乘积 product

回顾 Lucas 定理，将 a_i 拆分成 $\lfloor \frac{a_i}{2333} \rfloor$ 和 $a_i \bmod 2333$ 两部分. 因为两部分都小于 2333 ，所以组合数都不会是 2333 倍数.

那么答案转为两部分都不增的子序列个数，记每个数两部分为 x_i, y_i ，有 $O(n^2)$ DP，$f_i = \sum_{j<i, x_j \geq x_i, y_j \geq y_i} f_j$，树状数组可优化成 $O(n \log n)$.

```cpp
#include <cstdio>
const int mod = 998244353,p = 2333;
const int maxn = 3e5 + 10;
int n,ans,a[maxn],b[maxn],f[maxn];
struct fenwick {
    int c[p+5][p+5];
    inline int lowbit(int x) {
        return x & -x;
    }
    inline void add(int x,int y,int v) {
        for (int i = x;i <= p;i += lowbit(i))
            for (int j = y;j <= p;j += lowbit(j)) c[i][j] = (c[i][j]+v)%mod;
    }
    inline int sum(int x,int y) {
        int res(0);
        for (int i = x;i;i -= lowbit(i))
            for (int j = y;j;j -= lowbit(j)) res = (res+c[i][j])%mod;
        return res;
    }
} t;
int main() {
    freopen("product.in","r",stdin);
    freopen("product.out","w",stdout);
    scanf("%d",&n);
    t.add(1,1,1);
    for (int i = 1,x,res;i <= n;i++) {
        scanf("%d",&x);
        ans = (ans+(res = t.sum(p-x/p,p-x%p))-1)%mod;
        t.add(p-x/p,p-x%p,res);
    }
    printf("%d",ans);
    return 0;
}
```

圣诞节(CF755F) christmas

首先 n 个人构成了若干个环，题意可转化为选择 k 条边的两个端点染色

最大直接贪心，偶数环直接全染，奇数环剩下的一个点看 k 剩多少决定染不染

求最小可以发现，如果染了一个环，染完环上所有点是最优的. 所以如果有若干个环大小之和为 k，答案就是 k；否则必须多新开一个环，答案是 $k+1$. 所以就变成了一个背包问题，注意到不同大小的环个数不会超过 $\sqrt{n}$，所以二进制优化多重背包+bitset 即可通过.

```cpp
#include <algorithm>
#include <cstring>
```

```cpp
#include <bitset>
#include <cstdio>
using namespace std;
const int maxn = 2e5 + 10;
int t,n,k,ans,tot,p[maxn],cnt[maxn],sum[maxn];
bitset<maxn> f;
bool v[maxn];
inline void dfs(int now) {
    if (!v[now]) v[now] = true,cnt[tot]++,dfs(p[now]);
}
int main() {
    freopen("christmas.in","r",stdin);
    freopen("christmas.out","w",stdout);
    for (scanf("%d",&t);t--;f.reset(),ans = tot = 0) {
        memset(cnt,0,sizeof cnt);
        memset(sum,0,sizeof sum);
        memset(v,false,sizeof v);
        scanf("%d%d",&n,&k); f[0] = 1;
        for (int i = 1;i <= n;i++) scanf("%d",&p[i]);
        for (int i = 1;i <= n;i++)
            if (!v[i]) { tot++; dfs(i); }
        for (int i = 1;i <= tot;i++) sum[cnt[i]]++;
        for (int i = 1;i <= n;i++) if (sum[i]) {
            for (int j = 1;j < sum[i];sum[i] -= j,j <<= 1) f |= f<<i*j;
            f |= f<<i*sum[i];
        }
        for (int i = 1;i <= tot;i++) ans += cnt[i]/2;
        printf("%d %d",k+!f[k],min(n,min(ans,k)+k));
    }
    return 0;
}
```

Mex 路径(CF1083C) mex

用线段树维护，线段树上 $[l,r]$ 节点表示最短的点权包含 $l...r$ 的路径，合并时判断四个端点是否有其中两个在另外两个构成的路径上，求答案时在线段树上二分.

```cpp
#include <algorithm>
#include <vector>
#include <cstdio>
using namespace std;
const int maxn = 2e5 + 10;
int n,q,p[maxn],fa[maxn],nod[maxn],top[maxn],siz[maxn],son[maxn],dep[maxn];
vector<int> edge[maxn];
inline void dfs1(int now,int f) {
    dep[now] = dep[f]+1;
    siz[now] = 1;
    fa[now] = f;
    for (auto to : edge[now]) if (to ^ f) {
        dfs1(to,now);
        siz[now] += siz[to];
        if (siz[to] > siz[son[now]]) son[now] = to;
    }
}
inline void dfs2(int now,int sum) {
    top[now] = sum;
    if (son[now]) dfs2(son[now],sum);
    for (auto to : edge[now])
        if (to ^ fa[now] && to ^ son[now]) dfs2(to,to);
}
inline int lca(int u,int v) {
    for (;top[u] ^ top[v];u = fa[top[u]])
        if (dep[top[u]] < dep[top[v]]) swap(u,v);
    return dep[u] > dep[v] ? v : u;
}
inline int dis(int u,int v) { return dep[u]+dep[v]-(dep[lca(u,v)]<<1); }
inline bool on(int u,int v,int x) { return dis(u,x)+dis(x,v) == dis(u,v); }
struct seg {
    int u,v;
    inline void merge(seg a,seg b) {
        if (!a.u || !b.u) u = v = 0;
        else if (on(a.u,a.v,b.u) && on(a.u,a.v,b.v)) { u = a.u; v = a.v; }
        else if (on(b.u,b.v,a.u) && on(b.u,b.v,a.v)) { u = b.u; v = b.v; }
        else if (on(b.u,a.v,a.u) && on(b.u,a.v,b.v)) { u = b.u; v = a.v; }
        else if (on(a.u,b.v,a.v) && on(a.u,b.v,b.u)) { u = a.u; v = b.v; }
        else if (on(a.u,b.u,a.v) && on(a.u,b.u,b.v)) { u = a.u; v = b.u; }
        else if (on(a.v,b.v,a.u) && on(a.v,b.v,b.u)) { u = a.v; v = b.v; }
        else u = v = 0;
    }
} mex[maxn<<2];
inline void build(int l,int r,int root) {
    if (l == r) return mex[root] = (seg){ nod[l],nod[l] },void();
    int mid = l+r>>1;
    build(l,mid,root<<1);
    build(mid+1,r,root<<1|1);
    mex[root].merge(mex[root<<1],mex[root<<1|1]);
}
inline void update(int l,int r,int u,int root) {
    if (l > u || r < u) return;
    if (l == r) return mex[root] = (seg){ nod[u],nod[u] },void();
    int mid = l+r>>1;
    update(l,mid,u,root<<1);
    update(mid+1,r,u,root<<1|1);
    mex[root].merge(mex[root<<1],mex[root<<1|1]);
}
inline int query(int l,int r,int root,seg now) {
    if (l == r) {
        now.merge(now,mex[root]);
        return now.u && now.v ? l : l-1;
    }
    seg res; res.merge(now,mex[root<<1]);
    int mid = l+r>>1;
    return res.u && res.v ? query(mid+1,r,root<<1|1,res) : query(l,mid,root<<1,now);
}
int main() {
    freopen("mex.in","r",stdin);
    freopen("mex.out","w",stdout);
    scanf("%d",&n);
    for (int i = 1;i <= n;i++) { scanf("%d",&p[i]); nod[++p[i]] = i; }
    for (int i = 1,u,v;i < n;i++) {
        scanf("%d%d",&u,&v);
        edge[u].push_back(v);
        edge[v].push_back(u);
    }
    dfs1(1,0); dfs2(1,1);
    build(1,n,1);
    for (scanf("%d",&q);q--;) {
        int o,u,v;
        scanf("%d",&o);
        if (o == 2) printf("%d\n",query(1,n,1,(seg){ nod[1],nod[1] }));
        else {
            scanf("%d%d",&u,&v);
            swap(nod[p[u]],nod[p[v]]);
            swap(p[u],p[v]);
            update(1,n,p[u],1);
            update(1,n,p[v],1);
        }
    }
    return 0;
}
```