

动态规划dp

数位dp

数据结构

树状数组[区间修改单点查询]

线段树[单点修改区间查询]

主席树

主席树前k小的和

RBtree

splay

treap比x大的数有多少个

trie思想建树

数学

埃筛

大素数判定+泼辣的肉

第几个质数

费马小定理

高精度

高精度除法

高斯-约旦消元

矩阵快速幂

扩展欧几里得

欧拉函数

欧拉筛

线性基

圆和矩形的面积交

Min25

Zeller Formula

图论

网络流

二分图最大流

Dinic (Node版本)

次小生成树

二分图匹配-匈牙利算法

dijkstra

LCA

LCA-倍增

LCA-tarjan

tarjan求割点

tarjan缩点

字符串

KMP

Manacher

最大字典序子串

最大最小表示法

DFS

DSU (树上启发式合并)

STL&杂项

二分 (标准)

优先队列

exmu

highbit

LIS (最长上升子序列)

Tarjan

对拍.bat

动态规划dp

数位dp

```
1  /*
2      LOJ 10163
3      ACWing, 1081 度的数量
4  * @ author: dragon_bra
5  * @ email: tommy514@foxmail.com
6  * @ date: 2021-03-10 16:31
7  */
8
9  #include <bits/stdc++.h>
10 #define fastio ios::sync_with_stdio(false); cin.tie(0);
11 using namespace std;
12
13 typedef long long ll;
14 const int N = 35 + 10;
15
16 void redirect() {
17 #ifdef LOCAL
18     freopen("in.txt", "r", stdin);
19     freopen("out.txt", "w", stdout);
20 #endif
21 }
22
23 int B, K;
24 int f[N][N] = {0};
25
26 void init() {
27     for (int i = 0; i < N; i++) {
28         for (int j = 0; j <= i; j++) {
29             if (j == 0)
30                 f[i][j] = 1;
31             else
32                 f[i][j] = f[i - 1][j - 1] + f[i - 1][j];
33         }
34     }
35 }
36
37 int dp(int n) {
38     if (!n)
39         return 0;
40
41     vector<int> nums;
42
43     while (n)
44         nums.push_back(n % B), n /= B;
45
46     int res = 0;
47     int last = 0; // 当前已有1的个数
48
49     for (int i = nums.size() - 1; i >= 0; i--) {
50         int x = nums[i];
51     }
```

```

52         if (x) {
53             res += f[i][K - last];
54
55             if (x > 1) {
56                 if (K - last - 1 >= 0)
57                     res += f[i][K - last - 1];
58
59                 break;
60             } else {
61                 last ++;
62
63                 if (last > K)
64                     break;
65             }
66         }
67
68         if (i == 0 && last == K)
69             res ++;
70     }
71
72     return res;
73 }
74
75 int main() {
76     redirect();
77     init();
78
79     int l, r;
80     cin >> l >> r >> K >> B;
81
82     cout << dp(r) - dp(l - 1) << endl;
83
84     return 0;
85 }

```

数据结构

树状数组[区间修改单点查询]

```

1  int n,m;
2  int a[50005] = {0},c[50005]; //对应原数组和树状数组
3
4  int lowbit(int x){
5      return x&(-x);
6  }
7
8  void updata(int i,int k){    //在i位置加上k
9      while(i <= n){
10         c[i] += k;
11         i += lowbit(i);
12     }
13 }
14
15 int getSum(int i){           //求D[1 - i]的和，即A[i]值

```

```

16     int res = 0;
17     while(i > 0){
18         res += c[i];
19         i -= lowbit(i);
20     }
21     return res;
22 }
23
24 int main(){
25     cin>>n;
26     for(int i = 1; i <= n; i++){
27         cin>>a[i];
28         updata(i,a[i] - a[i-1]);    //输入初值的时候，也相当于更新了值
29     }
30
31     // [x,y] 区间内加上 k
32     updata(x,k);    // A[x] - A[x-1] 增加 k
33     updata(y+1,-k);    // A[y+1] - A[y] 减少 k
34
35     // 查询 i 位置的值
36     int sum = getsum(i);
37
38     return 0;
39 }

```

线段树[单点修改区间查询]

```

1  #include <cstdio>
2  #include <iostream>
3  #include <algorithm>
4  #include <cmath>
5  #include <cstring>
6  #include <map>
7  #include <set>
8  #include <queue>
9  #include <string>
10 #include <vector>
11 using namespace std;
12 typedef long long ll;
13 typedef unsigned long long ull;
14 const int INF = 0x7fffffff;
15 const int mod = 1e9+7;
16 const double eps = 1e-5;
17 const int N = 1e5+10;
18
19 void redirect(){
20     #ifdef LOCAL
21         freopen("test.txt","r",stdin);
22     #endif
23 }
24 inline ll read(){
25     ll f=1,x=0;char ch;
26     do{ch=getchar();if(ch=='-')f=-1;}while(ch<'0' || ch>'9');
27     do{x=x*10+ch-'0';ch=getchar();}while(ch>='0' && ch<='9');
28     return x*f;
29 }
30
31 int n,k;

```

```

32  int pos[N];int a[N];
33
34  struct NOOD {
35      int l, r, add, Max;
36  }tree[N * 4 + 5];
37  void Build(int L, int R, int x) {
38      tree[x].l = L, tree[x].r = R, tree[x].Max = 0;
39      if(L == R) {
40          tree[x].Max = a[L];
41          return ;
42      }
43      int mid = (L + R) / 2;
44      Build(L, mid, x * 2);
45      Build(mid + 1, R, x * 2 + 1);
46      tree[x].Max = max(tree[x * 2].Max, tree[x * 2 + 1].Max);
47  }
48  void PushDown(int x) {
49      if(tree[x].add) {
50          tree[x * 2].Max = tree[x].add;
51          tree[x * 2 + 1].Max = tree[x].add;
52          tree[x * 2].add = tree[x].add;
53          tree[x * 2 + 1].add = tree[x].add;
54          tree[x].add = 0;
55      }
56  }
57  void Update(int L, int R, int add, int x) {
58      if(L <= tree[x].l && tree[x].r <= R) {
59          tree[x].add = add;
60          tree[x].Max = add;
61          return ;
62      }
63      PushDown(x);
64      int mid = (tree[x].l + tree[x].r) / 2;
65      if(L <= mid)Update(L, R, add, x * 2);
66      if(R > mid)Update(L, R, add, x * 2 + 1);
67      tree[x].Max = max(tree[x * 2].Max, tree[x * 2 + 1].Max);
68  }
69
70  int Query(int L, int R, int x) {
71      if(L <= tree[x].l && tree[x].r <= R)return tree[x].Max;
72      PushDown(x);
73      int mid = (tree[x].l + tree[x].r) / 2;
74      int res = 0;
75      if(L <= mid) res = max(res, Query(L, R, x * 2));
76      if(R > mid) res = max(res, Query(L, R, x * 2 + 1));
77      return res;
78  }
79
80  int nxt[N];int ans[N];
81
82  int dfs(int i){
83      if(nxt[i]==0||ans[i]!=1) return ans[i];
84      else return ans[i]=dfs(nxt[i])+1;
85  }
86
87  int main(){
88      redirect();
89      int T;scanf("%d",&T);

```

```

90     while(T--){
91         scanf("%d%d",&n,&k);
92         memset(nxt,0,sizeof(nxt));memset(tree, 0, sizeof(tree));
93         for(int i=1;i<=n;i++){
94             scanf("%d",&a[i]);pos[a[i]]=i;ans[i]=1;
95         }
96         Build(1, n, 1);
97         for(int i=n;i>=1;i--){
98             update(pos[i], pos[i] , 0, 1);
99             int big = Query(max(pos[i]-k,1), min(pos[i]+k,n), 1);
100             if(big!=0) nxt[i]=big;
101         }
102
103         for(int i=1;i<=n;i++){
104             int ans = dfs(i);printf("%d%c",ans,i==n?'\\n':' ');
105         }
106
107     }
108     return 0;
109 }
110
111 /*
112 ---linux compile---
113 g++ aa.cpp -o aa
114 ./ aa
115 -----
116 author:dragon_bra
117 */

```

主席树

```

1  #include<iostream>
2  #include<algorithm>
3  #include<cstdio>
4  #include<cstring>
5  using namespace std;
6  const int N = 200500;
7
8  void redirect() {
9      #ifdef LOCAL
10         freopen("in.txt","r",stdin);
11         freopen("out.txt","w",stdout);
12     #endif
13 }
14
15 struct node{
16     int l, r, sum;
17     #define l(x) tree[x].l
18     #define r(x) tree[x].r
19     #define sum(x) tree[x].sum
20 }tree[N<<5];
21
22 int n, m, a[N], b[N];
23 int q, cnt, t[N];
24 int build(int l, int r) {
25     int rt = ++cnt;
26     sum(rt) = 0;

```

```

27     int mid = (l + r) >> 1;
28     if (l < r) {
29         l(rt) = build(l, mid);
30         r(rt) = build(mid + 1, r);
31     }
32     return rt;
33 }
34 inline int update(int pre,int l,int r,int x) {
35     int rt = ++cnt;
36     l(rt) = l(pre), r(rt) = r(pre);
37     sum(rt) = sum(pre) + 1;
38     int mid = (l + r) >> 1;
39     if (l < r) {
40         if (x <= mid) l(rt) = update(l(pre), l, mid, x);
41         else r(rt) = update(r(pre), mid + 1, r, x);
42     }
43     return rt;
44 }
45 inline int query(int u,int v,int l,int r,int k) {
46     if (l >= r) return l;
47     int x = sum(l(v)) - sum(l(u));
48     int mid = (l + r) >> 1;
49     if (x >= k) return query(l(u), l(v), l, mid, k);
50     else return query(r(u), r(v), mid + 1, r, k - x);
51 }
52 int main() {
53     redirect();
54     cin >> n >> q;
55     for (int i = 1; i <= n; i++) {
56         cin >> a[i]; b[i] = a[i];
57     }
58     sort(b + 1, b + n + 1);
59     m = unique(b + 1, b + n + 1) - b - 1;
60
61     t[0] = build(1, m);
62     for (int i = 1; i <= n; i++) {
63         int T = lower_bound(b + 1, b + m + 1, a[i]) - b;
64         t[i] = update(t[i-1], 1, m, T);
65     }
66
67     while (q--) {
68         int l, r, k;
69         cin >> l >> r >> k;
70         printf ("%d\n", b[query(t[l-1], t[r], 1, m, k)]);
71     }
72     return 0;
73 }

```

主席树前k小的和

```

1  #include<bits/stdc++.h>
2  using namespace std;
3  const int MAXN=100010;
4  const int M=MAXN*30;
5  int n,q,m,tot;
6  int a[MAXN],t[MAXN];
7  int T[MAXN],lson[M],rson[M],c[M];

```

```

8 long long sum[M];
9 void Init_hash(){
10     for(int i=1;i<=n;i++){
11         t[i] = a[i];
12     }
13     sort(t+1,t+1+n);
14     m=unique(t+1,t+1+n)-t-1;
15 }
16 int build(int l,int r){
17     int root=tot++;
18     c[root]=0; sum[root] = 0;
19     if(l==r){
20         int mid=(l+r)>>1;
21         lson[root] = build(l,mid);
22         rson[root] = build(mid+1,r);
23     }
24     return root;
25 }
26 int Hash(int x){
27     return lower_bound(t+1,t+1+m,x)-t;
28 }
29 int update(int root,int pos, int val){
30     int newroot = tot++,tmp = newroot;
31     c[newroot] = c[root] + val;
32     sum[newroot] = sum[root] + t[pos];
33     int l=1,r=m;
34     while(l<r){
35         int mid = (l+r)>>1;
36         if(pos <= mid){
37             lson[newroot]= tot++; rson[newroot] = rson[root];
38             newroot = lson[newroot];root = lson[root];
39             r = mid;
40         }
41         else{
42             rson[newroot] = tot++; lson[newroot] = lson[root];
43             newroot = rson[newroot]; root = rson[root];
44             l = mid+1;
45         }
46         c[newroot] = c[root] + val;
47         sum[newroot] = sum[root] + t[pos];
48     }
49     return tmp;
50 }
51 int query(int left_root,int right_root,int k){
52     int l=1,r=m;
53     long long res = 0;
54     while( l < r ){
55         int mid = (l+r)>>1;
56         if(c[lson[left_root]]-c[lson[right_root]]>=k){
57             r = mid;
58             left_root = lson[left_root];
59             right_root = lson[right_root];
60         }
61         else{
62             l = mid + 1;
63             k -= c[lson[left_root]]-c[lson[right_root]];
64             res += sum[lson[left_root]] - sum[lson[right_root]];
65             left_root = rson[left_root];

```



```

66         right_root = rson[right_root];
67     }
68 }
69 return res;
70 }
71 int main(){
72     #ifdef LOCAL
73         freopen("in.txt","r",stdin);
74         freopen("out.txt","w",stdout);
75     #endif
76     while(scanf("%d%d",&n,&q) == 2){
77         tot = 0;
78         for(int i = 1; i <= n;i++){
79             scanf("%d",&a[i]);
80         }
81         Init_hash();
82         T[n+1] = build(1,m);
83         for(int i = n;i ;i--){
84             int pos = Hash(a[i]);
85             T[i] = update(T[i+1], pos ,1);
86         }
87         while(q--){
88             int l,r,k;
89             scanf("%d%d%d",&l,&r,&k);
90             k = (r-l+1 + 1) - k; // 第k小变成第k大
91             printf("%d\n",query(T[l],T[r+1],k));
92         }
93     }
94 }

```

RBtree

```

1  template<class T>
2  struct RBtree{
3      #define l _M_left
4      #define r _M_right
5      #define p _M_parent
6      #define node _Rb_tree_node_base
7  #if __cplusplus<=199711L
8      #define key _M_value_field.first
9      #define size _M_value_field.second
10 #else //c++11
11     #define key _M_storage._M_ptr()->first
12     #define size _M_storage._M_ptr()->second
13 #endif
14     typedef _Rb_tree_node<pair<const T,int> > Node; map<T,int> M;
15     void fix_size(node *it){
16         int &it_size=static_cast<Node*>(it)->size;it_size=1;
17         if (it->l)it_size+=static_cast<Node*>(it->l)->size;
18         if (it->r)it_size+=static_cast<Node*>(it->r)->size;
19     }
20     void fix_all(node *it,node *end){
21         for (;it=it->p){
22             if (it->l)fix_size(it->l);if (it->r)fix_size(it->r);
23             if (it->p==end){fix_size(it);break;}
24         }
25     }

```

```

26     void insert(const T &x){
27         pair<typename map<T,int>::iterator,bool>
it=M.insert(make_pair(x,0));
28         if (!it.second)return;
29         fix_all(it.first._M_node,M.end()._M_node);
30     }
31     int select(int k){
32         node *p=get_root();
33         while (k){
34             int size1=p->l?static_cast<Node*>(p->l)->size:0;
35             if (k==size1+1)break;
36             if (k<=size1)p=p->l;
37             else k-=size1+1,p=p->r;
38         }
39         return static_cast<Node*>(p)->key;
40     }
41     int rank(int x){
42         node *p=get_root(); int res=0;
43         while (p){
44             int y=static_cast<Node*>(p)->key;
45             int s=p->l?static_cast<Node*>(p->l)->size:0;
46             if (y<=x)res+=s+1,p=p->r;
47             else p=p->l;
48         }
49         return res;
50     }
51     node *get_root(){
52         node *it=M.begin()._M_node;
53         while (it->p!=M.end()._M_node)it=it->p;
54         return it;
55     }
56     void print(){print_node(get_root(),"");}
57     void print_node(const node *it,string str){
58         if (!it){cout<<str<<"nil (0)"<<endl;return;}
59         cout<<str<<static_cast<const Node*>(it)->key;
60         cout<<"("<<static_cast<const Node*>(it)->size<<")"<<endl;
61         print_node(it->l,str+"    "); print_node(it->r,str+"    ");
62     }
63     #undef l
64     #undef r
65     #undef p
66     #undef node
67     #undef key
68     #undef size
69 };
70 RBtree<int> a;

```

splay

```

1  #include <bits/stdc++.h>
2  using namespace std;
3  typedef long long ll;
4  const int N = 2e5+10;
5
6  struct node{
7      int data;
8  }_a[N];

```

```

9
10 bool operator < (node const &_a,node const &_b){
11     return _a.data<_b.data;
12 }
13 bool operator > (node const &_a,node const &_b){
14     return _a.data>_b.data;
15 }
16 bool operator == (node const &_a,node const &_b){
17     return _a.data<_b.data;
18 }
19 bool operator != (node const &_a,node const &_b){
20     return _a.data<_b.data;
21 }
22
23 int n,t,_root,_sz;
24 int _fa[N],_s[N][2],_cnt[N],_size[N];ll _sum[N];
25
26 inline int ws(int x){return _s[_fa[x]][1]==x;}//which son
27 void setson(int son,int f,int w){//0-left,C;1-right,◆◆;
28     if(son!=0) _fa[son]=f;
29     if(f!=0) _s[f][w]=son;
30 }
31 void maintain(int x){
32     _size[x]=_size[_s[x][0]]+_size[_s[x][1]] + _cnt[x];
33     _sum[x]=_sum[_s[x][0]] + _sum[_s[x][1]] + (ll)_cnt[x]*_a[x].data;
34 }
35 void rot(int x){
36     int f=_fa[x]; int ff=_fa[f]; int w=ws(x); int wf=ws(f);
37     int p=_s[x][!w];
38     setson(p,f,w);
39     setson(x,ff,wf);
40     setson(f,x,!w);//!w
41     maintain(f);
42     maintain(x);
43 }
44 void splay(int x){
45     for(;;_fa[x];rot(x)) if(_fa[_fa[x]]&&ws(_fa[x])==ws(x))
46         rot(_fa[x]);//zig-zag or zig-zig
47     _root=x;
48 }
49 void insert(int now,node p){
50     if(_root==0){
51         _root=++_sz;
52         _a[_sz]=p;
53         _size[_sz]=_cnt[_sz]=1;
54         return;
55     }
56     while(_a[now]!=p){
57         _size[now]++;
58         if(p>_a[now]){
59             if(_s[now][1]==0){
60                 _a[++_sz]=p;
61                 setson(_sz,now,1);
62             }
63             now=_s[now][1];
64         }
65         else{
66             if(_s[now][0]==0){

```

```

66         _a[++_sz]=p;
67         setson(_sz,now,0);
68     }
69     now=_s[now][0];
70 }
71 }
72 _size[now]++; _cnt[now]++;
73 splay(now);
74 }

```

treap比x大的数有多少个

```

1  #include<bits/stdc++.h>
2  using namespace std;
3  typedef long long ll;
4
5  #define fastio ios::sync_with_stdio(false); cin.tie(0);
6  const int N = 2500 + 5;
7
8  struct Point{
9      int x,y;
10 } p[N];
11
12 bool cmp1 (Point a, Point b) {
13     return a.y < b.y;
14 }
15
16 bool cmp2 (Point a, Point b) {
17     return a.x < b.x;
18 }
19
20 void redirect() {
21     #ifdef LOCAL
22         freopen("in.txt","r",stdin);
23         freopen("out.txt","w",stdout);
24     #endif
25 }
26
27 struct treap {
28     int l[N], r[N], val[N], rnd[N], size[N], w[N];
29     int sz, ans, rt;
30     inline void pushup(int x) { size[x] = size[l[x]] + size[r[x]] + w[x]; }
31     void lrotate(int &k) {
32         int t = r[k];
33         r[k] = l[t];
34         l[t] = k;
35         size[t] = size[k];
36         pushup(k);
37         k = t;
38     }
39     void rrotate(int &k) {
40         int t = l[k];
41         l[k] = r[t];
42         r[t] = k;
43         size[t] = size[k];
44         pushup(k);
45         k = t;

```

```

46     }
47     void insert(int &k, int x) {
48         if (!k) {
49             sz++;
50             k = sz;
51             size[k] = 1;
52             w[k] = 1;
53             val[k] = x;
54             rnd[k] = rand();
55             return;
56         }
57         size[k]++;
58         if (val[k] == x) {
59             w[k]++;
60         } else if (val[k] < x) {
61             insert(r[k], x);
62             if (rnd[r[k]] < rnd[k]) lrotate(k);
63         } else {
64             insert(l[k], x);
65             if (rnd[l[k]] < rnd[k]) rrotate(k);
66         }
67     }
68
69     void del(int &k, int x) {
70         if (!k) return;
71         if (val[k] == x) {
72             if (w[k] > 1) {
73                 w[k]--;
74                 size[k]--;
75                 return;
76             }
77             if (l[k] == 0 || r[k] == 0)
78                 k = l[k] + r[k];
79             else if (rnd[l[k]] < rnd[r[k]]) {
80                 rrotate(k);
81                 del(k, x);
82             } else {
83                 lrotate(k);
84                 del(k, x);
85             }
86         } else if (val[k] < x) {
87             size[k]--;
88             del(r[k], x);
89         } else {
90             size[k]--;
91             del(l[k], x);
92         }
93     }
94
95     int queryrank(int k, int x) {
96         if (!k) return 0;
97         if (val[k] == x)
98             return size[l[k]] + 1;
99         else if (x > val[k]) {
100             return size[l[k]] + w[k] + queryrank(r[k], x);
101         } else
102             return queryrank(l[k], x);
103     }

```

```

104
105     int querynum(int k, int x) {
106         if (!k) return 0;
107         if (x <= size[l[k]])
108             return querynum(l[k], x);
109         else if (x > size[l[k]] + w[k])
110             return querynum(r[k], x - size[l[k]] - w[k]);
111         else
112             return val[k];
113     }
114
115     void querypre(int k, int x) {
116         if (!k) return;
117         if (val[k] < x)
118             ans = k, querypre(r[k], x);
119         else
120             querypre(l[k], x);
121     }
122
123     void querysub(int k, int x) {
124         if (!k) return;
125         if (val[k] > x)
126             ans = k, querysub(l[k], x);
127         else
128             querysub(r[k], x);
129     }
130 } T[N];
131
132 map<int, int> mpx;
133 map<int, int> mpy;
134
135 ll check(int i, int j){
136     int l = min(p[i].y, p[j].y), r = max(p[i].y, p[j].y);
137     T[i].insert(T[i].rt, p[j].y);
138     ll lcnt = T[i].queryrank(T[i].rt, l), rcnt = (j - i + 1) -
139     T[i].queryrank(T[i].rt, r) + 1;
140     return lcnt*rcnt;
141 }
142
143 int main(){
144     fastio;
145     redirect();
146     srand(unsigned(time(NULL)));
147     ll ans=0;
148     int n; cin >> n;
149     for(int i=0; i<n; i++){
150         cin>>p[i].x>>p[i].y;
151     }
152     sort (p, p + n, cmp1);
153     for (int i=0; i<n; i++) mpy[p[i].y] = i;
154
155     sort (p, p + n, cmp2);
156     for (int i=0; i<n; i++) mpx[p[i].x] = i;
157
158     for (int i=0; i<n; i++) {
159         p[i].x = mpx[p[i].x];
160         p[i].y = mpy[p[i].y];

```

```

161
162     for(int i=0;i<n;i++){
163         for(int j=i; j<n; j++){
164             ans += check(i,j);
165         }
166     }
167     cout<<ans+1<<endl;
168 }

```

trie思想建树

```

1  #include <bits/stdc++.h>
2  // codeforces 1416C XOR Inverse
3
4  #define mp make_pair
5  #define pb push_back
6  #define f first
7  #define s second
8  #define ll long long
9  #define forn(i, a, b) for(int i = (a); i <= (b); ++i)
10 #define forev(i, b, a) for(int i = (b); i >= (a); --i)
11 #define VAR(v, i) __typeof( i) v=(i)
12 #define forit(i, c) for(VAR(i, (c).begin()); i != (c).end(); ++i)
13 #define all(x) (x).begin(), (x).end()
14 #define sz(x) ((int)(x).size())
15 #define file(s) freopen(s".in","r",stdin); freopen(s".out","w",stdout);
16
17 using namespace std;
18
19 const int maxn = (int)5e6 + 100;
20 const int maxm = (int)1e6 + 100;
21 const int mod = (int)1e9 + 7;
22 const int P = (int) 1e6 + 7;
23 const double pi = acos(-1.0);
24
25 #define inf mod
26
27 typedef long double ld;
28 typedef pair<int, int> pii;
29 typedef pair<ll, ll> pll;
30 typedef vector<int> vi;
31 typedef vector<ll> vll;
32 typedef vector<pair<int, int> > vpii;
33 typedef vector<pair<ll, ll> > vpll;
34
35 int n, t[2][maxn], id = 1;
36 ll dp[2][30];
37 vi g[maxn];
38
39 void add(int x, int pos){
40     int v = 0;
41     forev(i, 29, 0){
42         int bit = ((x >> i) & 1);
43         if(!t[bit][v]) t[bit][v] = id++;
44         v = t[bit][v];
45         g[v].pb(pos);
46     }

```

```

47 }
48 void go(int v, int b = 29){
49     int l = t[0][v], r = t[1][v];
50     if(l) go(l, b - 1);
51     if(r) go(r, b - 1);
52     if(!l || !r) return;
53     ll res = 0;
54     int ptr = 0;
55     for(auto x : g[l]){
56         while(ptr < sz(g[r]) && g[r][ptr] < x) ptr++;
57         res += ptr;
58     }
59     dp[0][b] += res;
60     dp[1][b] += sz(g[l]) * 1ll * sz(g[r]) - res;
61 }
62 void solve(){
63     scanf("%d", &n);
64     forn(i, 1, n){
65         int x;
66         scanf("%d", &x);
67         add(x, i);
68     }
69     go(0);
70     ll inv = 0;
71     int res = 0;
72     forn(i, 0, 29){
73         inv += min(dp[0][i], dp[1][i]);
74         if(dp[1][i] < dp[0][i])
75             res += (1 << i);
76     }
77     printf("%lld %d", inv, res);
78 }
79
80 int main () {
81     int t = 1;
82     //scanf("%d", &t);
83     while(t--) solve();
84 }

```

数学

埃筛

```

1 //埃氏筛法
2 #define N 10000
3 int flag[N+1], p[N+1], pnum;
4 /*
5 flag[n] 表示n是否是素数, 1是素数, 0不是
6 prime 中是所有的素数按从小到大排列、
7 pnum 表示素数的个数
8 */
9 void CreatePrime(){
10     pnum=0; //初始化没有素数
11     //先将所有数看做素数, 然后开始筛选

```



```

12     for(int i=0; i<=N; i++){
13         flag[i]=1;
14     }
15     //遍历筛去所有最大因数是i的合数
16     for(int i=2; i<=N; i++){
17         if(flag[i]==1){
18             //把素数记录下来
19             p[pnum++]=i;
20         }
21         //遍历已知素数表中比i的最小素因数小的素数，并筛去合数
22         for(int j=0; j<pnum && p[j]*i<=N; j++){
23             //筛去合数
24             flag[p[j]*i]=0;
25             if(i%p[j]==0)
26                 //找到i的最小素因数
27                 break;
28         }
29     }
30 }

```

大素数判定+泼辣的肉

```

1  #include<iostream>
2  #include<cstdio>
3  #include<cstring>
4  #include<algorithm>
5  #include<cstdlib>
6  using namespace std;
7  typedef long long ll;
8
9  const int S=20;
10
11 long long mult_mod(long long a,long long b,long long c)
12 {
13     a%=c;
14     b%=c;
15     long long ret=0;
16     while(b)
17     {
18         if(b&1){ret+=a;ret%=c;}
19         a<<=1;
20         if(a>=c)a%=c;
21         b>>=1;
22     }
23     return ret;
24 }
25
26 long long pow_mod(long long x,long long n,long long mod)
27 {
28     if(n==1)return x%mod;
29     x%=mod;
30     long long tmp=x;
31     long long ret=1;
32     while(n)
33     {
34         if(n&1) ret=mult_mod(ret,tmp,mod);
35         tmp=mult_mod(tmp,tmp,mod);

```

```

36     n>=1;
37 }
38 return ret;
39 }
40
41 bool check(long long a,long long n,long long x,long long t)
42 {
43     long long ret=pow_mod(a,x,n);
44     long long last=ret;
45     for(int i=1;i<=t;i++)
46     {
47         ret=mult_mod(ret,ret,n);
48         if(ret==1&&last!=1&&last!=n-1) return true;//合数
49         last=ret;
50     }
51     if(ret!=1) return true;
52     return false;
53 }
54
55
56
57 bool Miller_Rabin(long long n)
58 {
59     if(n<2)return false;
60     if(n==2)return true;
61     if((n&1)==0) return false;
62     long long x=n-1;
63     long long t=0;
64     while((x&1)==0){x>=1;t++;}
65     for(int i=0;i<S;i++)
66     {
67         long long a=rand()%(n-1)+1;
68         if(check(a,n,x,t))
69             return false;
70     }
71     return true;
72 }
73
74 long long factor[100];
75 int tol;
76
77 long long gcd(long long a,long long b)
78 {
79     if(a==0)return 1;//??????
80     if(a<0) return gcd(-a,b);
81     while(b)
82     {
83         long long t=a%b;
84         a=b;
85         b=t;
86     }
87     return a;
88 }
89
90 long long Pollard_rho(long long x,long long c)
91 {
92     long long i=1,k=2;
93     long long x0=rand()%x;

```

```

94     long long y=x0;
95     while(1)
96     {
97         i++;
98         x0=(mult_mod(x0,x0,x)+c)%x;
99         long long d=gcd(y-x0,x);
100        if(d!=1&&d!=x) return d;
101        if(y==x0) return x;
102        if(i==k){y=x0;k+=k;}
103    }
104 }
105
106 void findfac(long long n)
107 {
108     if(Miller_Rabin(n))
109     {
110         factor[tol++]=n;
111         return;
112     }
113     long long p=n;
114     while(p>=n){
115         if (Pollard_rho(p, rand()%(n-1)+1)!=0) p=Pollard_rho(p,rand()%(n-
116     1)+1);
117     }
118     findfac(p);
119     findfac(n/p);
120 }
121
122 int main(void)
123 {
124     int t;
125     cin >> t;
126     while(t--)
127     {
128         ll n;
129         scanf("%lld", &n);
130         if(Miller_Rabin(n)) printf("%lld\n", n);
131         else
132         {
133             tol = 0;
134             findfac(n);
135             ll ans = factor[0];
136             for(int i = 1; i < tol; i++)
137                 ans = min(ans, factor[i]);
138             printf("%lld\n", ans);
139         }
140     }
141     return 0;
142 }

```

第几个质数

```

1 //G++ 1560ms 6544k
2 #include <bits/stdc++.h>
3 #define ll long long
4 using namespace std;
5 ll f[340000],g[340000],n;

```

```

6 void init(){
7     ll i,j,m;
8     for(m=1;m*m<=n;++m)f[m]=n/m-1;
9     for(i=1;i<=m;++i)g[i]=i-1;
10    for(i=2;i<=m;++i){
11        if(g[i]==g[i-1])continue;
12        for(j=1;j<=min(m-1,n/i/i);++j){
13            if(i*j<m)f[j]-=f[i*j]-g[i-1];
14            else f[j]-=g[n/i/j]-g[i-1];
15        }
16        for(j=m;j>=i*i;--j)g[j]-=g[j/i]-g[i-1];
17    }
18 }
19 int main(){
20     while(scanf("%I64d",&n)!=EOF){
21         init();
22         cout<<f[1]<<endl;
23     }
24     return 0;
25 }
26 /*
27
28 O(n^3/4) 筛一个大质数是第几个质数
29 疑似 Meisell-Lehmer算法
30
31 */

```

费马小定理

$$\frac{a}{b} \% mod = a * b^{mod-2} \% mod$$

高精度

```

1 #include<iostream>
2 #include<string>
3 #include<cstring>
4 #include<cstdio>
5 using namespace std;
6 const int N = 1005;
7 struct bign
8 {
9     int len,s[N];
10    bign() { memset(s,0,sizeof(s)); len=1; }
11    bign(int num) { *this=num; }
12    bign(char *num) { *this=num; }
13    bign operator =(int num)
14    {
15        char c[N];
16        sprintf(c,"%d",num);
17        *this=c;
18        return *this;
19    }
20    bign operator =(const char *num)
21    {
22        len=strlen(num);
23        for (int i=0;i<len;i++) s[i]=num[len-1-i]-'0';

```

```

24         return *this;
25     }
26     string str()
27     {
28         string res="";
29         for (int i=0;i<len;i++) res=(char)(s[i]+'0')+res;
30         return res;
31     }
32     void clean()
33     {
34         while (len>1&&!s[len-1]) len--;
35     }
36     bign operator +(const bign &b)
37     {
38         bign c;
39         c.len=0;
40         for (int i=0,g=0;g||i<len||i<b.len;i++)
41         {
42             int x=g;
43             if (i<len) x+=s[i];
44             if (i<b.len) x+=b.s[i];
45             c.s[c.len++]=x%10;
46             g=x/10;
47         }
48         return c;
49     }
50     bign operator -(const bign &b)
51     {
52         bign c;
53         c.len=0;
54         int x;
55         for (int i=0,g=0;i<len;i++)
56         {
57             x=s[i]-g;
58             if (i<b.len) x-=b.s[i];
59             if (x>=0) g=0;
60             else{
61                 x+=10;
62                 g=1;
63             };
64             c.s[c.len++]=x;
65         }
66         c.clean();
67         return c;
68     }
69     bign operator *(const bign &b)
70     {
71         bign c;
72         c.len=len+b.len;
73         for (int i=0;i<len;i++) for (int j=0;j<b.len;j++)
74             c.s[i+j]+=s[i]*b.s[j];
75         for (int i=0;i<c.len-1;i++) { c.s[i+1]+=c.s[i]/10; c.s[i]%=10; }
76         c.clean();
77         return c;
78     }
79     bool operator <(const bign &b)
80     {
81         if (len!=b.len) return len<b.len;

```

```

81         for (int i=len-1;i>=0;i--)
82             if (s[i]!=b.s[i]) return s[i]<b.s[i];
83         return false;
84     }
85     bign operator +=(const bign &b)
86     {
87         *this=*this+b;
88         return *this;
89     }
90     bign operator -=(const bign &b)
91     {
92         *this=*this-b;
93         return *this;
94     }
95 };
96 istream& operator >>(istream &in,bign &x)
97 {
98     string s;
99     in>>s;
100    x=s.c_str();
101    return in;
102 }
103 ostream& operator <<(ostream &out,bign &x)
104 {
105     out<<x.str();
106     return out;
107 }
108 int main(){
109     bign a,b,c;
110     ios::sync_with_stdio(false);
111     cin>>a>>b;
112     // cout<<a<<endl;
113     // cout<<b<<endl;
114     c=a+b;
115     cout<<c<<endl;
116     return 0;
117 }

```

高精度除法

```

1  #include<iostream>
2  #include<algorithm>
3  using namespace std;
4  string div(string a,int b)//高精度a除以单精度b
5  {
6      string r,ans;
7      int d=0;
8      if(a=="0") return a;//特判
9      for(int i=0;i<a.size();i++)
10     {
11         r+=(d*10+a[i]-'0')/b+'0';//求出商
12         d=(d*10+(a[i]-'0'))%b;//求出余数
13     }
14     int p=0;
15     for(int i=0;i<r.size();i++)
16         if(r[i]!='0') {p=i;break;}
17     return r.substr(p);

```

```

18 }
19 int main()
20 {
21     string a;
22     int b;
23     while(cin>>a>>b)
24     {
25         cout<<div(a,b)<<endl;
26     }
27     return 0;
28 }

```

高斯-约旦消元

```

1  int n;
2  double matrix[N][N];
3  double ans[N];
4
5  bool Gauss() {
6      for (int i=1; i<=n; ++i) {
7          //枚举列（项）
8          int mx=i;
9          for (int j=i+1; j<=n; ++j) {
10             //选出该列最大系数
11             if ( fabs(matrix[j][i]) > fabs(matrix[mx][i]) ) {
12                 //fabs是取浮点数的绝对值的函数
13                 mx = j;
14             }
15         }
16         for (int j=1; j<=n+1; ++j) {
17             //交换
18             swap( matrix[i][j], matrix[mx][j] );
19         }
20
21         if (!matrix[i][i]) {
22             //最大值等于0则说明该列都为0，肯定无解
23             // puts("No solution");
24             return false;
25         }
26
27         for(int j=1; j<=n; ++j) {
28             //每一项都减去一个数（就是小学加减消元）
29             if(j != i) {
30                 double temp = matrix[j][i] / matrix[i][i];
31                 for(int k=i+1;k<=n+1;++k) {
32                     matrix[j][k] -= matrix[i][k]*temp;
33                 }
34             }
35         }
36     }
37     //上述操作结束后，矩阵会变成这样
38     /*
39     k1*a=e1
40     k2*b=e2
41     k3*c=e3
42     k4*d=e4
43     */

```

```

44 //所以输出的结果要记得除以该项系数，消去常数
45 for(int i=1;i<=n;++i) {
46     ans[i] = matrix[i][n+1] / matrix[i][i];
47     if ( fabs(ans[i] - 0) < eps ) ans[i] = 0;
48     // printf("%.2lf\n",matrix[i][n+1]/matrix[i][i]);
49 }
50
51 return true;
52 }

```

矩阵快速幂

```

1  #include <bits/stdc++.h>
2  using namespace std;
3
4  long long T,a,b,c,pp,mod;
5  long long n;
6
7  struct mat{
8      long long m[4][4];
9  };
10
11 mat mul(mat a,mat b){
12     mat ans;int i,j,k;
13     for(i=1;i<=3;i++)
14         for(j=1;j<=3;j++)
15             ans.m[i][j]=0;
16     for(i=1;i<=3;i++)
17         for(j=1;j<=3;j++)
18             for(k=1;k<=3;k++)
19                 ans.m[i][j]=(ans.m[i][j]+a.m[i][k]*b.m[k][j])%mod;
20     return ans;
21 }
22
23 mat matqp(mat t,long long p)
24 {
25     mat ans;
26     int i,j;
27     for(i=1;i<=3;i++)
28         for(j=1;j<=3;j++)
29             if(i==j)ans.m[i][j]=1;
30             else ans.m[i][j]=0;
31     while(p)
32     {
33         if(p&1)
34             ans=mul(ans,t);
35         t=mul(t,t);
36         p=p>>1;
37     }
38     return ans;
39 }
40
41 long long qp(long long a,long long p)
42 {
43     long long ans=1;
44     while(p){
45         if(p&1) {ans*=a;ans%=pp;}

```



```

46     a=a*a; a%=pp;
47     p=p>>1;
48 }
49 return ans;
50 }
51
52 int main(){
53     //scanf("%d",&T);
54     cin>>T;
55     while(T--){
56     {
57         //scanf("%I64d %d %d %d %d",&n,&a,&b,&c,&pp);
58         cin>>n>>a>>b>>c>>pp;
59         ///*
60         mod=pp-1;
61         /**/
62         mat base;
63         for(int i=1;i<=3;i++)
64             for(int j=1;j<=3;j++)
65                 base.m[i][j]=0;
66         base.m[1][1]=c;base.m[1][2]=1;base.m[1][3]=1;base.m[2]
[1]=1;base.m[3][3]=1;
67         if(n==1){
68             cout<<1<<endl;
69         }
70         else{
71             mat out = matqp(base,n-2);
72             long long res = out.m[1][1]*b%mod + out.m[1][3]*b%mod;
73             //cout<<res<<endl;
74             long long ans = qp(a,res);
75             cout<<ans<<endl;
76         }
77     }
78
79     return 0;
80 }

```

扩展欧几里得

```

1  int extend_gcd( int a, int b, int &x, int &y ) {
2      if(b==0){
3          x=1;y=0;
4          return a;
5      }else{
6          int r = extend_gcd(b,a%b,y,x);
7          y-=x*(a/b);
8          return r;
9      }
10 }

```

欧拉函数

```

1  int phi(int x)
2  {
3      int ans = x;

```

```

4     for(int i = 2; i*i<=x; i++)
5     {
6         if(x%i==0)
7         {
8             ans = ans/i*(i-1);
9             while(x%i==0) x/=i;
10        }
11    }
12    if(x>1)
13        ans=ans/x*(x-1);
14    return ans;
15 }

```

欧拉筛

```

1 void init() {
2     phi[1] = 1;
3     for (int i = 2; i < MAXN; ++i) {
4         if (!vis[i]) {
5             phi[i] = i - 1;
6             pri[cnt++] = i;
7         }
8         for (int j = 0; j < cnt; ++j) {
9             if (1ll * i * pri[j] >= MAXN) break;
10            vis[i * pri[j]] = 1;
11            if (i % pri[j]) {
12                phi[i * pri[j]] = phi[i] * (pri[j] - 1);
13            } else {
14                phi[i * pri[j]] = phi[i] * pri[j];
15                break;
16            }
17        }
18    }
19 }

```

线性基

```

1 #include <bits/stdc++.h>
2 #define N 51
3 #define ll long long
4 using namespace std;
5
6 //给n个数，输出n个数里异或和的最大值
7
8 int n;
9 ll ans;
10 ll a[N], p[101];
11
12 inline ll read()
13 {
14     char ch = getchar();
15     ll x = 0, f = 1;
16     while(ch > '9' || ch < '0')
17     {
18         if(ch == '-')

```

```

19         f = -1;
20         ch = getchar();
21     }
22     while(ch >= '0' && ch <= '9')
23     {
24         x = x * 10 + ch - '0';
25         ch = getchar();
26     }
27     return x * f;
28 }
29
30 void Get_LB(ll x)
31 {
32     for(int i = 62; i >= 0; i--)
33     {
34         if(!(x >> (ll)i))
35             continue;
36         if(!p[i])
37         {
38             p[i] = x;
39             break;
40         }
41         x ^= p[i];
42     }
43 }
44
45 int main()
46 {
47     n = read();
48     for(int i = 1; i <= n; i++)
49         Get_LB(a[i] = read());
50     for(int i = 62; i >= 0; i--)
51         if((ans ^ p[i]) > ans)
52             ans ^= p[i];
53     cout << ans;
54
55     return 0;
56 }

```

圓和矩形的面积交

```

1  #include<bits/stdc++.h>
2  using namespace std;
3  #define INF 0x3f3f3f3f
4  #define eps 1e-17
5  #define pi acos(-1.0)
6  typedef long long ll;
7
8  void redirect() {
9      #ifdef LOCAL
10         freopen("1.in", "r", stdin);
11         freopen("1.out", "w", stdout);
12     #endif
13 }
14
15 int dcmp(double x){
16     if(fabs(x)<eps)return 0;

```

```

17     return x>0?-1:1;
18 }
19 struct Point{
20     double x,y;
21     Point(double _x=0,double _y=0){
22         x=_x;y=_y;
23     }
24 };
25 Point operator + (const Point &a,const Point &b){
26     return Point(a.x+b.x,a.y+b.y);
27 }
28 Point operator - (const Point &a,const Point &b){
29     return Point(a.x-b.x,a.y-b.y);
30 }
31 Point operator * (const Point &a,const double &p){
32     return Point(a.x*p,a.y*p);
33 }
34 Point operator / (const Point &a,const double &p){
35     return Point(a.x/p,a.y/p);
36 }
37 bool operator < (const Point &a,const Point &b){
38     return a.x<b.x||(dcmp(a.x-b.x)==0&& a.y<b.y);
39 }
40 bool operator == (const Point &a,const Point &b){
41     return dcmp(a.x-b.x)==0&& dcmp(a.y-b.y)==0;
42 }
43 double Dot(Point a,Point b){
44     return a.x*b.x+a.y*b.y;
45 }
46 double Length(Point a){
47     return sqrt(Dot(a,a));
48 }
49 double Angle(Point a,Point b){
50     return acos(Dot(a,b)/Length(a)/Length(b));
51 }
52 double angle(Point a){
53     return atan2(a.y,a.x);
54 }
55 double Cross(Point a,Point b){
56     return a.x*b.y-a.y*b.x;
57 }
58 Point vecunit(Point a){
59     return a/Length(a);
60 }
61 Point Normal(Point a){
62     return Point(-a.y,a.x)/Length(a);
63 }
64 Point Rotate(Point a,double rad){
65     return Point(a.x*cos(rad)-a.y*sin(rad),a.x*sin(rad)+a.y*cos(rad));
66 }
67 double Area2(Point a,Point b,Point c){
68     return Length(Cross(b-a,c-a));
69 }
70 bool OnSegment(Point p,Point a1,Point a2){
71     return dcmp(Cross(a1-p,a2-p))==0&& dcmp(Dot(a1-p,a2-p))<=0;
72 }
73 struct Line{
74     Point p,v;

```

```

75     double ang;
76     Line(){};
77     Line(Point p,Point v):p(p),v(v){
78         ang=atan2(v.y,v.x);
79     }
80     bool operator < (const Line &L) const {
81         return ang<L.ang;
82     }
83     Point point(double d){
84         return p+(v*d);
85     }
86 };
87 bool OnLeft(const Line &L,const Point &p){
88     return Cross(L.v,p-L.p)>=0;
89 }
90 Point GetLineIntersection(Point p,Point v,Point q,Point w){
91     Point u=p-q;
92     double t=Cross(w,u)/Cross(v,w);
93     return p+v*t;
94 }
95 Point GetLineIntersection(Line a,Line b){
96     return GetLineIntersection(a.p,a.v,b.p,b.v);
97 }
98 double PolyArea(vector<Point> p){
99     int n=p.size();
100    double ans=0;
101    for(int i=1;i<n-1;i++)
102        ans+=Cross(p[i]-p[0],p[i+1]-p[0]);
103    return fabs(ans)/2;
104 }
105 struct Circle{
106     Point c;
107     double r;
108     Circle(){}
109     Circle(Point c, double r):c(c), r(r){}
110     Point point(double a) { // 110 111 112 113 114 115 116 117 118 119 120 121 122 123 124 125 126 127 128 129 130 131 132
111         return Point(c.x+cos(a)*r, c.y+sin(a)*r);
112     }
113 };
114
115 bool InCircle(Point x,Circle c){
116     return dcmp(c.r-Length(c.c-x))>=0;
117 }
118 bool OnCircle(Point x,Circle c){
119     return dcmp(c.r-Length(c.c-x))==0;
120 }
121 int getSegCircleIntersection(Line L,Circle C,Point *sol){
122     Point nor=Normal(L.v);
123     Line p1=Line(C.c,nor);
124     Point ip=GetLineIntersection(p1,L);
125     double dis=Length(ip-C.c);
126     if(dcmp(dis-C.r)>0)return 0;
127     Point dxy=vecunit(L.v)*sqrt(C.r*C.r-dis*dis);
128     int ret=0;
129     sol[ret]=ip+dxy;
130     if(OnSegment(sol[ret],L.p,L.point(1)))ret++;
131     sol[ret]=ip-dxy;
132     if(OnSegment(sol[ret],L.p,L.point(1)))ret++;

```

```

133     return ret;
134 }
135 double SegCircleArea(Circle C,Point a,Point b){
136     double a1=angle(a-C.c);
137     double a2=angle(b-C.c);
138     double da=fabs(a1-a2);
139     if(da>pi)da=pi*2-da;
140     return dcmp(Cross(b-C.c,a-C.c))*da*C.r*C.r/2.0;
141 }
142 double PolyCircleArea(Circle C,Point *p,int n){
143     double ret=0;
144     Point sol[2];
145     p[n]=p[0];
146     for(int i=0;i<n;i++){
147         double t1,t2;
148         int cnt=getSegCircleIntersection(Line(p[i],p[i+1]-p[i]),C,sol); //
149         if(cnt==0){ //0得
150
151             if(!InCircle(p[i],C)||!InCircle(p[i+1],C))ret+=SegCircleArea(C,p[i],p[i+1]
152 ); //
153             else ret+=Cross(p[i+1]-C.c,p[i]-C.c)/2; //
154         }
155         if(cnt==1){
156             if(InCircle(p[i],C)&&
157 (!InCircle(p[i+1],C)||OnCircle(p[i+1],C)))ret+=Cross(sol[0]-C.c,p[i]-
158 C.c)/2,ret+=SegCircleArea(C,sol[0],p[i+1]);//,cout<<"jj-1"<<endl;
159             else ret+=SegCircleArea(C,p[i],sol[0]),ret+=Cross(p[i+1]-
160 C.c,sol[0]-C.c)/2;//,cout<<"jj-2"<<endl;
161         }
162         if(cnt==2){
163             if((p[i]<p[i+1])^(sol[0]<sol[1]))swap(sol[0],sol[1]);
164             ret+=SegCircleArea(C,p[i],sol[0]);
165             ret+=Cross(sol[1]-C.c,sol[0]-C.c)/2;
166             ret+=SegCircleArea(C,sol[1],p[i+1]);
167         }
168     }
169     return fabs(ret);
170 }
171 Point p[5];
172 int main(){
173     redirect();
174     double R,x1,y1,x2,y2,x3,y3;
175     cin>>x1>>y1>>R>>x2>>y2>>x3>>y3;
176
177     Circle C=Circle(Point(x1,y1),R);
178     if(x2>x3)swap(x2,x3);
179     if(y2>y3)swap(y2,y3);
180     p[0]=Point(x2,y2);
181     p[2]=Point(x3,y3);
182     p[1]=Point(x3,y2);
183     p[3]=Point(x2,y3);
184     double ans=PolyCircleArea(C,p,4);
185     if(ans < -eps) ans = -ans;
186     printf("%.4lf\n",ans);
187
188     return 0;

```

Min25

```

1  /*
2  * @ author: dragon_bra
3  * @ email: tommy514@foxmail.com
4  * @ data: 2020-09-20 13:59
5  */
6  // n以内素数和
7  #include <algorithm>
8  #include <cmath>
9  #include <cstdio>
10 #include <cstdlib>
11 #include <cstring>
12 #include <iostream>
13 #include <sstream>
14 #include <map>
15 #include <set>
16 #include <queue>
17 #include <vector>
18 using namespace std;
19
20 const int N = 2e5 + 10;
21
22 typedef long long ll;
23
24 void redirect() {
25     #ifdef LOCAL
26         freopen("in.txt", "r", stdin);
27         freopen("out.txt", "w", stdout);
28     #endif
29 }
30
31 int T; ll n, K;
32
33 namespace Min25 {
34
35     ll prime[N], id1[N], id2[N], flag[N], ncnt, m;
36
37     ll g[N], sum[N], a[N], T, n;
38
39     inline int ID(ll x) {
40         return x <= T ? id1[x] : id2[n / x];
41     }
42
43     inline ll calc(ll x) {
44         if (x % 2) return (x+1)/2 % K * x % K;
45         else return x/2 % K * (x+1) % K;
46         // return x * (x + 1) / 2 - 1;
47     }
48
49     inline ll f(ll x) {
50         return x;
51     }
52
53     inline void init() {

```

```

54     T = sqrt(n + 0.5);
55     ncnt = 0; m = 0;
56     memset(flag, 0, sizeof flag);
57     memset(sum, 0, sizeof sum);
58     memset(prime, 0, sizeof prime);
59     memset(a, 0, sizeof a);
60     for (int i = 2; i <= T; i++) {
61         if (!flag[i]) prime[++ncnt] = i, sum[ncnt] = (sum[ncnt - 1] +
i)%K;
62         for (int j = 1; j <= ncnt && i * prime[j] <= T; j++) {
63             flag[i * prime[j]] = 1;
64             if (i % prime[j] == 0) break;
65         }
66     }
67     for (ll l = 1; l <= n; l = n / (n / l) + 1) {
68         a[++m] = n / l;
69         if (a[m] <= T) id1[a[m]] = m; else id2[n / a[m]] = m;
70         g[m] = calc(a[m]) % K;
71     }
72     for (int i = 1; i <= ncnt; i++)
73         for (int j = 1; j <= m && (ll)prime[i] * prime[i] <= a[j]; j++)
74     {
75         g[j] = (g[j] - (ll)prime[i] * (g[id1[a[j]] / prime[i]] -
sum[i - 1] + K) % K + K) % K;
76     }
77     }
78
79     inline ll solve(ll x) {
80         if (x <= 1) return x;
81         return n = x, init(), g[id2(n)];
82     }
83
84 }
85
86 int main() {
87     redirect();
88
89     scanf("%d", &T);
90     while (T--) {
91         scanf("%lld %lld", &n, &K);
92         n = n+1;
93         ll ans = 0;
94         if (n%2) {
95             ans = (n+1)/2 % K * n % K;
96         } else {
97             ans = n/2 % K * (n+1) % K;
98         }
99         ans += Min25::solve(n) - 5;
100        ans %= K;
101        printf("%lld\n", ans);
102    }
103 }

```

Zeller Formula

```

1 | int Day(int year, int month, int day){

```



```

2   int ret = 0;
3   int c, y, m, d;
4   if(month <= 2){
5       c = ( year - 1 ) / 100;
6       y = ( year - 1 ) % 100;
7       m = month + 12;
8       d = day;
9   }
10  else{
11      c = year / 100;
12      y = year % 100;
13      m = month;
14      d = day;
15  }
16  ret = y + y / 4 + c / 4 - 2 * c + 26 * ( m + 1 ) / 10 + d - 1;
17  ret = ret >= 0 ? ( ret % 7 ) : ( ret % 7 + 7 );
18  return ret;
19  }

```

图论

网络流

二分图最大流

```

1   const int maxn = 200005;
2   const int INF = 0x3f3f3f3f;
3
4   struct Edge
5   {
6       int from, to, flow, cap;
7       Edge(int x, int y, int f, int c) : from(x), to(y), flow(f), cap(c) {}
8   };
9
10  vector<Edge> edges;
11  vector<int> G[maxn];
12  int cur[maxn], d[maxn];
13  int S,T;
14  int cnt;
15
16  inline void addedge(int from, int to, int cap)
17  {
18      edges.push_back(Edge(from, to, 0, cap));
19      edges.push_back(Edge(to, from, 0, 0));
20      int m = edges.size();
21      G[from].push_back(m - 2);
22      G[to].push_back(m - 1);
23  }
24
25  int dfs(int u, int a)
26  {
27      if (u == T || a == 0)
28      {
29          return a;

```

```

30     }
31     int flow = 0, f;
32     for (int &i = cur[u]; i < G[u].size(); i++)
33     {
34         Edge &e = edges[G[u][i]];
35         if (d[e.to] > d[u] && (f = dfs(e.to, min(a, e.cap - e.flow))) > 0)
36         {
37             flow += f;
38             e.flow += f;
39             edges[G[u][i] ^ 1].flow -= f;
40             a -= f;
41             if (a == 0)
42             {
43                 break;
44             }
45         }
46     }
47     if (a)
48     {
49         d[u] = -1;
50     }
51     return flow;
52 }
53
54 bool bfs()
55 {
56     memset(d, -1, (T + 1) * sizeof(int));
57     queue<int> q;
58     q.push(S);
59     d[S] = 0;
60     while (!q.empty())
61     {
62         int u = q.front();
63         q.pop();
64         for (int i = 0; i < G[u].size(); i++)
65         {
66             Edge &e = edges[G[u][i]];
67             if (d[e.to] == -1 && e.cap > e.flow)
68             {
69                 d[e.to] = d[u] + 1;
70                 q.push(e.to);
71             }
72         }
73     }
74     return d[T] != -1;
75 }
76
77 int max_flow()
78 {
79     int ans = 0;
80     while (bfs())
81     {
82         memset(cur, 0, (T+1)*sizeof(int));
83         ans += dfs(S, INF);
84     }
85     return ans;
86 }

```

Dinic (Node版本)

```
1 //以下是网络流模板
2 struct Edge{
3     int to,nxt,w;
4 }e[M<<1];
5 int head[N],ecnt;
6 void AddEdge(int u,int v,int w) {
7     e[ecnt]=(Edge){v,head[u],w};
8     head[u]=ecnt++;
9 }
10 void Link(int u,int v,int w){ AddEdge(u,v,w),AddEdge(v,u,0); }
11 #define erep(u,i) for(int i=head[u];~i;i=e[i].nxt)
12
13 int dis[N];
14 int Bfs(){
15     static queue <int> que;
16     rep(i,1,vc) dis[i]=INF;
17     que.push(S),dis[S]=0;
18     while(!que.empty()) {
19         int u=que.front(); que.pop();
20         erep(u,i) {
21             int v=e[i].to,w=e[i].w;
22             if(!w || dis[v]<=dis[u]+1) continue;
23             dis[v]=dis[u]+1,que.push(v);
24         }
25     }
26     return dis[T]<INF;
27 }
28
29 int Dfs(int u,int flowin) {
30     if(u==T) return flowin;
31     int flowout=0;
32     erep(u,i) {
33         int v=e[i].to,w=e[i].w;
34         if(dis[v]!=dis[u]+1 || !w) continue;
35         int t=Dfs(v,min(flowin-flowout,w));
36         flowout+=t,e[i].w-=t,e[i^1].w+=t;
37         if(flowin==flowout) break;
38     }
39     if(!flowout) dis[u]=0;
40     return flowout;
41 }
42
43 int Dinic(){
44     int ans=0;
45     while(Bfs()) ans+=Dfs(S,INF);
46     return ans;
47 }
```

次小生成树

```
1 //Acwing 356. 次小生成树
2 #include <bits/stdc++.h>
3 using namespace std;
4
```

```

5  typedef long long LL;
6
7  const int N = 100010, M = 300010, INF = 0x3f3f3f3f;
8
9  int n, m;
10 struct Edge {
11     int a, b, w;
12     bool used;
13     bool operator< (const Edge &t) const {
14         return w < t.w;
15     }
16 } edge[M];
17 int p[N];
18 int h[N], e[M], w[M], ne[M], idx;
19 int depth[N], fa[N][17], d1[N][17], d2[N][17];
20 int q[N];
21
22 void add(int a, int b, int c) {
23     e[idx] = b, w[idx] = c, ne[idx] = h[a], h[a] = idx ++;
24 }
25
26 int find(int x) {
27     return p[x] == x ? x : p[x] = find(p[x]);
28 }
29
30 LL kruskal() {
31     for (int i = 1; i <= n; i ++ ) p[i] = i;
32     sort (edge, edge + m);
33
34     LL res = 0;
35     for (int i = 0; i < m; i ++ ) {
36         int a = find(edge[i].a), b = find(edge[i].b), w = edge[i].w;
37         if (a != b) {
38             p[a] = b;
39             res += w;
40             edge[i].used = true;
41         }
42     }
43
44     return res;
45 }
46
47 void build() {
48     memset(h, -1, sizeof h);
49     for (int i = 0; i < m; i ++ ) {
50         if (edge[i].used) {
51             int a = edge[i].a, b = edge[i].b, w = edge[i].w;
52             add(a, b, w); add(b, a, w);
53         }
54     }
55 }
56
57 void bfs() {
58     memset(depth, 0x3f, sizeof depth);
59     depth[0] = 0, depth[1] = 1;
60     q[0] = 1;
61     int hh = 0, tt = 0;
62     while (hh <= tt) {

```

```

63     int t = q[hh ++ ];
64     for (int i = h[t]; ~i; i = ne[i]) {
65         int j = e[i];
66         if (depth[j] > depth[t] + 1) {
67             depth[j] = depth[t] + 1;
68             q[ ++ tt] = j;
69             fa[j][0] = t;
70             d1[j][0] = w[i], d2[j][0] = -INF;
71             for (int k = 1; k <= 16; k ++ ) {
72                 int anc = fa[j][k - 1];
73                 fa[j][k] = fa[fa[j][k - 1]][k - 1];
74                 int distance[4] = {d1[j][k - 1], d2[j][k - 1], d1[anc]
[k - 1], d2[anc][k - 1]};
75                 d1[j][k] = d2[j][k] = -INF;
76                 for (int u = 0; u < 4; u ++ ) {
77                     int d = distance[u];
78                     if (d > d1[j][k]) d2[j][k] = d2[j][k], d1[j][k] =
d;
79                     else if (d != d1[j][k] && d > d2[j][k]) d2[j][k] =
d;
80                 }
81             }
82         }
83     }
84 }
85 }
86
87 int lca(int a, int b, int w) {
88     static int distance[N * 2];
89     int cnt = 0;
90     if (depth[a] < depth[b]) swap(a, b);
91     for (int k = 16; k >= 0; k -- ) {
92         if (depth[fa[a][k]] >= depth[b]) {
93             distance[cnt ++ ] = d1[a][k];
94             distance[cnt ++ ] = d2[a][k];
95             a = fa[a][k];
96         }
97     }
98     if (a != b) {
99         for (int k = 16; k >= 0; k -- ) {
100             if (fa[a][k] != fa[b][k]) {
101                 distance[cnt ++ ] = d1[a][k];
102                 distance[cnt ++ ] = d2[a][k];
103                 distance[cnt ++ ] = d1[b][k];
104                 distance[cnt ++ ] = d2[b][k];
105                 a = fa[a][k], b = fa[b][k];
106             }
107         }
108         distance[cnt ++ ] = d1[a][0];
109         distance[cnt ++ ] = d1[b][0];
110     }
111
112     int dist1 = -INF, dist2 = -INF;
113     for (int i = 0; i < cnt; i ++ ) {
114         int d = distance[i];
115         if (d > dist1) dist2 = dist1, dist1 = d;
116         else if (d != dist1 && d > dist2) dist2 = d;
117     }

```

```

118
119     if (w > dist1) return w - dist1;
120     if (w > dist2) return w - dist2;
121     return INF;
122 }
123
124 int main() {
125     cin >> n >> m;
126     for (int i = 0; i < m; i++) {
127         int a, b, c;
128         cin >> a >> b >> c;
129         edge[i] = {a, b, c};
130     }
131
132     LL sum = kruskal();
133     build();
134
135     bfs(); // 倍增初始化部分
136
137     LL res = 1e18 + 10;
138     for (int i = 0; i < m; i++) {
139         if (!edge[i].used) {
140             int a = edge[i].a, b = edge[i].b, w = edge[i].w;
141             res = min(res, sum + lca(a, b, w));
142         }
143     }
144
145     cout << res << "\n";
146 } // Acwing 356. 次小生成树
147 #include <bits/stdc++.h>
148 using namespace std;
149
150 typedef long long LL;
151
152 const int N = 100010, M = 300010, INF = 0x3f3f3f3f;
153
154 int n, m;
155 struct Edge {
156     int a, b, w;
157     bool used;
158     bool operator< (const Edge &t) const {
159         return w < t.w;
160     }
161 } edge[M];
162 int p[N];
163 int h[N], e[M], w[M], ne[M], idx;
164 int depth[N], fa[N][17], d1[N][17], d2[N][17];
165 int q[N];
166
167 void add(int a, int b, int c) {
168     e[idx] = b, w[idx] = c, ne[idx] = h[a], h[a] = idx++;
169 }
170
171 int find(int x) {
172     return p[x] == x ? x : p[x] = find(p[x]);
173 }
174
175 LL kruskal() {

```

```

176     for (int i = 1; i <= n; i ++ ) p[i] = i;
177     sort (edge, edge + m);
178
179     LL res = 0;
180     for (int i = 0; i < m; i ++ ) {
181         int a = find(edge[i].a), b = find(edge[i].b), w = edge[i].w;
182         if (a != b) {
183             p[a] = b;
184             res += w;
185             edge[i].used = true;
186         }
187     }
188
189     return res;
190 }
191
192 void build() {
193     memset(h, -1, sizeof h);
194     for (int i = 0; i < m; i ++ ) {
195         if (edge[i].used) {
196             int a = edge[i].a, b = edge[i].b, w = edge[i].w;
197             add(a, b, w); add(b, a, w);
198         }
199     }
200 }
201
202 void bfs() {
203     memset(depth, 0x3f, sizeof depth);
204     depth[0] = 0, depth[1] = 1;
205     q[0] = 1;
206     int hh = 0, tt = 0;
207     while (hh <= tt) {
208         int t = q[hh ++ ];
209         for (int i = h[t]; ~i; i = ne[i]) {
210             int j = e[i];
211             if (depth[j] > depth[t] + 1) {
212                 depth[j] = depth[t] + 1;
213                 q[ ++ tt] = j;
214                 fa[j][0] = t;
215                 d1[j][0] = w[i], d2[j][0] = -INF;
216                 for (int k = 1; k <= 16; k ++ ) {
217                     int anc = fa[j][k - 1];
218                     fa[j][k] = fa[fa[j][k - 1]][k - 1];
219                     int distance[4] = {d1[j][k - 1], d2[j][k - 1], d1[anc]
220 [k - 1], d2[anc][k - 1]};
221                     d1[j][k] = d2[j][k] = -INF;
222                     for (int u = 0; u < 4; u ++ ) {
223                         int d = distance[u];
224                         if (d > d1[j][k]) d2[j][k] = d2[j][k], d1[j][k] =
225 d;
226                         else if (d != d1[j][k] && d > d2[j][k]) d2[j][k] =
227 d;
228                     }
229                 }
230             }
231         }
232     }
233 }
234 }
235 }
236 }
237 }
238 }
239 }
240 }

```

```

231
232 int lca(int a, int b, int w) {
233     static int distance[N * 2];
234     int cnt = 0;
235     if (depth[a] < depth[b]) swap(a, b);
236     for (int k = 16; k >= 0; k -- ) {
237         if (depth[fa[a][k]] >= depth[b]) {
238             distance[cnt ++ ] = d1[a][k];
239             distance[cnt ++ ] = d2[a][k];
240             a = fa[a][k];
241         }
242     }
243     if (a != b) {
244         for (int k = 16; k >= 0; k -- ) {
245             if (fa[a][k] != fa[b][k]) {
246                 distance[cnt ++ ] = d1[a][k];
247                 distance[cnt ++ ] = d2[a][k];
248                 distance[cnt ++ ] = d1[b][k];
249                 distance[cnt ++ ] = d2[b][k];
250                 a = fa[a][k], b = fa[b][k];
251             }
252         }
253         distance[cnt ++ ] = d1[a][0];
254         distance[cnt ++ ] = d1[b][0];
255     }
256
257     int dist1 = -INF, dist2 = -INF;
258     for (int i = 0; i < cnt; i ++ ) {
259         int d = distance[i];
260         if (d > dist1) dist2 = dist1, dist1 = d;
261         else if (d != dist1 && d > dist2) dist2 = d;
262     }
263
264     if (w > dist1) return w - dist1;
265     if (w > dist2) return w - dist2;
266     return INF;
267 }
268
269 int main() {
270     cin >> n >> m;
271     for (int i = 0; i < m; i ++ ) {
272         int a, b, c;
273         cin >> a >> b >> c;
274         edge[i] = {a, b, c};
275     }
276
277     LL sum = kruskal();
278     build();
279
280     bfs(); // 倍增初始化部分
281
282     LL res = 1e18 + 10;
283     for (int i = 0; i < m; i ++ ) {
284         if (!edge[i].used) {
285             int a = edge[i].a, b = edge[i].b, w = edge[i].w;
286             res = min(res, sum + lca(a, b, w));
287         }
288     }

```



```

289
290     cout << res << "\n";
291 }

```

二分图匹配-匈牙利算法

```

1  /*
2  Problem: HDU 2063 过山车 匈牙利算法-二分图匹配模板题
3  * @ author: dragon_bra
4  * @ email: tommy514@foxmail.com
5  * @ date: 2021-01-26 22:11
6  */
7
8  #include <bits/stdc++.h>
9  #define fastio ios::sync_with_stdio(false); cin.tie(0);
10 using namespace std;
11
12 typedef long long ll;
13 const int N = 500 + 10;
14
15 void redirect() {
16     #ifdef LOCAL
17         freopen("in.txt", "r", stdin);
18         freopen("out.txt", "w", stdout);
19     #endif
20 }
21
22 int k, m, n;
23 int line[N][N], used[N], nxt[N];
24
25 bool Find(int x) {
26     for (int i=1; i<=m; i++) {
27         if (line[x][i] && !used[i]) {
28             used[i] = 1;
29             if (nxt[i] == 0 || Find(nxt[i])) {
30                 nxt[i] = x;
31                 return true;
32             }
33         }
34     }
35     return false;
36 }
37
38 int match() {
39     int sum = 0;
40     for (int i=1; i<=n; i++) {
41         memset(used, 0, sizeof(used));
42         if (Find(i)) sum ++;
43     }
44     return sum;
45 }
46
47 int main() {
48     redirect();
49
50     while (cin >> k && k) {
51         memset(line, 0, sizeof(line));

```

```

52     memset(nxt, 0, sizeof(nxt));
53     cin >> n >> m;
54     for (int i=1; i<=k; i++) {
55         int u, v;
56         cin >> u >> v;
57         line[u][v] = true;
58     }
59     cout << match() << "\n";
60 }
61
62 return 0;
63 }

```

dijkstra

```

1
2 // Problem: C. Dijkstra?
3 // Contest: Codeforces - Codeforces Alpha Round #20 (Codeforces format)
4 // URL: https://codeforces.com/problemset/problem/20/C
5 // Memory Limit: 64 MB
6 // Time Limit: 1000 ms
7 // Powered by CP Editor (https://github.com/cpeditor/cpeditor)
8
9 /*
10  @ author: dragon_bra
11  @ QQ: 1277037638
12  @ email: tommy514@foxmail.com
13 */
14
15 #include <bits/stdc++.h>
16 #define fastio ios_base::sync_with_stdio(false); cin.tie(0);
17 using namespace std;
18
19 typedef long long ll;
20 const ll INF = 1e18;
21 const int N = 2e5 + 10;
22
23 int n, m;
24 struct edge {
25     int v; ll w;
26     edge(int v, ll w):v(v), w(w){}
27 };
28 vector<edge> G[N];
29 struct node {
30     int u; ll dis;
31     node(int u, ll dis):u(u), dis(dis){}
32     friend bool operator<(node a, node b) {
33         return a.dis > b.dis;
34     }
35 };
36 ll dis[N];
37 ll f[N];
38 bool vis[N];
39 int ans[N];
40

```

```

41 void init() {
42     for (int i=1; i<=n; i++) {
43         dis[i] = INF;
44         vis[i] = false;
45     }
46 }
47
48 int main() {
49     fastio;
50     cin >> n >> m;
51
52     init();
53
54     for (int i=1; i<=m; i++) {
55         int u, v; ll w;
56         cin >> u >> v >> w;
57         G[u].push_back(edge(v, w));
58         G[v].push_back(edge(u, w));
59     }
60
61     priority_queue<node> Q; Q.push(node(1, 0)); dis[1] = 0;
62     while (!Q.empty()) {
63         node now = Q.top(); Q.pop();
64         int u = now.u; ll d = now.dis;
65         if (vis[u]) continue;
66         vis[u] = true;
67         // cout << u << ' ' << d << endl;
68         for (auto nxt: G[u]) {
69             int v = nxt.v; ll w = nxt.w;
70             if (d + w < dis[v]) {
71                 dis[v] = d + w;
72                 f[v] = u;
73                 Q.push(node(v, dis[v]));
74             }
75         }
76     }
77
78     int cnt = 0; int x = n;
79     while (x != 1) {
80         if (f[x] == 0) break;
81         ans[++cnt] = x;
82         x = f[x];
83     }
84     if (cnt == 0) {
85         puts("-1");
86     } else {
87         ans[++cnt] = 1;
88         for (int i=cnt; i>=1; i--) {
89             cout << ans[i] << ' ';
90         }
91     }
92
93     return 0;
94 }
95
96

```

LCA

LCA-倍增

```
1  /*
2      洛谷P3379, LCA模板
3  */
4  #include <bits/stdc++.h>
5  using namespace std;
6
7  const int N = 5e5 + 10, M = N * 2;
8  const int LOG = 30 + 1;
9
10 int n, m;
11 int h[N], e[M], ne[M], idx;
12 int depth[N], fa[N][LOG];
13 int q[N];
14
15 void add(int a, int b) {
16     e[idx] = b, ne[idx] = h[a], h[a] = idx ++;
17 }
18
19 void bfs(int root) {
20     memset(depth, 0x3f3f3f3f, sizeof depth);
21     depth[0] = 0, depth[root] = 1;
22     int hh = 0, tt = 0;
23     q[0] = root;
24     while (hh <= tt) {
25         int t = q[hh ++ ];
26         for (int i = h[t]; ~i; i = ne[i] ) {
27             int j = e[i];
28             if (depth[j] > depth[t] + 1) {
29                 depth[j] = depth[t] + 1;
30                 q[ ++ tt] = j;
31                 fa[j][0] = t;
32                 for (int k = 1; k < LOG; k ++ )
33                     fa[j][k] = fa[fa[j][k - 1]][k - 1];
34             }
35         }
36     }
37 }
38
39 int lca(int a, int b) {
40     if (depth[a] < depth[b]) swap(a, b);
41     for (int k = LOG - 1; k >= 0; k -- ) {
42         if (depth[fa[a][k]] >= depth[b]) // 哨兵解决depth['0'] = '0' 满足不成立
            的条件
43             a = fa[a][k];
44     }
45
46     if (a == b) return a;
47     for (int k = LOG - 1; k >= 0; k -- ) {
48         if (fa[a][k] != fa[b][k]) { // 哨兵解决跳出去后
49             a = fa[a][k];
50             b = fa[b][k];
51         }
52     }
53     return fa[a][0];
54 }
```

```

51     }
52 }
53 return fa[a][0];
54 }
55
56 int main() {
57     #ifdef LOCAL
58         freopen("in.txt", "r", stdin);
59         freopen("out.txt", "w", stdout);
60     #endif
61     int root = 0;
62     cin >> n >> m >> root;
63     memset(h, -1, sizeof h);
64
65     for (int i = 1; i < n; i++) {
66         int a, b;
67         scanf("%d%d", &a, &b);
68         add(a, b), add(b, a);
69     }
70
71     bfs(root);
72
73     while (m--) {
74         int a, b;
75         scanf("%d%d", &a, &b);
76         int p = lca(a, b);
77         printf("%d\n", p);
78     }
79
80 }

```

LCA-tarjan

```

1 //Acwing 1171. 距离
2 #include <bits/stdc++.h>
3 using namespace std;
4
5 typedef pair<int, int> PII;
6
7 const int N = 2e4 + 10, M = N * 2;
8
9 int n, m;
10 int h[N], e[M], w[M], ne[M], idx;
11 int dist[N];
12 int p[N];
13 int st[N];
14 int res[N];
15 vector<PII> query[N]; // first存查询的另外一个点, second存查询编号
16
17 void add(int a, int b, int c) {
18     e[idx] = b, w[idx] = c, ne[idx] = h[a], h[a] = idx++;
19 }
20
21 void dfs(int u, int fa) {
22     for (int i = h[u]; ~i; i = ne[i]) {
23         int j = e[i];
24         if (j == fa) continue;

```

```

25     dist[j] = dist[u] + w[i];
26     dfs(j, u);
27 }
28 }
29
30 int find(int x) {
31     return p[x] == x ? x : p[x] = find(p[x]);
32 }
33
34 void tarjan(int u) {
35     st[u] = 1;
36     for (int i = h[u]; ~i; i = ne[i]) {
37         int j = e[i];
38         if (!st[j]) {
39             tarjan(j);
40             p[j] = u;
41         }
42     }
43
44     for (auto item : query[u]) {
45         int y = item.first, id = item.second;
46         if (st[y] == 2) {
47             int anc = find(y);
48             res[id] = dist[u] + dist[y] - 2 * dist[anc];
49         }
50     }
51
52     st[u] = 2;
53 }
54
55 int main() {
56     cin >> n >> m;
57     memset(h, -1, sizeof h);
58     for (int i = 1; i < n; i++) {
59         int a, b, c;
60         cin >> a >> b >> c;
61         add(a, b, c); add(b, a, c);
62     }
63
64     for (int i = 1; i <= m; i++) {
65         int a, b;
66         cin >> a >> b;
67         if (a != b) {
68             query[a].push_back({b, i});
69             query[b].push_back({a, i});
70         }
71     }
72
73     for (int i = 1; i <= n; i++) p[i] = i;
74
75     dfs(1, -1);
76     tarjan(1);
77
78     for (int i = 1; i <= m; i++) cout << res[i] << "\n";
79 }

```

tarjan求割点

```
1
2 // Problem: P3388 【模板】割点（割顶）
3 // Contest: Luogu
4 // URL: https://www.luogu.com.cn/problem/P3388
5 // Memory Limit: 125 MB
6 // Time Limit: 1000 ms
7 // Powered by CP Editor (https://github.com/cpeditor/cpeditor)
8
9 /*
10  @ author: dragon_bra
11  @ QQ: 1277037638
12  @ email: tommy514@foxmail.com
13 */
14
15 #include <bits/stdc++.h>
16 #define fastio ios_base::sync_with_stdio(false); cin.tie(0);
17 using namespace std;
18
19 const int N = 2e5 + 10;
20
21 int n,m;
22 struct edge {
23     int next,to;
24 } p[N];
25
26 int head[N], num; // num stands for edge number
27
28 void addEdge(int x,int y) {
29     p[++num].next=head[x];
30     p[num].to=y;
31     head[x]=num;
32 }
33 int dfn[N], low[N], tim, cut[N];
34 // tim 代表入栈的顺序是第几个
35 // cut[i]代表该点是否是割点
36
37 void tag (int x,int zx) {
38     // zx 代表最早出现的祖先
39     int kid = 0;
40     dfn[x] = low[x] = ++tim;
41
42     for(int i=head[x]; i; i=p[i].next) {
43         int v = p[i].to;
44
45         if(!dfn[v]) {
46             tag(v, zx);
47             low[x] = min(low[v], low[x]);
48             if(low[v] >= dfn[x] && x!=zx) cut[x]=1;
49             if(x==zx) kid++;
50         }
51
52         low[x] = min(low[x], dfn[v]);
53     }
54     if(kid>1 && x==zx) cut[x]=1;
55     // 如果有两个及以上的儿子，则也是割点
```

```

56 }
57
58 int ans;
59
60 int main() {
61     fastio;
62     cin >> n >> m;
63     for (int i=1; i<=m; i++) {
64         int u, v;
65         cin >> u >> v;
66         addEdge(u, v);
67         addEdge(v, u);
68     }
69
70     for(int i=1;i<=n;i++) if(!dfn[i]) tag(i,i);
71
72     for(int i=1;i<=n;i++) ans += cut[i];
73     printf("%d\n",ans);
74     for(int i=1;i<=n;i++)
75         if(cut[i]) printf("%d ",i);
76
77     return 0;
78 }
79

```

tarjan缩点

```

1
2 // Problem: P3387 【模板】缩点
3 // Contest: Luogu
4 // URL: https://www.luogu.com.cn/problem/P3387
5 // Memory Limit: 125 MB
6 // Time Limit: 1000 ms
7 // Powered by CP Editor (https://github.com/cpeditor/cpeditor)
8
9 /*
10  @ author: dragon_bra
11  @ QQ: 1277037638
12  @ email: tommy514@foxmail.com
13 */
14
15 #include <bits/stdc++.h>
16 #define fastio ios_base::sync_with_stdio(false); cin.tie(0);
17 using namespace std;
18
19 const int N = 10000+15;
20 int n, m;
21 vector<int> G[N];
22 vector<int> G2[N];
23 int tim, top;
24 int p[N], belong[N], dfn[N], low[N];
25 //DFN(u)为节点u搜索被搜索到时的次序编号(时间戳), Low(u)为u或u的子树能够追溯到的最早的
   栈中节点的次序号
26 int stac[N], vis[N];
27 //栈只为了表示此时是否有父子关系
28 int in[N], dist[N];
29

```



```

30 void tarjan(int x) {
31     // tarjan 缩点核心代码
32     low[x]=dfn[x]++;
33     stac[++top]=x;vis[x]=1;
34     for (int v:G[x]) {
35         if (!dfn[v]) {
36             tarjan(v);
37             low[x] = min(low[x], low[v]);
38         } else if (vis[v]) {
39             low[x] = min(low[x], low[v]);
40         }
41     }
42     if (dfn[x]==low[x]) {
43         int y;
44         while (y=stac[top--]) {
45             belong[y] = x;
46             vis[y] = 0;
47             if (x==y) break;
48             p[x] += p[y]; // 增加点权, 本题有效
49         }
50     }
51 }
52
53 int topo() {
54     queue<int> Q;
55     for (int i=1; i<=n; i++) {
56         if (belong[i]==i && !in[i]) {
57             Q.push(i);
58             dist[i] = p[i];
59         }
60     }
61
62     while (!Q.empty()) {
63         int now = Q.front(); Q.pop();
64         for (int v:G2[now]) {
65             dist[v] = max(dist[v], dist[now] + p[v]);
66             in[v]--;
67             if (in[v]==0) Q.push(v);
68         }
69     }
70
71     int ans = 0;
72     for (int i=1; i<=n; i++) ans = max(ans, dist[i]);
73
74     return ans;
75 }
76
77 int main() {
78     fastio;
79     cin >> n >> m;
80     for (int i=1; i<=n; i++) cin >> p[i];
81
82     for (int i=1; i<=m; i++) {
83         int u, v; cin >> u >> v;
84         G[u].push_back(v);
85     }
86
87     for (int i=1; i<=n; i++)

```

```

88         if (!dfn[i]) tarjan(i);
89
90     for (int i=1; i<=n; i++) {
91         for (int v:G[i]) {
92             if (belong[i] == belong[v]) continue;
93             G2[belong[i]].push_back(belong[v]);
94             in[belong[v]] ++;
95         }
96     }
97
98     printf("%d",topo());
99
100    return 0;
101 }

```

字符串

```

1  #include <cstdio>
2  #include <iostream>
3  #include <algorithm>
4  #include <cmath>
5  #include <cstring>
6  #include <map>
7  #include <set>
8  #include <queue>
9  #include <string>
10 #include <vector>
11 using namespace std;
12 typedef long long ll;
13 typedef unsigned long long ull;
14 const int INF = 0x7fffffff;
15 const int mod = 1e9+7;
16 const double eps = 1e-5;
17 const int N = 1e6+10;
18
19 void redirect() {
20     #ifdef LOCAL
21         //freopen("test.txt","r",stdin);
22         //freopen("out.txt","w",stdout);
23     #endif
24 }
25 inline ll read() {
26     ll f=1,x=0;char ch;
27     do {ch=getchar(); if(ch=='-') f=-1;} while (ch<'0' || ch>'9');
28     do {x=x*10+ch-'0'; ch=getchar(); } while (ch>='0' && ch<='9');
29     return x*f;
30 }
31
32 struct Trie {
33     int next[N][26],fail[N],end[N];
34     int root,L;
35     int newnode(){
36         for(int i=0;i<26;i++)
37             next[L][i] = -1;

```

```

38     end[L++] = 0;
39     return L-1;
40 }
41 void init(){
42     L = 0;
43     root = newnode();
44 }
45 void insert(char buf[]){
46     int len = strlen(buf);
47     int now = root;
48     for(int i=0;i<len;i++){
49         if(next[now][buf[i]-'a'] == -1)
50             next[now][buf[i]-'a'] = newnode();
51         now = next[now][buf[i]-'a'];
52     }
53     end[now]++;
54 }
55 void build(){
56     queue<int>Q;
57     fail[root] = root;
58     for(int i=0;i<26;i++){
59         if(next[root][i] == -1)
60             next[root][i] = root;
61         else{
62             fail[next[root][i]] = root;
63             Q.push(next[root][i]);
64         }
65     }
66     while( !Q.empty() ) {
67         int now = Q.front();
68         Q.pop();
69         for(int i=0;i<26;i++){
70             if(next[now][i] == -1)
71                 next[now][i] = next[fail[now]][i];
72             else{
73                 fail[next[now][i]] = next[fail[now]][i];
74                 Q.push(next[now][i]);
75             }
76         }
77     }
78     int query(char buf[]){
79         int len = strlen(buf);
80         int now = root;
81         int res = 0;
82         for(int i=0;i<len;i++){
83             now = next[now][buf[i]-'a'];
84             int temp = now;
85             while( temp != root ) {
86                 res += end[temp];
87                 end[temp] = 0;
88                 temp = fail[temp];
89             }
90         }
91         return res;
92     }
93     void debug(){
94         for(int i = 0;i < L;i++){
95             printf("id=%3d,fail=%3d,end=%3d,chi=%3d",i,fail[i],end[i]);

```

```

96         printf("%2d",next[i][j]);
97     printf("]\n");
98     }
99 }
100 };
101 char buf[N];
102 Trie ac;
103
104 int main() {
105     redirect();
106     int T; scanf("%d",&T);
107     int n;
108     while ( T-- ) {
109         scanf("%d",&n);
110         ac.init();
111         for(int i=0;i<n;i++){
112             scanf("%s",buf);
113             ac.insert(buf);
114         }
115         ac.build();
116         scanf("%s",buf);
117         printf("%d\n",ac.query(buf));
118     }
119 }
120
121 /*
122 -----
123 author:dragon_bra
124 -----
125 */

```

KMP

```

1 void makeNext(string s) {
2     int i = 0, k = -1;
3     next[0] = -1;
4     int len = strlen(s);
5     while (i < len-1) {
6         while (k >= 0 && s[i] != s[k]) k = next[k];
7         i++; k++;
8         if (s[i] == s[k]) next[i] = next[k];
9         else next[i] = k;
10    }
11 }
12
13 int kmpMatch(string t, string p) {
14     int i = 0, j = 0;
15     int len_1 = strlen(t), len2 = strlen(p);
16     while (i < len_1 && j < len2) {
17         if (i == -1 || p[i] == c[j]) {
18             i++; j++;
19         } else {
20             i = next[i];
21         }
22     }
23     if (i >= len_1) return j - len_1 + 1;
24     else return 0;

```

Manachar

```

1  /*
2  * @ author: dragon_bra
3  * @ email: tommy514@foxmail.com
4  * @ data: 2020-05-16 15:19
5  */
6
7  #include <algorithm>
8  #include <cmath>
9  #include <cstdio>
10 #include <cstdlib>
11 #include <cstring>
12 #include <iostream>
13 #include <sstream>
14 #include <map>
15 #include <set>
16 #include <queue>
17 #include <vector>
18
19 using namespace std;
20
21 typedef long long ll;
22 const int INF = 0x3f3f3f3f;
23 const int mod = 1e9+7;
24 const double eps = 1e-5;
25 const int N = 2e5 + 10;
26
27 void redirect() {
28     #ifdef LOCAL
29         freopen("in.txt", "r", stdin);
30         freopen("out.txt", "w", stdout);
31     #endif
32 }
33
34 int p[N*2];
35 char str[N*2], t[N*2];
36
37 int Manacher(char *str, int len){
38     // 初始化部分
39     t[0] = '$'; t[1] = '#';
40     int tot = 2;
41     for(int i=0; i<len; i++){
42         t[tot++] = str[i];
43         t[tot++] = '#';
44     }
45
46     int mx = 0, id = 0, reslen = 0, resCenter = 0;
47     for(int i=0; i<tot; i++){
48         if(i<mx) p[i] = min(p[2*id - i], mx - i); // 2*id - i = id - (i-
49         id); j和i关于id对称;
50         else p[i] = 1; // i比mx大了, 也就是当前最大的回文串够不着它了
51
52         while( t[i+p[i]] == t[i-p[i]] ) p[i] ++; // 计算i为中心大时候, 最大的回文
53         串有多大

```

```

52         if(p[i]+i > mx){
53             mx = i + p[i];
54             id = i;
55         }
56
57         if(reslen < p[i]) {
58             reslen = p[i], resCenter = i;
59         }
60
61     }
62     return reslen;
63 }
64
65 int main(){
66     while(~scanf("%s", str)){
67         int len = strlen(str);
68         printf("%d\n",Manacher(str,len)-1);
69     }
70     return 0;
71 }

```

最大字典序子串

```

1  string lastSubstring(string s) {
2      int left=0;
3      int right=left+1;
4      int step=0;
5      while(right + step < s.size()){
6          if(s[left+step]<s[right+step]){
7              left=right;
8              right=left+1;
9              step=0;
10         }
11         else if(s[left+step]==s[right+step]){
12             step++;
13         }
14         else{ // s[left+step]>s[right+step]
15             right+=step+1;
16             step=0;
17         }
18     }
19     return s.substr(left, s.size()-left);
20 }

```

最大最小表示法

```

1  int min_max_express(bool flag) // flag=true的时候为字典序最小，=false的时候为字典
序最大
2  {
3      int i = 0, j = 1, k = 0, t;
4      while(i < len && j < len && k < len)
5      {
6          t = str[(i + k) % len] - str[(j + k) % len];
7          if(!t) k++;
8          else

```

```

9      {
10         if(flag)
11         {
12             if(t > 0) i = i + k + 1;
13             else j = j + k + 1;
14         }
15         else
16         {
17             if(t > 0) j = j + k + 1;
18             else i = i + k + 1;
19         }
20
21         if(j == i) j++;
22         k = 0;
23     }
24 }
25
26 return i < j ? i : j;
27 }

```

DFS

DSU（树上启发式合并）

```

1  /*
2
3  DSU-on-tree
4  树上启发式合并
5  重点：{
6      dfs1(): 找出所有节点的重儿子，记录每个节点的子树大小
7      dfs2(): 搜索下去更新答案，
8              如果是重儿子，
9                  将兄弟所有的集合合并到重儿子，并将重儿子的答案合并到父亲节点
10      else 如果是轻儿子
11              寻找他的重儿子并先把答案合并到自己
12  }
13
14  */
15  #include <bits/stdc++.h>
16  using namespace std;
17
18  typedef long long ll;
19  const int N = 1e5 + 5;
20
21  void redirect() {
22      #ifdef LOCAL
23          freopen("1.in", "r", stdin);
24          freopen("1.out", "w", stdout);
25      #endif
26  }
27
28  int n, f[N];
29  int son[N], size[N];
30  ll ans[N], rans[N];

```

```

31
32 vector<int> G[N];
33 set<ll> S[N];
34
35 void merge(int a,int b) {
36     while(!S[b].empty()){
37         ll t = *( S[b].begin() ); S[b].erase( t );
38
39         ll up=0, low=0;
40
41         if( S[a].upper_bound(t) == S[a].begin() ) {
42             up = *S[a].begin();
43             ans[a] += ( up - t ) * ( up - t );
44         } else if( S[a].upper_bound(t) == S[a].end() ) {
45             low = * ( --S[a].lower_bound(t) );
46             ans[a] += ( t - low ) * ( t - low );
47         } else {
48             up = * ( S[a].upper_bound(t) ); low = * ( --S[a].lower_bound(t)
49         );
50             ans[a] -= ( up - low ) * ( up - low ); ans[a] += ( up - t ) * (
51         up - t ); ans[a] += ( t - low ) * ( t - low );
52         }
53     }
54     S[a].insert(t);
55 }
56
57 void dfs1(ll u, ll fa) { //记录了所有子树的size 和 每个节点的重儿子
58     size[u] = 1;
59     for ( auto v:G[u] ) {
60         dfs1(v, u);
61         size[u] += size[v];
62         if ( size[v] > size[son[u]] ) son[u] = v;
63     }
64 }
65
66 void dfs2(ll u,ll fa,bool keep,bool isson){
67     for( auto v:G[u] ) {
68         if( v!=son[u] ){
69             dfs2(v,u,0,0);
70         }
71     }
72     if( son[u] ) {
73         dfs2(son[u],u,1,1);
74     }
75
76     if( keep ) {
77         for( auto v:G[fa] ) {
78             if( u==v ) continue;
79             merge( u, v );
80         }
81
82         if( S[fa].size() < S[u].size() ) S[fa].swap(S[u]),
83         swap(ans[fa],ans[u]);
84         merge( fa, u );
85         rans[fa] = ans[fa];
86     }
87 }

```



```

86 }
87
88 int main() {
89     redirect();
90
91     scanf("%d",&n); f[1] = 1; S[1].insert(1);
92     for(ll i=2;i<=n;i++){
93         scanf("%d",&f[i]);
94         G[ f[i] ].push_back(i); S[i].insert(i);
95     }
96
97     dfs1(1,1);
98     dfs2(1,1,0,0);
99
100    for(ll i=1;i<=n;i++) {
101        printf("%lld\n",rans[ i ]);
102    }
103
104    return 0;
105 }
106
107 /*
108 -----
109 author:dragon_bra
110 -----
111 */

```

STL&杂项

二分（标准）

```

1  /**
2   * struct Interval {
3   *   int start;
4   *   int end;
5   *   Interval(int s, int e) : start(start), end(e) {}
6   * };
7   */
8
9  class Solution {
10 public:
11     /**
12      * 代码中的类名、方法名、参数名已经指定，请勿修改，直接返回方法规定的值即可
13      *
14      * @param n int整型 玩偶数
15      * @param m int整型 区间数
16      * @param intervals Interval类vector 表示区间
17      * @return int整型
18      */
19     static bool cmp(Interval a, Interval b) {
20         return a.start < b.start;
21     }
22
23     int doll(int n, int m, vector<Interval>& intervals) {
24         // write code here
25         long long l = 1, r = n;

```

```

26         while (l <= r) {
27             mid = (l+r) / 2;
28             // check code here
29             if (flag) {
30                 ans = mid; l = mid + 1;
31             }
32             else r = mid - 1;
33         }
34
35         return ans;
36     }
37 };

```

优先队列

```

1  #include<iostream>
2  #include<vector>
3  #include<queue>
4  using namespace std;
5  int tmp[100];
6  struct cmp1{
7      bool operator()(int x,int y)
8      {
9          return x>y;//小的优先级高 ,从小到大排
10     }
11 };
12 struct cmp2{
13     bool operator()(const int x,const int y)
14     {
15         return tmp[x]>tmp[y];
16     }
17 };
18 struct node{
19     int x,y;
20     friend bool operator<(node a,node b)
21     {
22         return a.x>b.x;//按x从小到大排
23     }
24 };
25 priority_queue<int>q1;
26 priority_queue<int,vector<int>,cmp1>q2;
27 priority_queue<int,vector<int>,cmp2>q3;
28 priority_queue<node>q4;
29 int main()
30 {
31     int i,j,k,m,n;
32     int x,y;
33     node a;
34     while(cin>>n)
35     {
36         for(int i=0;i<n;i++)
37         {
38             cin>>a.y>>a.x;
39             q4.push(a);
40         }
41         cout<<endl;
42         while(!q4.empty())

```

```

43     {
44         cout<<q4.top().y<<" "<<q4.top().x<<" "<<endl;
45         q4.pop();
46     }
47     cout<<endl;
48
49     int t;
50     for(i=0;i<n;i++)
51     {
52         cin>>t;
53         q2.push(t);
54     }
55     while(!q2.empty())
56     {
57         cout<<q2.top()<<endl;
58         q2.pop();
59     }
60     cout<<endl;
61 }
62 return 0;
63 }

```

exmu

```

1  #include <cstdio>
2  #include <iostream>
3  #include <algorithm>
4  #include <cmath>
5  #include <cstring>
6  #include <map>
7  #include <set>
8  #include <queue>
9  #include <string>
10 #include <vector>
11 using namespace std;
12 typedef long long ll;
13 typedef unsigned long long ull;
14 const int INF = 0x7fffffff;
15 const int mod = 1e9+7;
16 const double eps = 1e-5;
17 const int N = 1e5+10;
18
19 void redirect() {
20     #ifdef LOCAL
21         freopen("test.txt", "r", stdin);
22         //freopen("out.txt", "w", stdout);
23     #endif
24 }
25 inline ll read() {
26     ll f=1,x=0;char ch;
27     do {ch=getchar(); if(ch=='-') f=-1;} while (ch<'0' || ch>'9');
28     do {x=x*10+ch-'0'; ch=getchar(); } while (ch>='0' && ch<='9');
29     return x*f;
30 }
31
32 int main() {
33     //redirect();

```

```

34     cout<<"Hello world."<<endl;
35 }
36
37 /*
38 -----
39 author:dragon_bra
40 -----
41 */

```

highbit

```

1  int highbit(int x) {
2      // lefttest digit of 1
3      // nearly O(1)
4      union { double a; int b[2]; };
5      a = x;
6      return (b[1] >> 20) - 1023;
7  }
8
9  { // 我爱发明
10     vector<long long> p(32);
11
12     void init() {
13         p[0] = 1;
14         for (int i=1; i<=31; i++) p[i] = p[i-1] * 2;
15     }
16
17     int highbit(int x) {
18         return upper_bound(p.begin(), p.end(), x) - p.begin() - 1;
19     }
20 }

```

LIS (最长上升子序列)

```

1  /*
2   * @ author: dragon_bra
3   * @ email: tommy514@foxmail.com
4   * @ data: 2020-07-25 12:12
5   */
6
7  #include <algorithm>
8  #include <cmath>
9  #include <cstdio>
10 #include <cstdlib>
11 #include <cstring>
12 #include <iostream>
13 #include <sstream>
14 #include <map>
15 #include <set>
16 #include <queue>
17 #include <vector>
18
19 using namespace std;
20
21 typedef long long ll;

```

```

22  const int INF = 0x3f3f3f3f;
23  const int mod = 1e9+7;
24  const double eps = 1e-5;
25  const int N = 1e3 + 10;
26
27  void redirect() {
28      #ifdef LOCAL
29          freopen("in.txt", "r", stdin);
30          freopen("out.txt", "w", stdout);
31      #endif
32  }
33
34  int n, a[N];
35  int f[N];
36
37  int lis(int x) {
38      f[0] = -INF;
39      int s = 0, t;
40      for(int i = 1; i <= n; i++) {
41          t = a[i+x-1];
42          if(t > f[s]) f[++s] = t;
43          else {
44              int l = 1, r = s, m;
45              while(l <= r) {
46                  m = (l+r)/2;
47                  if(t > f[m]) l = m+1;
48                  else r = m-1;
49              }
50              f[l] = t;
51          }
52      }
53      return s;
54  }
55
56  int main() {
57      redirect();
58
59      cin >> n;
60      for (int i = 1; i <= n; i++) {
61          cin >> a[i];
62          a[i+n] = a[i];
63      }
64
65      int mx = 0;
66      for (int i = 1; i <= n; i++) {
67          mx = max(mx, lis(i));
68      }
69
70      cout << n - mx << endl;
71  }

```

Tarjan

```

1
2  void tarjan(int i) {
3      int j;
4      DFN[i] = LOW[i] = ++Dindex;

```

```

5     instack[i]=true;
6     Stap[++Stop]=i;
7     for (edge *e=v[i];e;e=e->next)
8     {
9         j=e->t;
10        if (!DFN[j])
11        {
12            tarjan(j);
13            if (LOW[j]<LOW[i])
14                LOW[i]=LOW[j];
15        }
16        else if (instack[j] && DFN[j]<LOW[i])
17            LOW[i]=DFN[j];
18    }
19    if (DFN[i]==LOW[i])
20    {
21        Bcnt++;
22        do
23        {
24            j=Stap[Stop--];
25            instack[j]=false;
26            Belong[j]=Bcnt;
27        }
28        while (j!=i);
29    }
30 }
31 void solve()
32 {
33     int i;
34     Stop=Bcnt=Dindex=0;
35     memset(DFN,0,sizeof(DFN));
36     for (i=1;i<=N;i++)
37         if (!DFN[i])
38             tarjan(i);
39 }

```

对拍.bat

```

1  :loop
2
3  rand.exe
4  A.exe
5  A2.exe
6
7  fc 1.out baoli.cout
8  if errorlevel==1 pause
9
10 goto loop

```