CF1100F

区间查询线性基。

```
#include <bits/stdc++.h>
#define maxn 500086
using namespace std;

int n, q, x, y;
int b[maxn][31], pos[maxn][31];//debug:数组开小调了一万年

inline void insert(int x, int y){
    int z = y;
    for(int i = 30, j = 1 << 30;j;i--, j >>= 1){
        if(x & j){
            if(!b[y][i]){
                b[y][i] = x, pos[y][i] = z;
                break;
            }else{
                if(z > pos[y][i]) swap(x, b[y][i]), swap(z, pos[y][i]);
                x ^= b[y][i];
            }
        }
    }
}

inline int query(int l, int r){
    int ans = 0;
    for(int i = 30;~i;i--){
        //printf("%d %d %d--\n", i, b[r][i], pos[r][i]);
        if(pos[r][i] >= l && (ans ^ b[r][i]) > ans) ans ^= b[r][i];
    }
    return ans;
}

int main(){
    scanf("%d", &n);
    for(int i = 1;i <= n;i++){
        //for(int j = 0;j <= 30;j++) b[i][j] = b[i - 1][j], pos[i][j] =
pos[i - 1][j];
        memcpy(b[i], b[i - 1], sizeof(b[i - 1])), memcpy(pos[i], pos[i -
1], sizeof(pos[i - 1]));
        scanf("%d", &x);
        insert(x, i);
    }
    scanf("%d", &q);
    while(q--){
        scanf("%d%d", &x, &y);
        printf("%d\n", query(x, y));
```

```
    }
}
```

CF461B

树上连通块，树形dp

```
#include <bits/stdc++.h>
#define maxn 100086
using namespace std;

const int p = 1e9 + 7;

vector<int> v[maxn];
int f[maxn][2];
int ans;

void dfs(int i){
    for(int j = 0;j < v[i].size();j++){
        int to = v[i][j];
        dfs(to);
        f[i][1] = (1ll * f[i][1] * (f[to][0] + f[to][1]) + 1ll * f[i][0] *
f[to][1]) % p;
        f[i][0] = 1ll * f[i][0] * (f[to][0] + f[to][1]) % p;
    }
}

int n, x;

int main(){
    scanf("%d", &n);
    for(int i = 2;i <= n;i++) scanf("%d", &x), v[++x].push_back(i);
    for(int i = 1;i <= n;i++) scanf("%d", &x), f[i][x] = 1;
    dfs(1);
    printf("%d", f[1][1]);
}
```

CF300D

卷积dp

```
#include <bits/stdc++.h>
#define maxn 1086
#define maxm 105
```

```
using namespace std;

const int p = 7340033;

int t;
int n, K;
int dep;
int f[maxn / 10][maxn], g[maxn / 10][maxn];

int main(){
    for(int i = 0;i < maxm;i++) f[i][0] = g[i][0] = 1;
    for(int i = 1;i < maxm;i++){
        for(int j = 1;j < maxn;j++){
            for(int k = 0;k < j;k++){
                f[i][j] += 1ll * g[i - 1][k] * g[i - 1][j - 1 - k] % p;
                if(f[i][j] >= p) f[i][j] -= p;
            }
            for(int k = 0;k <= j;k++){
                g[i][j] += 1ll * f[i][k] * f[i][j - k] % p;
                if(g[i][j] >= p) g[i][j] -= p;
            }
        }
    }
    scanf("%d", &t);
    while(t--){
        scanf("%d%d", &n, &K), dep = 0;
        while((n & 1) && n >= 3) ++dep, n >>= 1;
        printf("%d\n", f[dep][K]);
    }
}
```

CF319C

斜率优化。

```
#include <bits/stdc++.h>
#define maxn 100086
using namespace std;

typedef long long ll;

int n;
ll a[maxn], b[maxn], f[maxn];
int q[maxn], l, r;

inline double k(int i, int j){
    return 1.0 * (f[i] - f[j]) / (b[j] - b[i]);
}
```

```
int main(){
    scanf("%d", &n);
    for(int i = 1;i <= n;i++) scanf("%lld", &a[i]);
    for(int i = 1;i <= n;i++) scanf("%lld", &b[i]);
    f[1] = 0, q[0] = 1;
    for(int i = 2;i <= n;i++){
        while(l < r && k(q[l], q[l + 1]) <= a[i]) ++l;
        f[i] = f[q[l]] + a[i] * b[q[l]];
        //printf("%d %d %lld--\n", i, q[l], f[i]);
        while(l < r && k(q[r], q[r - 1]) >= k(q[r], i)) --r;
        q[++r] = i;
    }
    printf("%lld", f[n]);
}
```

CF455D

数据结构，分块+deque

```
#include <bits/stdc++.h>
#define maxn 100086
#define maxm 320
using namespace std;

int f[maxm][maxn];
int n, sn;
int opt, l, r, k;
int m, x;
deque<int> q[maxm];
int lastans;

int main(){
    scanf("%d", &n), sn = (int)sqrt(n);
    for(int i = 1;i <= n;i++){
        scanf("%d", &x);
        int ii = (i - 1) / sn;
        q[ii].push_back(x), ++f[ii][x];
    }
    scanf("%d", &m);
    while(m--){
        scanf("%d", &opt);
        if(opt == 1){
            scanf("%d%d", &l, &r);
            l = (l + lastans - 1) % n + 1, r = (r + lastans - 1) % n + 1;
            if(l > r) swap(l, r);
            int ll = (l - 1) / sn, rr = (r - 1) / sn, li = (l - 1) % sn, ri
```

```
= (r - 1) % sn;
            //printf("%d %d %d %d--\n", ll, rr, li, ri);
            if(ll == rr){
                int x = q[ll][ri];
                q[ll].erase(q[ll].begin() + ri);
                q[ll].insert(q[ll].begin() + li, x);
            }else{
                int x = q[ll].back();q[ll].pop_back(), --f[ll][x];
                for(int i = ll + 1;i < rr;i++){
                    q[i].push_front(x), ++f[i][x];
                    x = q[i].back(), q[i].pop_back(), --f[i][x];
                }
                q[rr].push_front(x), ++f[rr][x];
                x = q[rr][ri + 1], --f[rr][x], q[rr].erase(q[rr].begin() +
ri + 1);//debug:已经pushfront了所以下标都要+1
                ++f[ll][x], q[ll].insert(q[ll].begin() + li, x);
            }
        }else{
            scanf("%d%d%d", &l, &r, &k);
            l = (l + lastans - 1) % n + 1, r = (r + lastans - 1) % n + 1, k
= (k + lastans - 1) % n + 1;
            if(l > r) swap(l, r);
            int ll = (l - 1) / sn, rr = (r - 1) / sn, li = (l - 1) % sn, ri
= (r - 1) % sn;
            int ans = 0;
            if(ll == rr){
                for(int i = li;i <= ri;i++) ans += q[ll][i] == k;
            }else{
                for(int i = ll + 1;i < rr;i++) ans += f[i][k];
                //printf("%d %d %d %d--\n", ll, rr, li, ri);
                for(int i = li;i < sn;i++) ans += q[ll][i] == k;
                for(int i = 0;i <= ri;i++) ans += q[rr][i] == k;
            }
            printf("%d\n", lastans = ans);
        }
    }
}
```

CF383E

容斥dp+高维前缀和

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

using namespace std;

int n;
int f[1 << 25];
```

```
char s[233];
int ans;

int main(){
    scanf("%d", &n);
    for(int i = 1;i <= n;i++){
        scanf("%s", s + 1);
        int x = 0;
        for(int j = 1;j <= 3;j++){
            x |= 1 << (s[j] - 'a');
        }
        for(int j = x;j;j = (j - 1) & x){
            if(__builtin_popcount(j) & 1) f[j]++;
            else f[j]--;
        }
    }
    for(int i=0;i<24;i++)
        for(int j=0;j<(1<<24);j++)
            if(j&(1<<i))f[j]+=f[j^(1<<i)];
    for(int i=0;i<(1<<24);i++)
        ans=ans^(f[i]*f[i]);
    printf("%d", ans);
}
```

CF1045G

思维+二维数点。

```
#include <bits/stdc++.h>
#define maxn 100086
using namespace std;

struct Node{
    int son[2], val, pri, siz;
}t[maxn];

int cnt;
int root[maxn];


inline int rand()
{
    static int seed=12345;
    return seed=(int)seed*482711LL%2147483647;
}

inline int ls(int x){
```

```
    return t[x].son[0];
}

inline int rs(int x){
    return t[x].son[1];
}

void up(int x){
    t[x].siz = t[ls(x)].siz + t[rs(x)].siz + 1;
}

int newnode(int val){
    cnt++;
    t[cnt].val = val;
    t[cnt].pri = rand();
    t[cnt].siz = 1;
    return cnt;
}

void split(int x, int val, int &a, int &b){//x起始 val权值 a b为两个根编号
    if(!x){
        a = b = 0;
        return;
    }
    if(t[x].val <= val) a = x, split(rs(x), val, t[x].son[1], b);
    else b = x, split(ls(x), val, a, t[x].son[0]);
    up(x);
}

int merge(int x, int y){//返回根编号
    if(x == 0 || y == 0) return x + y;
    if(t[x].pri > t[y].pri){
        t[x].son[1] = merge(rs(x), y);
        up(x);
        return x;
    }else{
        t[y].son[0] = merge(x, ls(y));
        up(y);
        return y;
    }
}

int n, k;

struct N{
    int x, r, q;
}a[maxn];

inline bool cmp(N x, N y){
    return x.r > y.r;
}
```

```
long long ans;

int b[maxn], n0;
int A, B, C, D;

int main(){
    scanf("%d%d", &n, &k);
    for(int i = 1;i <= n;i++) scanf("%d%d%d", &a[i].x, &a[i].r, &a[i].q),
b[i] = a[i].q;
    sort(a + 1, a + 1 + n, cmp);
    sort(b + 1, b + 1 + n);
    n0 = unique(b + 1, b + 1 + n) - (b + 1);
    for(int i = 1;i <= n;i++){
        int x = lower_bound(b + 1, b + 1 + n0, a[i].q) - b;
        for(int j = x;j <= n0 && b[j] - b[x] <= k;j++){
            split(root[j], a[i].x - a[i].r - 1, A, B);
            split(B, a[i].x + a[i].r, C, D);
            ans += t[C].siz;
            root[j] = merge(A, merge(C, D));
        }
        for(int j = x - 1;j && b[x] - b[j] <= k;j--){
            split(root[j], a[i].x - a[i].r - 1, A, B);
            split(B, a[i].x + a[i].r, C, D);
            ans += t[C].siz;
            root[j] = merge(A, merge(C, D));
        }
        split(root[x], a[i].x, A, B);
        root[x] = merge(A, merge(newnode(a[i].x), B));
    }
    printf("%lld", ans);
} //
```

CF797D

CF979D

CF1093G