb[0] == 1) ? no[0]%p : 0;
    for(i = 0; i < len_sa; i++){
        if(sa[i] == 1) ans = (ans+1) % p;
        LL y = exp_modp(sa[i],p-2,p);//
            module inverse
        int it = lower_bound(sb,sb+
            len_sb2,y) - sb;
        if(it < len_sb2 && sb[it] == y){
            ans = (ans + no[it]) % p;
```

```
        }
    }
    cout << ans << '\n';
    return 0;
}
```

2.6 LCS

```
int dp[1002][1002],i,j; //text1 前i個 & text2 前j個
    for(i = 0; i < 1002; i++)
        dp[i][0] = 0,dp[0][i] = 0;
    for(i = 1; i <= text1.size(); i++){
        //1 base <=
        for(j = 1; j <= text2.size(); j++){
            if(text1[i-1] == text2[j-1])
                dp[i][j] = dp[i-1][j-1]+1;
            else
                dp[i][j] = max(dp[i-1][j],dp[i][j-1]);
        }
    }
    cout << dp[text1.size()][text2.size()];
```

2.7 LIS

```
int main(){
    good;
    LL n,i,length = 0,num;
    cin >> n;
    LL last[RSIZE];//長度為it的最小可能結尾
    for(i = 0; i < n; i++){
        cin >> num;
        LL it = lower_bound(last,last+length,num)-last;
        last[it] = num;
        if(it == length) length++;
    }
    cout << length;
    return 0;
}
```

2.8 skew heap

```
node *merge(node *a,node *b){
  if(!a||!b) return a?a:b;
  if(b->data<a->data) swap(a,b);
  swap(a->l,a->r);
  a->l=merge(b,a->l);
  return a;
}
```

2.9 sliding window

```
//same size
for(i = 0; i < m; i++){//making first window
    LL color = discret[a[right]];
    cnt[color]++;
    if(cnt[color] == 1) n_color++;
    right++;
}
while(right < n){
    if(n_color == m)
        ans++;
    LL l_remove = discret[a[left]];
    cnt[l_remove]--;//remove left one
    left++;
    if(cnt[l_remove] == 0) n_color--;
    LL add = discret[a[right]];
    cnt[add]++,right++;//add next one
    if(cnt[add] == 1) n_color++;
}
```

2.10 undo disjoint set

```
struct DisjointSet {
  // save() is like recursive
  // undo() is like return
  int n, fa[MXN], sz[MXN];
  vector<pair<int*,int>> h;
  vector<int> sp;
  void init(int tn) {
    n=tn;
    for (int i=0; i<n; i++) sz[fa[i]=i]=1;
    sp.clear(); h.clear();
  }
  void assign(int *k, int v) {
    h.PB({k, *k});
    *k=v;
  }
  void save() { sp.PB(SZ(h)); }
  void undo() {
    assert(!sp.empty());
    int last=sp.back(); sp.pop_back();
    while (SZ(h)!=last) {
      auto x=h.back(); h.pop_back();
      *x.F=x.S;
    }
  }
  int f(int x) {
    while (fa[x]!=x) x=fa[x];
    return x;
  }
  void uni(int x, int y) {
    x=f(x); y=f(y);
    if (x==y) return ;
    if (sz[x]<sz[y]) swap(x, y);
    assign(&sz[x], sz[x]+sz[y]);
    assign(&fa[y], x);
  }
}djs;
```

3 Graph

3.1 bellman ford

```
#include<bits/stdc++.h>
#define good ios_base::sync_with_stdio(0);cin.tie(0)
#define RSIZE 101
#define pll pair<LL,LL>
#define lc 2*index
#define rc 2*index+1
#define maxdis 1000000009
typedef long long LL;
using namespace std;
struct Node{
    LL in,out,weight;
};
LL dis[RSIZE];
void bellman_ford(vector<Node> &edges,LL v,LL e,LL start){//v for vertex, e for edge
    //vector<LL> dis(v,LLONG_MAX);
    dis[start] = 0;
    //relaxation
    for(LL i = 0; i < v-1; i++){
        for(LL j = 0; j < e; j++){
            LL x = edges[j].in,y = edges[j].out;
            LL w = edges[j].weight;
            if(dis[x] != maxdis && dis[x]+w < dis[y])
                dis[y] = dis[x] + w;
        }
    }
    //check -weight round
    cout << "from " << start << "'s shortest path to each vertexes is:" << endl;
    for(LL i = 0; i < v; i++){
        cout << i << "\t\t" << dis[i] << endl;
    }
}
bool checkinf(vector<Node> &edges,LL e){
    for(LL i = 0; i < e; i++){
        LL x =edges[i].in,y = edges[i].out;
        LL w = edges[i].weight;
        if(dis[x] != maxdis && dis[x] + w < dis[y])
            return true;
    }
    return false;
}
int main(){
    good;
    LL vertex,edge;
    cin >> vertex >> edge;
    memset(dis,maxdis,sizeof(dis));
    vector<Node> graph(edge);
    for(LL i = 0; i < edge; i++)
        cin >> graph[i].in >> graph[i].out >> graph[i].weight;
    LL source;
    cin >> source;
    bellman_ford(graph,vertex,edge,source);
    if(checkinf(graph,edge))
        cout << "found negative round" << endl;
    else
        cout << "not found negative round" << endl;
    return 0;
}
```

3.2 BFS

```
LL val;//unnecessary
bool visited[5000] = {false};
vector<LL> graph[5000];
void BFS(LL start) {
    queue<LL> q;
    q.push(start);
    visited[start] = true;
    while (!q.empty()){
        LL curr = q.front();
        q.pop();
        for(auto it: graph[curr]){
            if(!visited[it]){
                q.push(it);
                visited[it] = true;
            }
        }
    }
}
```

3.3 DFS

```
#include<bits/stdc++.h>
#define good ios_base::sync_with_stdio(0);cin.tie(0);cout.tie(0)
typedef long long LL;
using namespace std;
int fa[100000],d[100000] = {0};//unnecessary
bool visit[100000] = {false};
vector<LL> v[100000];
void dfs(LL now,LL depth){
    for(auto x:v[now]){
        if(!visit[x]){
            cout << x << ' ';
            visit[x] = true;
            d[x] = depth;
            fa[x] = now;
            dfs(x,depth+1);
        }
    }
}
int main(){
    good;
    LL i,n,a,b;
    cin >> n;
    for(i = 0; i < n; i++){
        cin >> a >> b;
        v[a].push_back(b);
        v[b].push_back(a);
    }
    dfs(0,1);
    return 0;
}
```

3.4 dijkstra

```
#include<bits/stdc++.h>
#define good ios_base::sync_with_stdio(0);cin.tie(0)
#define N 10002
```

```cpp
#define oo 1000000001//1e9+1
typedef long long LL;
using namespace std;

vector<pair<LL,LL>> adjacent[N];//out neighbor,weight of edge
LL dis[N],parent[N];
bool visit[N] = {false};

int main(){
    LL i,n,m;
    cin >> n >> m;
    for(i = 0; i < m; i++){
        LL x,y,w;
        cin >> x >> y >> w;
        adjacent[x].push_back({y,w});
        adjacent[y].push_back({x,w});
    }
    //initial
    LL source = 0;
    memset(dis,oo,sizeof(dis));
    memset(parent,-1,sizeof(parent));
    priority_queue<pair<LL,LL>> PQ;//-dis[],vertex，技巧性讓最小值pop
    PQ.push({dis[source] = 0,source});
    //dijkstra
    while (!PQ.empty()){
        auto p = PQ.top();
        PQ.pop();
        LL v = p.second;//vertex
        if(visit[v]) continue;
        visit[v] = true;
        for(auto it : adjacent[v]){
            LL e = it.first,w = it.second;
            if(w + dis[v] < dis[e]){
                dis[e] = w + dis[v];
                parent[e] = v;
                PQ.push({-dis[e],e});
            }
        }
    }
    LL maxd = -1,cnt = 0,far;
    for(i = 0; i < n; i++){
        if(dis[i] < oo){
            if(dis[i] > maxd)
                maxd = dis[i],far = i;
        }
        else
            cnt++;//for can't reach
    }
    cout << maxd << endl << cnt;
    return 0;
}
```

3.5 floyd warshall

```cpp
#include<bits/stdc++.h>
#define good ios_base::sync_with_stdio(0);cin.tie(0)
#define RSIZE 502
#define pll pair<LL,LL>
#define lc 2*index
#define rc 2*index+1
#define far 1000000001//1e9+1
typedef long long LL;
using namespace std;

LL dis[RSIZE][RSIZE];
vector<pll> adjacent[RSIZE];//out neighbor,weight
void floyd_warshall(LL vertex){
    LL t,k,i,j;
    for(t = 1; t <= vertex; t++)
        dis[t][t] = 0;
    for(k = 1; k <= vertex; k++){
        for(i = 1; i <= vertex; i++){
            for(j = 1; j <= vertex; j++){
                if(dis[i][k] + dis[k][j] < dis[i][j])
                    dis[i][j] = dis[i][k]+dis[k][j];
            }
        }
    }
    return;
}
int main(){
    good;
    LL m,n,x;
    cin >> n >> m >> x;//n nodes, m edges ,from 1 to n
    memset(dis,far,sizeof(dis));
    for(LL i = 0; i < m; i++){
        LL a,b,w;
        cin >> a >> b >> w;
        adjacent[a].push_back({b,w});
        dis[a][b] = w;
    }
    return 0;
}
```

3.6 topology sort

```cpp
int main(){
    good;
    LL indeg[1002] = {0};
    vector<LL> graph[1002];
    LL n,m,a,b;
    cin >> n >> m;
    for(LL i = 0; i < m; i++){
        cin >> a >> b;
        graph[a].push_back(b);
        indeg[b]++;
    }
    LL topo[1002],head = 0,tail = 0;//?? queue
    for(LL i = 0; i < n; i++)
        if(indeg[i] == 0)
            topo[tail++] = i;
    while(head < tail){
        LL v = topo[head++];//get data and pop
        for(LL u : graph[v]){
            if(--indeg[u] == 0)
                topo[tail++] = u;
        }
    }
    if(tail < n) cout << "not a DAG" << endl;
    else{
        for(LL i = 0; i < n; i++)
            cout << topo[i] << ' ';
    }
    return 0;
}
```

3.7 union and find

```cpp
#include<bits/stdc++.h>
#define good ios_base::sync_with_stdio(0);cin.tie(0)
#define RSIZE 101
typedef long long LL;
using namespace std;

LL parent[503*503];
int graph[503*503] = {0};
int dxy[4] = {1,-1};
LL now_area = 0,max_area = 0;

LL sfind(LL dots){//find leader,leader's parent = size of set
    if(parent[dots] < 0)
        return dots;
    return parent[dots] = sfind(parent[dots]);
}
LL BFS(LL now,LL root){//find root and return size
    parent[now] = root;
    LL cnt = 1;
    for(int k = 0; k < 4; k++){//4 directions
        int u = now+dxy[k];
        if(graph[u] == 1 && parent[u] == -1)//unvisited
            cnt += BFS(u,root);
    }
    return cnt;
}
void combine(LL u,LL v){//merge two sets
    LL set1 = sfind(u),set2 = sfind(v);
    if(set1 == set2) return;//same set

    max_area = max(max_area,-parent[set1]-parent[set2]);
    now_area--;//merge -> 2 pools become 1
    if(parent[set1] < parent[set2]){//1 is larger
        parent[set1] += parent[set2];
        parent[set2] = set1;
    }
    else{
        parent[set2] += parent[set1];
        parent[set1] = set2;
    }
    return;
}
int main(){
    good;
    //freopen("file name", "r", stdin); //input redirection
    LL i,j,m,n,k;
    cin >> m >> n >> k;
    memset(parent,-1,sizeof(parent));
    for(i = 1; i <= m; i++){
        for(j = 1; j <= n; j++)
            cin >> graph[i*(n+2)+j];
    }
    n += 2;
    dxy[2] = n,dxy[3] = -n;
    LL mn = (m+1)*n;
    for(LL x = n; x < mn; x++){
        if(graph[x] == 1 && parent[x] == -1){//unvisited
            parent[x] = -BFS(x,x);//first point consider as root
            now_area++;
            max_area = max(max_area,-parent[x]);
        }
    }
    LL ans = now_area,max_ans = max_area;
    while(k--){
        LL x,y,temp;
        cin >> x >> y;
        temp = x*n+y;
        if(graph[temp] == 1) continue;
        graph[temp] = 1;
        now_area++;
        max_area = max(max_area,(LL)1);
        for(i = 0; i < 4; i++){
            if(graph[temp+dxy[i]] == 0)
                continue;
            combine(temp,temp+dxy[i]);
        }
        ans += now_area;
        max_ans += max_area;
    }
    cout << max_ans << endl << ans;
    return 0;
}
```

4 Number Theory

4.1 basic

```cpp
template<typename T>
void gcd(const T &a,const T &b,T &d,T &x,T &y){
  if(!b) d=a,x=1,y=0;
  else gcd(b,a%b,d,y,x), y-=x*(a/b);
}
long long int phi[N+1];
void phiTable(){
  for(int i=1;i<=N;i++)phi[i]=i;
  for(int i=1;i<=N;i++)for(x=i*2;x<=N;x+=i)phi[x]-=phi[i];
}
void all_divdown(const LL &n) {// all n/x
  for(LL a=1;a<=n;a=n/(n/(a+1))){
    // dosomething;
  }
}
const int MAXPRIME = 1000000;
int iscom[MAXPRIME], prime[MAXPRIME], primecnt;
int phi[MAXPRIME], mu[MAXPRIME];
void sieve(void){
  memset(iscom,0,sizeof(iscom));
  primecnt = 0;
  phi[1] = mu[1] = 1;
  for(int i=2;i<MAXPRIME;++i) {
    if(!iscom[i]) {
      prime[primecnt++] = i;
      mu[i] = -1;
      phi[i] = i-1;
    }
    for(int j=0;j<primecnt;++j) {
      int k = i * prime[j];
      if(k>=MAXPRIME) break;
      iscom[k] = prime[j];
      if(i%prime[j]==0) {
        mu[k] = 0;
        phi[k] = phi[i] * prime[j];
```

```
        break;
      } else {
        mu[k] = -mu[i];
        phi[k] = phi[i] * (prime[j]-1);
      }
    }
  }
}

bool g_test(const LL &g, const LL &p,
    const vector<LL> &v) {
  for(int i=0;i<v.size();++i)
    if(modexp(g,(p-1)/v[i],p)==1)
      return false;
  return true;
}
LL primitive_root(const LL &p) {
  if(p==2) return 1;
  vector<LL> v;
  Factor(p-1,v);
  v.erase(unique(v.begin(), v.end()), v.
      end());
  for(LL g=2;g<p;++g)
    if(g_test(g,p,v))
      return g;
  puts("primitive_root NOT FOUND");
  return -1;
}
int Legendre(const LL &a, const LL &p) {
    return modexp(a%p,(p-1)/2,p); }

LL inv(const LL &a, const LL &n) {
  LL d,x,y;
  gcd(a,n,d,x,y);
  return d==1 ? (x+n)%n : -1;
}

int inv[maxN];
LL invtable(int n,LL P){
  inv[1]=1;
  for(int i=2;i<n;++i)
    inv[i]=(P-(P/i))*inv[P%i]%P;
}

LL log_mod(const LL &a, const LL &b,
    const LL &p) {
  // a ^ x = b ( mod p )
  int m=sqrt(p+.5), e=1;
  LL v=inv(modexp(a,m,p), p);
  map<LL,int> x;
  x[1]=0;
  for(int i=1;i<m;++i) {
    e = LLmul(e,a,p);
    if(!x.count(e)) x[e] = i;
  }
  for(int i=0;i<m;++i) {
    if(x.count(b)) return i*m + x[b];
    b = LLmul(b,v,p);
  }
  return -1;
}

LL Tonelli_Shanks(const LL &n, const LL &
    p) {
  // x^2 = n ( mod p )
  if(n==0) return 0;
  if(Legendre(n,p)!=1) while(1) { puts("
      SQRT ROOT does not exist"); }
  int S = 0;
  LL Q = p-1;
  while( !(Q&1) ) { Q>>=1; ++S; }
  if(S==1) return modexp(n%p,(p+1)/4,p);
  LL z = 2;
  for(;Legendre(z,p)!=-1;++z)
  LL c = modexp(z,Q,p);
  LL R = modexp(n%p,(Q+1)/2,p), t =
      modexp(n%p,Q,p);
  int M = S;
  while(1) {
    if(t==1) return R;
    LL b = modexp(c,1L<<(M-i-1),p);
    R = LLmul(R,b,p);
    t = LLmul( LLmul(b,b,p), t, p);
    c = LLmul(b,b,p);
    M = i;
  }
  return -1;
}

template<typename T>
T Euler(T n){
  T ans=n;
  for(T i=2;i*i<=n;++i){
    if(n%i==0){
      ans=ans/i*(i-1);
      while(n%i==0)n/=i;
    }
  }
  if(n>1)ans=ans/n*(n-1);
  return ans;
}

//Chinese_remainder_theorem
template<typename T>
T pow_mod(T n,T k,T m){
  T ans=1;
  for(n=(n>=m?n%m:n);k;k>>=1){
    if(k&1)ans=ans*n%m;
    n=n*n%m;
  }
  return ans;
}
template<typename T>
T crt(vector<T> &m,vector<T> &a){
  T M=1,tM,ans=0;
  for(int i=0;i<(int)m.size();++i)M*=m[i
      ];
  for(int i=0;i<(int)a.size();++i){
    tM=M/m[i];
    ans=(ans+(a[i]*tM%M)*pow_mod(tM,Euler
        (m[i])-1,m[i])%M)%M;
    /*如果m[i]是質數，Euler(m[i])-1=m[i
        ]-2，就不用算Euler了*/
  }
  return ans;
}

//java code
//求sqrt(N)的連分數
public static void Pell(int n){
  BigInteger N,p1,p2,q1,q2,a0,a1,a2,g1,g2
      ,h1,h2,p,q;
  g1=q2=p1=BigInteger.ZERO;
  h1=q1=p2=BigInteger.ONE;
  a0=a1=BigInteger.valueOf((int)Math.sqrt
      (1.0*n));
  BigInteger ans=a0.multiply(a0);
  if(ans.equals(BigInteger.valueOf(n))){
    System.out.println("No solution!");
    return ;
  }
  while(true){
    g2=a1.multiply(h1).substract(g1);
    h2=N.substract(g2.pow(2)).divide(h1);
    a2=g2.add(a0).divide(h2);
    p=a1.multiply(p2).add(p1);
    q=a1.multiply(q2).add(q1);
    if(p.pow(2).substract(N.multiply(q.
        pow(2)))).compareTo(BigInteger.
        ONE)==0)break;
    g1=g2;h1=h2;a1=a2;
    p1=p2;p2=p;
    q1=q2;q2=q;
  }
  System.out.println(p+" "+q);
}
```

4.2 bit set

```
void sub_set(int S){
  int sub=S;
  do{
    //對某集合的子集合的處理
    sub=(sub-1)&S;
  }while(sub!=S);
}
void k_sub_set(int k,int n){
  int comb=(1<<k)-1,S=1<<n;
  while(comb<S){
    //對大小為k的子集合的處理
    int x=comb&-comb,y=comb+x;
    comb=((comb&~y)/x>>1)|y;
  }
}
```

4.3 matrix exponential

```

void exp(LL m[2][2], LL x){
    LL c[2][2] = {{1,1},{1,0}},n[2][2];
    n[0][0] = m[0][0]*c[0][0] + m[0][1]*c
        [1][0];
    n[0][1] = m[0][0]*c[0][1] + m[0][1]*c
        [1][1];
    n[1][0] = m[1][0]*c[0][0] + m[1][1]*c
        [1][0];
    n[1][1] = m[1][0]*c[0][1] + m[1][1]*c
        [1][1];
    if(x != 1)
        exp(n,x-1);
    else
        cout << n[0][0];
}
int main(){
    LL u[2][2] = {{1,1},{1,0}},n;
    cin >> n;
    cout << "¶O¤@Ó¦C²Õ" << n+2 << "¶µ¬°";
    exp(u,n);
}
```

4.4 SpeedExpo

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

4.5 外星模運算

```
//a[0]^(a[1]^a[2]^...)
#define maxn 1000000
int euler[maxn+5];
bool is_prime[maxn+5];
void init_euler(){
  is_prime[1]=1;//一不是質數
  for(int i=1;i<=maxn;i++)euler[i]=i;
  for(int i=2;i<=maxn;i++){
    if(!is_prime[i]){//是質數
      euler[i]--;
      for(int j=i<<1;j<=maxn;j+=i){
        is_prime[j]=1;
        euler[j]=euler[j]/i*(i-1);
      }
    }
  }
}
LL pow(LL a,LL b,LL mod){//a^b%mod
  LL ans=1;
  for(;b;a=a*a%mod,b>>=1)
    if(b&1)ans=ans*a%mod;
  return ans;
}
bool isless(LL *a,int n,int k){
  if(*a==1)return k>1;
  if(--n==0)return *a<k;
  int next=0;
  for(LL b=1;b<k;++next)
    b*=*a;
  return isless(a+1,n,next);
}
LL high_pow(LL *a,int n,LL mod){
  if(*a==1||--n==0)return *a%mod;
  int k=0,r=euler[mod];
  for(LL tma=1;tma!=pow(*a,k+r,mod);++k)
    tma=tma*(*a)%mod;
  if(isless(a+1,n,k))return pow(*a,
      high_pow(a+1,n,k),mod);
  int tmd=high_pow(a+1,n,r), t=(tmd-k+r)%
      r;
  return pow(*a,k+t,mod);
}
LL a[1000005];
int t,mod;
int main(){
  init_euler();
  scanf("%d",&t);
  #define n 4
  while(t--){
    for(int i=0;i<n;++i)scanf("%lld",&a[i
        ]);
    scanf("%d",&mod);
    printf("%lld\n",high_pow(a,n,mod));
  }
  return 0;
}
```

4.6 大數取模

```
LL exp(LL x,LL y,LL p){
    if(y == 0) return 1;
    if(y & 1) return (exp(x,y-1,p)*x) % p
        ;//y is odd
    else{
        LL temp = exp(x,y/2,p);
        return (temp*temp) % p;
    }
}
LL calcmod(LL index,LL p){
    if(index == 0) return base[index]-'0'
        ;
    LL single = calcmod(index-1,p)*10;
    return (single%p + base[index]-'0')%p
        ;
}
```

4.7 模逆元

```
#include<bits/stdc++.h>
#define good ios_base::sync_with_stdio(0)
    ;cin.tie(0)
typedef long long LL;
using namespace std;

LL mod_inverse_by_speed_exp(LL x,LL y,LL
    p){
    if(y == 0) return 1;
    if(y % 2) return (
        mod_inverse_by_speed_exp(x,y-1,p
        )*x)%p;
    else{
        LL temp =
            mod_inverse_by_speed_exp(x,y
            /2,p);
        return (temp*temp)%p;
    }
}
int main(){
    good;
    LL n,i,p,x;
    cin >> n >> p;
    for(i = 0; i < n; i++){
        cin >> x;
        cout << mod_inverse_by_speed_exp(
            x,p-2,p) <<' ';
    }
    return 0;
}
```

4.8 質因數分解

```
LL func(const LL n,const LL mod,const int
    c) {
  return (LLmul(n,n,mod)+c+mod)%mod;
}

LL pollorrho(const LL n, const int c) {//
    循環節長度
  LL a=1, b=1;
  a=func(a,n,c)%n;
  b=func(b,n,c)%n; b=func(b,n,c)%n;
  while(gcd(abs(a-b),n)==1) {
    a=func(a,n,c)%n;
    b=func(b,n,c)%n; b=func(b,n,c)%n;
  }
  return gcd(abs(a-b),n);
}

void prefactor(LL &n, vector<LL> &v) {
  for(int i=0;i<12;++i) {
    while(n%prime[i]==0) {
      v.push_back(prime[i]);
      n/=prime[i];
    }
  }
}

void smallfactor(LL n, vector<LL> &v) {
  if(n<MAXPRIME) {
    while(isp[(int)n]) {
      v.push_back(isp[(int)n]);
      n/=isp[(int)n];
    }
    v.push_back(n);
  } else {
    for(int i=0;i<primecnt&&prime[i]*
        prime[i]<=n;++i) {
      while(n%prime[i]==0) {
        v.push_back(prime[i]);
        n/=prime[i];
      }
    }
    if(n!=1) v.push_back(n);
  }
}

void comfactor(const LL &n, vector<LL> &v
    ) {
  if(n<1e9) {
    smallfactor(n,v);
    return;
  }
  if(Isprime(n)) {
    v.push_back(n);
    return;
  }
  LL d;
  for(int c=3;;++c) {
    d = pollorrho(n,c);
    if(d!=n) break;
  }
  comfactor(d,v);
  comfactor(n/d,v);
}

void Factor(const LL &x, vector<LL> &v) {
  LL n = x;
  if(n==1) { puts("Factor 1"); return; }
  prefactor(n,v);
  if(n==1) return;
  comfactor(n,v);
  sort(v.begin(),v.end());
}

void AllFactor(const LL &n,vector<LL> &v)
    {
  vector<LL> tmp;
  Factor(n,tmp);
  v.clear();
  v.push_back(1);
  int len;
  LL now=1;
  for(int i=0;i<tmp.size();++i) {
    if(i==0 || tmp[i]!=tmp[i-1]) {
      len = v.size();
      now = 1;
    }
    now*=tmp[i];
    for(int j=0;j<len;++j)
      v.push_back(v[j]*now);
  }
}
```

5 String

5.1 manacher(最小回文字串)

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

6 Tree Problem

6.1 findLCA

```
LL findLCA(LL u,LL v){
    while(depth[u] > depth[v])
        u = father[u];
    while(depth[v] > depth[u])
        v = father[v];
    while(u != v){
        u = father[u];
        v = father[v];
    }
    return u;//or return v
}
```

6.2 kruskal(MST)

```
#include<bits/stdc++.h>
#define good ios_base::sync_with_stdio(0)
    ;cin.tie(0)
#define RSIZE 10002
#define pll pair<LL,LL>
typedef long long LL;
using namespace std;

struct EDGE{
    LL u,v,w;
};

vector<EDGE> adjacent;//out neighbor,
    weight
LL fa[RSIZE];

bool cmp(EDGE &a,EDGE &b){//sort by
    weight
    return a.w < b.w;
}
LL sfind(LL now){//find root,root's
    father=set size
    if(fa[now] < 0)
        return now;
    return fa[now] = sfind(fa[now]);
}
bool merge(LL u,LL v){//find two root,
    comparing size(by root's father)
    LL set1 = sfind(u),set2 = sfind(v);
    if(set1 == set2) return false;//same
        root-> no need to merge
    if(fa[set1] < fa[set2]){ //set1 is
        larger
        fa[set1] += fa[set2];
        fa[set2] = set1;
    }
    else{
        fa[set2] += fa[set1];
        fa[set1] = set2;
    }
    return true;
}

int main(){
    good;
    //freopen("file name", "r", stdin);
        //input redirection
    LL i,n,m;
    cin >> n >> m;
    for(i = 0; i < m; i++){
        LL x,y,weight;
        cin >> x >> y >> weight;
        adjacent.push_back({x,y,weight});
    }
    memset(fa,-1,sizeof(fa));//unvisited
    sort(adjacent.begin(),adjacent.end(),
        cmp);//sort by weight
    LL cost = 0,now_edge = 0;
    for(EDGE e : adjacent){
        if(merge(e.u,e.v)){//connect edge
            cost += e.w;
            now_edge++;
        }
    }
    if(now_edge < n-1)//not a MST
        cout << -1 << endl;
    else
        cout << cost << endl;
    return 0;
}
```

6.3 LCA

```
const int MAXN=100000; // 1-base
const int MLG=17; //log2(MAXN)+1;
int pa[MLG+2][MAXN+5];
int dep[MAXN+5];
vector<int> G[MAXN+5];
void dfs(int x,int p=0){//dfs(root);
  pa[0][x]=p;
  for(int i=0;i<=MLG;++i)
    pa[i+1][x]=pa[i][pa[i][x]];
  for(auto &i:G[x]){
    if(i==p)continue;
    dep[i]=dep[x]+1;
    dfs(i,x);
  }
}
inline int jump(int x,int d){
  for(int i=0;i<=MLG;++i)
    if((d>>i)&1) x=pa[i][x];
```

```
  return x;
}
inline int find_lca(int a,int b){
  if(dep[a]>dep[b])swap(a,b);
  b=jump(b,dep[b]-dep[a]);
  if(a==b)return a;
  for(int i=MLG;i>=0;--i){
    if(pa[i][a]!=pa[i][b]){
      a=pa[i][a];
      b=pa[i][b];
    }
  }
  return pa[0][a];
}
```

6.4 Prim(MST)

```
#include<bits/stdc++.h>
#define good ios_base::sync_with_stdio(0)
    ;cin.tie(0)
#define RSIZE 502
#define oo 1000000001 //1e9+1
typedef long long LL;
using namespace std;

vector<pair<LL,LL>> adjacent[RSIZE];//out
     neighbor, weight of edge
LL dis[RSIZE],fa[RSIZE];//dis for weight
    of two vertexes
bool visit[RSIZE] = {false};

int main(){
    good;
    //freopen("file name", "r", stdin);
        //input redirection
    LL i,n,m;
    cin >> n >> m;
    for(i = 0; i < m; i++){
        LL x,y,w;
        cin >> x >> y >> w;
        adjacent[x].push_back({y,w});
        adjacent[y].push_back({x,w});
    }
    //initial
    LL start = 0;
    memset(dis,oo,sizeof(dis));
    memset(fa,-1,sizeof(fa));
    priority_queue<pair<LL,LL>> PQ;//dis
        [],vertex
    PQ.push({dis[start] = 0,start});
    //prim
    while (!PQ.empty()){
        auto pt = PQ.top();
        PQ.pop();
        LL v = pt.second;
        if(visit[v]) continue;
        visit[v] = true;
        for(auto it : adjacent[v]){
            LL neibor = it.first,w = it.
                second;
            if(visit[neibor]) continue;
            if(w < dis[neibor]){ //new
                edge is shorter
                dis[neibor] = w;
                fa[neibor] = v;
                PQ.push({-dis[neibor],
                    neibor});
            }
        }
    }
    LL cost = 0,cnt = 0;
    //count cost and check if MST exists
    for(i = 0; i < n; i++){
        if(dis[i] < oo)
            cost += dis[i];
        else
            cnt++;
    }
    if(cnt)
        cout << -1 << endl;
    else
        cout << cost << endl;
    return 0;
}
```

6.5 segment tree

```
#include<bits/stdc++.h>
#define good ios_base::sync_with_stdio(0)
    ;cin.tie(0)
#define RSIZE 100000
#define pll pair<LL,LL>
#define lc 2*index //c means child
#define rc 2*index+1
typedef long long LL;
using namespace std;

LL tree[4*RSIZE];//saving range maximum
LL lazy[4*RSIZE] = {0};
LL num[RSIZE],cnt = 1;

//using range maximum as example
void build(LL L,LL R,LL index){
    LL temp = index;
    if(L == R){
        tree[index] = num[cnt];
        cnt++;
        return;
    }
    LL M = (L+R)/2;
    build(L,M,lc);
    build(M+1,R,rc);
    tree[index] = max(tree[lc],tree[rc]);
}
//single point modify
void modify(LL x,LL v,LL L,LL R,LL index)
    {
    if(L == R){
        tree[index] = v;
        return;
    }
    LL M = (L+R)/2;
    if(x <= M)//left side
        modify(x,v,L,M,lc);
    else
        modify(x,v,M+1,R,rc);
    tree[index] = max(tree[lc],tree[rc]);
}
//a range including index has to add tag
void addtag(LL tag,LL index){
    tree[index] += tag;
    lazy[index] += tag;
}
//transfering tag to child
void push(LL index){
    addtag(lazy[index],lc);
    addtag(lazy[index],rc);
    lazy[index] = 0;//tag is transfered
        to child
}
//Lower variables are queried range,UPPER
     ones are full range
LL query(LL l,LL r,LL L,LL R,LL index){
    if(l <= L && R <= r) return tree[
        index];
    push(index);//if use single point
        modify,no need
    LL M = (L+R)/2;
    if(r <= M) //answer in the left side,
        don't need to query right side
        return query(l,r,L,M,lc);
    else if(l > M) //in right side
        return query(l,r,M+1,R,rc);
    else //answer cross both side
        return max(query(l,r,L,M,lc),
            query(l,r,M+1,R,rc));//
            choose better one
}
void multi_modify(LL l,LL r,LL v,LL L,LL
    R,LL index){
    if(l <= L && R <= r){
        addtag(v,index);
        return;
    }
    push(index);
    LL M = (L+R)/2;
    if(r <= M) multi_modify(l,r,v,L,M,lc)
        ;
    else if(l > M) multi_modify(l,r,v,M
        +1,R,rc);
    else{
        multi_modify(l,r,v,L,M,lc);
        multi_modify(l,r,v,M+1,R,rc);
    }
    tree[index] = max(tree[lc],tree[rc]);
}

int main(){
    good;
    LL k,n;
    //build(1,n,1)at first,can use query(
        l,r,1,n,1).
    return 0;
}
```

7 default

7.1 8 queen

```
LL nqueen(LL n){
    int p[17],total = 0;
    for(int i = 0; i < n; i++)
        p[i] = i;
    do{
        bool valid = true;
        for(int i = 0; i < n; i++){
            for(int j = i+1; j < n; j++){
                if(abs(p[i]-p[j]) == j-i)
                    {//same diagonal
                    valid = false;
                    break;
                }
            }
        }
        if(valid) total++;
    } while (next_permutation(p,p+n));
    return total;
}
```

7.2 debug

```
#ifdef DEBUG
#define dbg(...) {\
  fprintf(stderr,"%s - %d : (%s) = ",
      __PRETTY_FUNCTION__,__LINE__,#
      __VA_ARGS__);\
  _DO(__VA_ARGS__);\
}
template<typename I> void _DO(I&&x){cerr
    <<x<<endl;}
template<typename I,typename...T> void
    _DO(I&&x,T&&...tail){cerr<<x<<", ";
    _DO(tail...);}
#else
#define dbg(...)
#endif
```

7.3 IncStack

```
//Magic
#pragma GCC optimize "Ofast"
//stack resize,change esp to rsp if 64-
    bit system
asm("mov %0,%%esp\n" ::"g"(mem+10000000))
    ;
-Wl,--stack,214748364 -trigraphs
#pragma comment(linker, "/STACK
    :1024000000,1024000000")
//linux stack resize
#include<sys/resource.h>
void increase_stack(){
  const rlim_t ks=64*1024*1024;
  struct rlimit rl;
  int res=getrlimit(RLIMIT_STACK,&rl);
  if(!res&&rl.rlim_cur<ks){
    rl.rlim_cur=ks;
    res=setrlimit(RLIMIT_STACK,&rl);
  }
}
```

7.4 input

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

7.5 randomize

```
map<LL,LL> discret;
for(i = 0; i < n; i++){
        cin >> a[i];
        discret[a[i]] = 0;
    }
    LL index = 0;
    for(auto &it : discret)
        it.second = index++;
```

7.6 sweepline

```
#include<bits/stdc++.h>
#define good ios_base::sync_with_stdio(0)
    ;cin.tie(0)
typedef long long LL;
using namespace std;

struct Seg{
    LL left,right;
};
bool cmp(Seg &a,Seg &b){
    return a.left < b.left;
}

int main(){
    good;
    LL n;
    cin >> n;
    Seg line[n];
    for(LL i = 0; i < n; i++)
        cin >> line[i].left >> line[i].
            right;
    sort(line,line+n,cmp);
    Seg last = line[0];
    LL total = 0;
    for(LL i = 1; i < n; i++){
        if(line[i].left > last.right){
            total += last.right - last.
                left;
            last = line[i];
            continue;
        }
        last.right = max(last.right,line[
            i].right);//merge last and
            line[i]
    }
    total += last.right - last.left;
    cout << total;
    return 0;
}
```

7.7 模板

```
#include<bits/stdc++.h>
#define good ios_base::sync_with_stdio(0)
    ;cin.tie(0)
#define RSIZE 101
#define oo 1000000001//1e9+1
#define pll pair<LL,LL>
#define lc 2*index
#define rc 2*index+1
typedef long long LL;
using namespace std;

int main(){
    good;

    return 0;
}
```

8 other

8.1 1D0D dp

```
#include<bits/stdc++.h>
#define good ios_base::sync_with_stdio(0)
    ;cin.tie(0)
typedef long long LL;
using namespace std;

int main(){
    good;
    //freopen("file name", "r", stdin);
        //input redirection
    LL i,n;
    LL money[100002],dp[100002] ={0};//dp
        for last i day max profit
    cin >> n;
    for(i = 1; i <= n; i++)
        cin >> money[i];
    dp[0] = 0,dp[1] = money[1],dp[2] =
        max(money[1],money[2]);
    for(i = 3; i <= n; i++)
        dp[i] = max(dp[i-2]+money[i],dp[i
            -1]);//choose,discard
    cout << dp[i-1];
    return 0;
}
```

8.2 mergesort

```
#include <bits/stdc++.h>
using namespace std;
#define good ios_base::sync_with_stdio(0)
    ;cin.tie(0)
#define NORMALSIZE 8
typedef long long LL;

void merge(LL *,LL,LL,LL);//prototype
void mergesort(LL *,LL,LL);

void mergesort(LL* arr,LL l,LL r){
    LL m;
    if(l < r){
        m = (l+r)/2;
        mergesort(arr,l,m);//sort left
            subarray
        mergesort(arr,m+1,r);//sort right
            sunarray
        merge(arr,l,m,r);
    }
    return;
}
void merge(LL* arr,LL l,LL m,LL r){
    LL left,right,tmp[NORMALSIZE],i;
    left = l; // left subarr.begin
    right = m+1; //right subarr.begin
    i = l;
    while ((left <= m) && (right <= r)){
        //merging left subarr and right
        subarr
        if(arr[left] < arr[right]){ //
            left subarr is smaller
            tmp[i] = arr[left];
            i++,left++;
        }
        else{ //the other situation
            tmp[i] = arr[right];
            i++,right++;
        }
    }
    while(left <= m){ // right subarr is
        fully sorted
        tmp[i] = arr[left];
        i++,left++;
    }
    while(right <= r){ // left subarr is
        fully sorted
        tmp[i] = arr[right];
        i++,right++;
    }
    for(i = l; i <= r; i++)
        arr[i] = tmp[i];
}
```

8.3 WhatDay

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

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

1. 對於平面圖，$F = E - V + C + 1$，C 是連通分量數
2. 對於平面圖，$E \leq 3V - 6$
3. 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：
   (a) $I(G) + Cv(G) = |V|$
   (b) $M(G) + Ce(G) = |V|$
4. 對於連通二分圖：
   (a) $I(G) = Cv(G)$
   (b) $M(G) = Ce(G)$
5. 最大權閉合圖：
   (a) $C(u,v) = \infty, (u,v) \in E$
   (b) $C(S,v) = W_v, W_v > 0$
   (c) $C(v,T) = -W_v, W_v < 0$
   (d) ans=$\sum_{W_v>0} W_v - flow(S,T)$
6. 最大密度子圖：
   (a) 求 $max\left(\frac{W_e+W_v}{|V'|}\right), e \in E', v \in V'$
   (b) $U = \sum_{v\in V} 2W_v + \sum_{e\in E} W_e$
   (c) $C(u,v) = W_{(u,v)}, (u,v) \in E$，雙向邊
   (d) $C(S,v) = U, v \in V$
   (e) $D_u = \sum_{(u,v)\in E} W_{(u,v)}$
   (f) $C(v,T) = U + 2g - D_v - 2W_v, v \in V$
   (g) 二分搜 g：
   $l = 0, r = U, eps = 1/n^2$
   if$((U \times |V| - flow(S,T))/2 > 0)\ l = mid$
   else $r = mid$
   (h) ans=$min\_cut(S,T)$
   (i) $|E| = 0$ 要特殊判斷
7. 弦圖：
   (a) 點數大於 3 的環都要有一條弦
   (b) 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
   (c) 最大團大小 = 色數
   (d) 最大獨立集: 完美消除序列從前往後能選就選
   (e) 最小團覆蓋: 最大獨立集的點和他延伸的邊構成
   (f) 區間圖是弦圖
   (g) 區間圖的完美消除序列: 將區間按造又端點由小到大排序
   (h) 區間圖染色: 用線段樹做

9.1.3 dinic 特殊圖複雜度

1. 單位流：$O\left(min\left(V^{3/2}, E^{1/2}\right)E\right)$
2. 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 $max\left(\frac{\sum B_i x_i}{\sum C_i x_i}\right)$

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

```
binary_search(){
  while(r-l>eps){
    g=(l+r)/2;
    for(i:所有元素)D[i]=B[i]-g*C[i];//D(i
        ,g)
    找出一組合法x[i]使f(g)最大;
    if(f(g)>0) l=g;
    else r=g;
  }
  Ans = r;
}
Dinkelbach(){
  g=任意狀態(通常設為0);
  do{
    Ans=g;
    for(i:所有元素)D[i]=B[i]-g*C[i];//D(i
        ,g)
    找出一組合法x[i]使f(g)最大;
    p=0,q=0;
    for(i:所有元素)
      if(x[i])p+=B[i],q+=C[i];
    g=p/q;//更新解，注意q=0的情況
  }while(abs(Ans-g)>EPS);
  return Ans;
}
```

9.1.5 學長公式

1. $\sum_{d|n}\phi(n)=n$
2. $g(n)=\sum_{d|n}f(d) => f(n)=\sum_{d|n}\mu(d)\times g(n/d)$
3. Harmonic series $H_n=\ln(n)+\gamma+1/(2n)-1/(12n^2)+1/(120n^4)$
4. $\gamma=0.57721566490153286060651209008240243104215$
5. 格雷碼 $=n\oplus(n>>1)$
6. $SG(A+B)=SG(A)\oplus SG(B)$
7. 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

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

9.1.7 排組公式

1. k 卡特蘭 $\frac{C_n^{kn}}{n(k-1)+1}\cdot C_m^n=\frac{n!}{m!(n-m)!}$
2. $H(n,m)\cong x_1+x_2\ldots+x_n=k, num=C_k^{n+k-1}$
3. Stirling number of 2^{nd},n 人分 k 組方法數目
   (a) $S(0,0)=S(n,n)=1$
   (b) $S(n,0)=0$
   (c) $S(n,k)=kS(n-1,k)+S(n-1,k-1)$
4. Bell number,n 人分任意多組方法數目
   (a) $B_0=1$
   (b) $B_n=\sum_{i=0}^{n}S(n,i)$
   (c) $B_{n+1}=\sum_{k=0}^{n}C_k^nB_k$
   (d) $B_{p+n}\equiv B_n+B_{n+1}modp$, p is prime
   (e) $B_{p^m+n}\equiv mB_n+B_{n+1}modp$, p is prime
   (f) From B_0 : 1, 1, 2, 5, 15, 52, 203, 877, 4140, 21147, 115975
5. Derangement, 錯排, 沒有人在自己位置上
   (a) $D_n=n!(1-\frac{1}{1!}+\frac{1}{2!}-\frac{1}{3!}\ldots+(-1)^n\frac{1}{n!})$
   (b) $D_n=(n-1)(D_{n-1}+D_{n-2}), D_0=1, D_1=0$
   (c) From D_0 : 1, 0, 1, 2, 9, 44, 265, 1854, 14833, 133496
6. Binomial Equality
   (a) $\sum_k\binom{r}{m+k}\binom{s}{n-k}=\binom{r+s}{m+n}$
   (b) $\sum_k\binom{l}{m+k}\binom{s}{n+k}=\binom{l+s}{l-m+n}$
   (c) $\sum_k\binom{l}{m+k}\binom{s+k}{n}(-1)^k=(-1)^{l+m}\binom{s-m}{n-l}$
   (d) $\sum_{k\le l}\binom{l-k}{m}\binom{s}{k-n}(-1)^k=(-1)^{l+m}\binom{s-m-1}{l-n-m}$
   (e) $\sum_{0\le k\le l}\binom{l-k}{m}\binom{q+k}{n}=\binom{l+q+1}{m+n+1}$
   (f) $\binom{r}{k}=(-1)^k\binom{k-r-1}{k}$
   (g) $\binom{r}{m}\binom{m}{k}=\binom{r}{k}\binom{r-k}{m-k}$
   (h) $\sum_{k\le n}\binom{r+k}{k}=\binom{r+n+1}{n}$
   (i) $\sum_{0\le k\le n}\binom{k}{m}=\binom{n+1}{m+1}$
   (j) $\sum_{k\le m}\binom{m+r}{k}x^ky^k=\sum_{k\le m}\binom{-r}{k}(-x)^k(x+y)^{m-k}$

9.1.8 冪次, 冪次和

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

9.1.9 Burnside's lemma

1. $|X/G|=\frac{1}{|G|}\sum_{g\in G}|X^g|$
2. $X^g=t^{c(g)}$
3. 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。
4. 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 $\frac{1}{24}(3^6+6\times3^3+3\times3^4+8\times3^2+6\times3^3)=57$

9.1.10 Count on a tree

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

Codebook - ss

Contents

1 Computational Geometry **1**
- 1.1 SmallestCircle 1
- 1.2 最近點對 1

2 Data Structure **1**
- 2.1 01 背包 1
- 2.2 array simulate linked list . 1
- 2.3 binary search 1
- 2.4 discretization 1
- 2.5 half enumeration 1
- 2.6 LCS 2
- 2.7 LIS 2
- 2.8 skew heap 2
- 2.9 sliding window 2
- 2.10 undo disjoint set 2

3 Graph **2**
- 3.1 bellman ford 2
- 3.2 BFS 2
- 3.3 DFS 2
- 3.4 dijkstra 2
- 3.5 floyd warshall 3
- 3.6 topology sort 3
- 3.7 union and find 3

4 Number Theory **3**
- 4.1 basic 3
- 4.2 bit set 4
- 4.3 matrix exponential 4
- 4.4 SpeedExpo 4
- 4.5 外星模運算 4
- 4.6 大數取模 4
- 4.7 模逆元 5
- 4.8 質因數分解 5

5 String **5**
- 5.1 manacher(最小回文字串) 5

6 Tree Problem **5**
- 6.1 findLCA 5
- 6.2 kruskal(MST) 5
- 6.3 LCA 5
- 6.4 Prim(MST) 6
- 6.5 segment tree 6

7 default **6**
- 7.1 8 queen 6
- 7.2 debug 6
- 7.3 IncStack 6
- 7.4 input 6
- 7.5 randomize 7
- 7.6 sweepline 7
- 7.7 模板 7

8 other **7**
- 8.1 1D0D dp 7
- 8.2 mergesort 7
- 8.3 WhatDay 7

9 zformula **7**
- 9.1 formula 7
  - 9.1.1 Pick 公式 7
  - 9.1.2 圖論 7
  - 9.1.3 dinic 特殊圖複雜度 7
  - 9.1.4 0-1 分數規劃 . . . 7
  - 9.1.5 學長公式 8
  - 9.1.6 基本數論 8
  - 9.1.7 排組公式 8
  - 9.1.8 冪次, 冪次和 . . . 8
  - 9.1.9 Burnside's lemma . 8
  - 9.1.10 Count on a tree . . 8

Codebook - ss

C++ Resource Test

```cpp
#include <bits/stdc++.h>
using namespace std;

namespace system_test {

const size_t KB = 1024;
const size_t MB = KB * 1024;
const size_t GB = MB * 1024;

size_t block_size, bound;
void stack_size_dfs(size_t depth = 1) {
  if (depth >= bound)
    return;
  int8_t ptr[block_size]; // 若無法編譯將 block_size 改成常數
  memset(ptr, 'a', block_size);
  cout << depth << endl;
  stack_size_dfs(depth + 1);
}

void stack_size_and_runtime_error(size_t block_size, size_t bound = 1024) {
  system_test::block_size = block_size;
  system_test::bound = bound;
  stack_size_dfs();
}

double speed(int iter_num) {
  const int block_size = 1024;
  volatile int A[block_size];
  auto begin = chrono::high_resolution_clock::now();
  while (iter_num--)
    for (int j = 0; j < block_size; ++j)
      A[j] += j;
  auto end = chrono::high_resolution_clock::now();
  chrono::duration<double> diff = end - begin;
  return diff.count();
}

void runtime_error_1() {
  // Segmentation fault
  int *ptr = nullptr;
  *(ptr + 7122) = 7122;
}

void runtime_error_2() {
  // Segmentation fault
  int *ptr = (int *)memset;
  *ptr = 7122;
}

void runtime_error_3() {
  // munmap_chunk(): invalid pointer
  int *ptr = (int *)memset;
  delete ptr;
}

void runtime_error_4() {
  // free(): invalid pointer
  int *ptr = new int[7122];
  ptr += 1;
  delete[] ptr;
}

void runtime_error_5() {
  // maybe illegal instruction
  int a = 7122, b = 0;
  cout << (a / b) << endl;
}

void runtime_error_6() {
  // floating point exception
  volatile int a = 7122, b = 0;
  cout << (a / b) << endl;
}

void runtime_error_7() {
  // call to abort.
  assert(false);
}

} // namespace system_test

#include <sys/resource.h>
void print_stack_limit() { // only work in Linux
  struct rlimit l;
  getrlimit(RLIMIT_STACK, &l);
  cout << "stack_size = " << l.rlim_cur << " byte" << endl;
}
```