分块/AcWing 243. 一个简单的整数问题2

```
#include <iostream>
#include <cstring>
#include <algorithm>
#include <cstdio>
#include <cmath>using namespace std;
typedef long long LL;
const int N = 100010, M = 350;
int n, m, len;
LL add[M], sum[M];
int w[N];
int get(int i)
{
    return i / len;
}
void change(int l, int r, int d)
{
    if (get(l) == get(r))  // 段内直接暴力
    {
        for (int i = l; i <= r; i ++ ) w[i] += d, sum[get(i)] += d;
    }
    else
    {
        int i = l, j = r;
        while (get(i) == get(l)) w[i] += d, sum[get(i)] += d, i ++ ;
        while (get(j) == get(r)) w[j] += d, sum[get(j)] += d, j -- ;
        for (int k = get(i); k <= get(j); k ++ ) sum[k] += len * d, ad
    }
}
LL query(int l, int r)
{
    LL res = 0;
    if (get(l) == get(r))  // 段内直接暴力
    {
        for (int i = l; i <= r; i ++ ) res += w[i] + add[get(i)];
    }
    else
    {
        int i = l, j = r;
        while (get(i) == get(l)) res += w[i] + add[get(i)], i ++ ;
        while (get(j) == get(r)) res += w[j] + add[get(j)], j -- ;
        for (int k = get(i); k <= get(j); k ++ ) res += sum[k];
    }
    return res;
}
```

```
int main()
{
    scanf("%d%d", &n, &m);
    len = sqrt(n);
    for (int i = 1; i <= n; i ++ )
    {
        scanf("%d", &w[i]);
        sum[get(i)] += w[i];
    }
    char op[2];
    int l, r, d;
    while (m -- )
    {
        scanf("%s%d%d", op, &l, &r);
        if (*op == 'C')
        {
            scanf("%d", &d);
            change(l, r, d);
        }
        else printf("%lld\n", query(l, r));
    }
    return 0;
}
```

作者：yxc 链接：https://www.acwing.com/activity/content/code/content/489828/ 来源：AcWing

块状链表/AcWing 947. 文本编辑器

```
#include <iostream>
#include <cstdio>
#include <cstring>
#include <algorithm>
using namespace std;
const int N = 2000, M = 2010;
int n, x, y;
struct Node
{
    char s[N + 1];
    int c, l, r;
}p[M];
char str[2000010];
int q[M], tt;  // 内存回收
void move(int k)  // 移到第k个字符后面
```

```
{
    x = p[0].r;
    while (k > p[x].c) k -= p[x].c, x = p[x].r;
    y = k - 1;
}
void add(int x, int u)  // 将节点u插到节点x的右边
{
    p[u].r = p[x].r, p[p[u].r].l = u;
    p[x].r = u, p[u].l = x;
}
void del(int u)  // 删除节点u
{
    p[p[u].l].r = p[u].r;
    p[p[u].r].l = p[u].l;
    p[u].l = p[u].r = p[u].c = 0;  // 清空节点u
    q[ ++ tt] = u;  // 回收节点u
}
void insert(int k)  // 在光标后插入k个字符
{
    if (y < p[x].c - 1)  // 从光标处分裂
    {
        int u = q[tt -- ];  // 新建一个节点
        for (int i = y + 1; i < p[x].c; i ++ )
            p[u].s[p[u].c ++ ] = p[x].s[i];
        p[x].c = y + 1;
        add(x, u);
    }
    int cur = x;
    for (int i = 0; i < k;)
    {
        int u = q[tt -- ];  // 创建一个新的块
        while (p[u].c < N && i < k)
            p[u].s[p[u].c ++ ] = str[i ++ ];
        add(cur, u);
        cur = u;
    }
}
void remove(int k)  // 删除光标后的k个字符
{
    if (p[x].c - 1 - y >= k)  // 节点内删
    {
        for (int i = y + k + 1, j = y + 1; i < p[x].c; i ++, j ++ ) p[
        p[x].c -= k;
    }
    else
    {
        k -= p[x].c - y - 1;  // 删除当前节点的剩余部分
        p[x].c = y + 1;
        while (p[x].r && k >= p[p[x].r].c)
        {
            int u = p[x].r;
            k -= p[u].c;
            del(u);
        }
        int u = p[x].r;  // 删除结尾节点的前半部分
        for (int i = 0, j = k; j < p[u].c; i ++, j ++ ) p[u].s[i] = p[
        p[u].c -= k;
    }
}
void get(int k)  // 返回从光标开始的k个字符
{
    if (p[x].c - 1 - y >= k)  // 节点内返回
    {
        for (int i = 0, j = y + 1; i < k; i ++, j ++ ) putchar(p[x].s[
    }
    else
    {
        k -= p[x].c - y - 1;
        for (int i = y + 1; i < p[x].c; i ++ ) putchar(p[x].s[i]);  //
        int cur = x;
        while (p[cur].r && k >= p[p[cur].r].c)
        {
            int u = p[cur].r;
            for (int i = 0; i < p[u].c; i ++ ) putchar(p[u].s[i]);
            k -= p[u].c;
            cur = u;
        }
        int u = p[cur].r;
        for (int i = 0; i < k; i ++ ) putchar(p[u].s[i]);
    }
    puts("");
}
void prev()  // 光标向前移动一位
{
    if (!y)
    {
        x = p[x].l;
        y = p[x].c - 1;
    }
    else y -- ;
}
void next()  // 光标向后移动一位
{
    if (y < p[x].c - 1) y ++ ;
    else
    {
        x = p[x].r;
        y = 0;
    }
}
void merge()  // 将长度较短的相邻节点合并，保证块状链表时间复杂度的核心
{
    for (int i = p[0].r; i; i = p[i].r)
    {
        while (p[i].r && p[i].c + p[p[i].r].c < N)
        {
            int r = p[i].r;
            for (int j = p[i].c, k = 0; k < p[r].c; j ++, k ++ )
                p[i].s[j] = p[r].s[k];
            if (x == r) x = i, y += p[i].c;  // 更新光标的位置
            p[i].c += p[r].c;
            del(r);
        }
    }
}
int main()
{
    for (int i = 1; i < M; i ++ ) q[ ++ tt] = i;
    scanf("%d", &n);
    char op[10];
    str[0] = '>';
    insert(1);  // 插入哨兵
    move(1);  // 将光标移动到哨兵后面
    while (n -- )
    {
        int a;
        scanf("%s", op);
        if (!strcmp(op, "Move"))
        {
            scanf("%d", &a);
            move(a + 1);
        }
        else if (!strcmp(op, "Insert"))
        {
            scanf("%d", &a);
            int i = 0, k = a;
            while (a)
            {
                str[i] = getchar();
                if (str[i] >= 32 && str[i] <= 126) i ++, a -- ;
            }
            insert(k);
            merge();
        }
        else if (!strcmp(op, "Delete"))
        {
            scanf("%d", &a);
            remove(a);
            merge();
        }
        else if (!strcmp(op, "Get"))
        {
            scanf("%d", &a);
            get(a);
        }
        else if (!strcmp(op, "Prev")) prev();
        else next();
    }
    return 0;
}
```

作者：yxc 链接：https://www.acwing.com/activity/content/code/content/489888/ 来源：AcWing

普通莫队/AcWing 2492. HH的项链

```
#include <iostream>
#include <cstring>
#include <cstdio>
#include <algorithm>
#include <cmath>
using namespace std;
const int N = 50010, M = 200010, S = 1000010;
int n, m, len;
int w[N], ans[M];
struct Query
{
    int id, l, r;
}q[M];
int cnt[S];
int get(int x)
{
    return x / len;
}
bool cmp(const Query& a, const Query& b)
```

```
{
    int i = get(a.l), j = get(b.l);
    if (i != j) return i < j;
    return a.r < b.r;
}
void add(int x, int& res)
{
    if (!cnt[x]) res ++ ;
    cnt[x] ++ ;
}
void del(int x, int& res)
{
    cnt[x] -- ;
    if (!cnt[x]) res -- ;
}
int main()
{
    scanf("%d", &n);
    for (int i = 1; i <= n; i ++ ) scanf("%d", &w[i]);
    scanf("%d", &m);
    len = max(1, (int)sqrt((double)n * n / m));
    for (int i = 0; i < m; i ++ )
    {
        int l, r;
        scanf("%d%d", &l, &r);
        q[i] = {i, l, r};
    }
    sort(q, q + m, cmp);
    for (int k = 0, i = 0, j = 1, res = 0; k < m; k ++ )
    {
        int id = q[k].id, l = q[k].l, r = q[k].r;
        while (i < r) add(w[ ++ i], res);
        while (i > r) del(w[i -- ], res);
        while (j < l) del(w[j ++ ], res);
        while (j > l) add(w[ -- j], res);
        ans[id] = res;
    }
    for (int i = 0; i < m; i ++ ) printf("%d\n", ans[i]);
    return 0;
}
```

作者：yxc 链接：
https://www.acwing.com/activity/content/code/content/499980/ 来源：
AcWing

回滚莫队/AcWing 2523. 历史研究

```
#include <iostream>
#include <cstring>
#include <cstdio>
#include <algorithm>
#include <cmath>
#include <vector>
using namespace std;
typedef long long LL;
const int N = 100010;
int n, m, len;
int w[N], cnt[N];
LL ans[N];
struct Query
{
    int id, l, r;
}q[N];
vector<int> nums;
int get(int x)
{
    return x / len;
}
bool cmp(const Query& a, const Query& b)
{
    int i = get(a.l), j = get(b.l);
    if (i != j) return i < j;
    return a.r < b.r;
}
void add(int x, LL& res)
{
    cnt[x] ++ ;
    res = max(res, (LL)cnt[x] * nums[x]);
}
int main()
{
    scanf("%d%d", &n, &m);
    len = sqrt(n);
    for (int i = 1; i <= n; i ++ ) scanf("%d", &w[i]), nums.push_back(
    sort(nums.begin(), nums.end());
    nums.erase(unique(nums.begin(), nums.end()), nums.end());
    for (int i = 1; i <= n; i ++ )
        w[i] = lower_bound(nums.begin(), nums.end(), w[i]) - nums.begi
    for (int i = 0; i < m; i ++ )
    {
        int l, r;
        scanf("%d%d", &l, &r);
        q[i] = {i, l, r};
    }
    sort(q, q + m, cmp);
    for (int x = 0; x < m;)
    {
        int y = x;
        while (y < m && get(q[y].l) == get(q[x].l)) y ++ ;
        int right = get(q[x].l) * len + len - 1;
        // 暴力求块内的询问
        while (x < y && q[x].r <= right)
        {
            LL res = 0;
            int id = q[x].id, l = q[x].l, r = q[x].r;
            for (int k = l; k <= r; k ++ ) add(w[k], res);
            ans[id] = res;
            for (int k = l; k <= r; k ++ ) cnt[w[k]] -- ;
            x ++ ;
        }
        // 求块外的询问
        LL res = 0;
        int i = right, j = right + 1;
        while (x < y)
        {
            int id = q[x].id, l = q[x].l, r = q[x].r;
            while (i < r) add(w[ ++ i], res);
            LL backup = res;
            while (j > l) add(w[ -- j], res);
            ans[id] = res;
            while (j < right + 1) cnt[w[j ++ ]] -- ;
            res = backup;
            x ++ ;
        }
        memset(cnt, 0, sizeof cnt);
    }
    for (int i = 0; i < m; i ++ ) printf("%lld\n", ans[i]);
    return 0;
}
```

作者：yxc 链接：https://www.acwing.com/activity/content/code/content/500112/ 来源：AcWing

树上莫队/AcWing 2534. 树上计数2

```
#include <iostream>
#include <cstring>
```

```
#include <cstdio>
#include <algorithm>
#include <cmath>
#include <vector>
using namespace std;
const int N = 100010;
int n, m, len;
int w[N];
int h[N], e[N], ne[N], idx;
int depth[N], f[N][16];
int seq[N], top, first[N], last[N];
int cnt[N], st[N], ans[N];
int que[N];
struct Query
{
    int id, l, r, p;
}q[N];
vector<int> nums;
void add_edge(int a, int b)
{
    e[idx] = b, ne[idx] = h[a], h[a] = idx ++ ;
}
void dfs(int u, int father)
{
    seq[ ++ top] = u;
    first[u] = top;
    for (int i = h[u]; ~i; i = ne[i])
    {
        int j = e[i];
        if (j != father) dfs(j, u);
    }
    seq[ ++ top] = u;
    last[u] = top;
}
void bfs()
{
    memset(depth, 0x3f, sizeof depth);
    depth[0] = 0, depth[1] = 1;
    int hh = 0, tt = 0;
    que[0] = 1;
    while (hh <= tt)
    {
        int t = que[hh ++ ];
        for (int i = h[t]; ~i; i = ne[i])
        {
            int j = e[i];
            if (depth[j] > depth[t] + 1)
            {
                depth[j] = depth[t] + 1;
                f[j][0] = t;
                for (int k = 1; k <= 15; k ++ )
                    f[j][k] = f[f[j][k - 1]][k - 1];
                que[ ++ tt] = j;
            }
        }
    }
}
int lca(int a, int b)
{
    if (depth[a] < depth[b]) swap(a, b);
    for (int k = 15; k >= 0; k -- )
        if (depth[f[a][k]] >= depth[b])
            a = f[a][k];
    if (a == b) return a;
    for (int k = 15; k >= 0; k -- )
        if (f[a][k] != f[b][k])
        {
            a = f[a][k];
            b = f[b][k];
        }
    return f[a][0];
}
int get(int x)
{
    return x / len;
}
bool cmp(const Query& a, const Query& b)
{
    int i = get(a.l), j = get(b.l);
    if (i != j) return i < j;
    return a.r < b.r;
}
void add(int x, int& res)
{
    st[x] ^= 1;
    if (st[x] == 0)
    {
        cnt[w[x]] -- ;
        if (!cnt[w[x]]) res -- ;
    }
    else
    {
        if (!cnt[w[x]]) res ++ ;
        cnt[w[x]] ++ ;
    }
}
int main()
{
    scanf("%d%d", &n, &m);
    for (int i = 1; i <= n; i ++ ) scanf("%d", &w[i]), nums.push_back(
    sort(nums.begin(), nums.end());
    nums.erase(unique(nums.begin(), nums.end()), nums.end());
    for (int i = 1; i <= n; i ++ )
        w[i] = lower_bound(nums.begin(), nums.end(), w[i]) - nums.begi
    memset(h, -1, sizeof h);
    for (int i = 0; i < n - 1; i ++ )
    {
        int a, b;
        scanf("%d%d", &a, &b);
        add_edge(a, b), add_edge(b, a);
    }
    dfs(1, -1);
    bfs();
    for (int i = 0; i < m; i ++ )
    {
        int a, b;
        scanf("%d%d", &a, &b);
        if (first[a] > first[b]) swap(a, b);
        int p = lca(a, b);
        if (a == p) q[i] = {i, first[a], first[b]};
        else q[i] = {i, last[a], first[b], p};
    }
    len = sqrt(top);
    sort(q, q + m, cmp);
    for (int i = 0, L = 1, R = 0, res = 0; i < m; i ++ )
    {
        int id = q[i].id, l = q[i].l, r = q[i].r, p = q[i].p;
        while (R < r) add(seq[ ++ R], res);
        while (R > r) add(seq[R -- ], res);
        while (L < l) add(seq[L ++ ], res);
        while (L > l) add(seq[ -- L], res);
        if (p) add(p, res);
        ans[id] = res;
        if (p) add(p, res);
    }
    for (int i = 0; i < m; i ++ ) printf("%d\n", ans[i]);
    return 0;
}
```

作者：yxc 链接：
https://www.acwing.com/activity/content/code/content/504559/ 来源：
AcWing

树状数组/AcWing 242. 一个简单的整数问题

```
#include <cstdio>
#include <cstring>
#include <iostream>
#include <algorithm>
using namespace std;
typedef long long LL;
const int N = 100010;
int n, m;
int a[N];
LL tr[N];
int lowbit(int x)
{
    return x & -x;
}
void add(int x, int c)
{
    for (int i = x; i <= n; i += lowbit(i)) tr[i] += c;
}
LL sum(int x)
{
    LL res = 0;
    for (int i = x; i; i -= lowbit(i)) res += tr[i];
    return res;
}
int main()
{
    scanf("%d%d", &n, &m);
    for (int i = 1; i <= n; i ++ ) scanf("%d", &a[i]);
    for (int i = 1; i <= n; i ++ ) add(i, a[i] - a[i - 1]);
    while (m -- )
    {
        char op[2];
        int l, r, d;
        scanf("%s%d", op, &l);
        if (*op == 'C')
        {
            scanf("%d%d", &r, &d);
            add(l, d), add(r + 1, -d);
        }
        else
        {
            printf("%lld\n", sum(l));
        }
    }
    return 0;
}
```

作者：yxc 链接：https://www.acwing.com/activity/content/code/content/164726/ 来源：AcWing

二叉堆/AcWing 839. 模拟堆

```
#include <iostream>
#include <algorithm>
#include <string.h>
using namespace std;
const int N = 100010;
int h[N], ph[N], hp[N], cnt;
void heap_swap(int a, int b)
{
    swap(ph[hp[a]],ph[hp[b]]);
    swap(hp[a], hp[b]);
    swap(h[a], h[b]);
}
void down(int u)
{
    int t = u;
    if (u * 2 <= cnt && h[u * 2] < h[t]) t = u * 2;
    if (u * 2 + 1 <= cnt && h[u * 2 + 1] < h[t]) t = u * 2 + 1;
    if (u != t)
    {
        heap_swap(u, t);
        down(t);
    }
}
void up(int u)
{
    while (u / 2 && h[u] < h[u / 2])
    {
        heap_swap(u, u / 2);
        u >>= 1;
    }
}
int main()
```

```
{
    int n, m = 0;
    scanf("%d", &n);
    while (n -- )
    {
        char op[5];
        int k, x;
        scanf("%s", op);
        if (!strcmp(op, "I"))
        {
            scanf("%d", &x);
            cnt ++ ;
            m ++ ;
            ph[m] = cnt, hp[cnt] = m;
            h[cnt] = x;
            up(cnt);
        }
        else if (!strcmp(op, "PM")) printf("%d\n", h[1]);
        else if (!strcmp(op, "DM"))
        {
            heap_swap(1, cnt);
            cnt -- ;
            down(1);
        }
        else if (!strcmp(op, "D"))
        {
            scanf("%d", &k);
            k = ph[k];
            heap_swap(k, cnt);
            cnt -- ;
            up(k);
            down(k);
        }
        else
        {
            scanf("%d%d", &k, &x);
            k = ph[k];
            h[k] = x;
            up(k);
            down(k);
        }
    }
    return 0;
}
```

作者：yxc 链接：
https://www.acwing.com/activity/content/code/content/45305/ 来源：
AcWing

左偏树/AcWing 2714. 左偏树

```
#include <iostream>
#include <cstring>
#include <algorithm>
using namespace std;
const int N = 200010;
int n;
int v[N], dist[N], l[N], r[N], idx;
int p[N];
bool cmp(int x, int y)
{
    if (v[x] != v[y]) return v[x] < v[y];
    return x < y;
}
int find(int x)
{
    if (p[x] != x) p[x] = find(p[x]);
    return p[x];
}
int merge(int x, int y)
{
    if (!x || !y) return x + y;
    if (cmp(y, x)) swap(x, y);
    r[x] = merge(r[x], y);
    if (dist[r[x]] > dist[l[x]]) swap(l[x], r[x]);
    dist[x] = dist[r[x]] + 1;
    return x;
}
int main()
{
    scanf("%d", &n);
    v[0] = 2e9;
    while (n -- )
    {
        int t, x, y;
        scanf("%d%d", &t, &x);
        if (t == 1)
        {
            v[ ++ idx] = x;
            dist[idx] = 1;
            p[idx] = idx;
        }
        else if (t == 2)
        {
            scanf("%d", &y);
            x = find(x), y = find(y);
            if (x != y)
            {
                if (cmp(y, x)) swap(x, y);
                p[y] = x;
                merge(x, y);
            }
        }
        else if (t == 3)
        {
            printf("%d\n", v[find(x)]);
        }
        else
        {
            x = find(x);
            if (cmp(r[x], l[x])) swap(l[x], r[x]);
            p[x] = l[x], p[l[x]] = l[x];
            merge(l[x], r[x]);
        }
    }
    return 0;
}
```

作者：yxc 链接：https://www.acwing.com/activity/content/code/content/558830/ 来源：AcWing

ST表/AcWing 1273. 天才的记忆

```
#include <cstdio>
#include <cstring>
#include <algorithm>
#include <cmath>
using namespace std;
const int N = 200010, M = 18;
int n, m;
int w[N];
int f[N][M];
void init()
{
    for (int j = 0; j < M; j ++ )
```

```
        for (int i = 1; i + (1 << j) - 1 <= n; i ++ )
            if (!j) f[i][j] = w[i];
            else f[i][j] = max(f[i][j - 1], f[i + (1 << j - 1)][j - 1]
}
int query(int l, int r)
{
    int len = r - l + 1;
    int k = log(len) / log(2);
    return max(f[l][k], f[r - (1 << k) + 1][k]);
}
int main()
{
    scanf("%d", &n);
    for (int i = 1; i <= n; i ++ ) scanf("%d", &w[i]);
    init();
    scanf("%d", &m);
    while (m -- )
    {
        int l, r;
        scanf("%d%d", &l, &r);
        printf("%d\n", query(l, r));
    }
    return 0;
}
```

作者：yxc 链接：https://www.acwing.com/activity/content/code/content/205231/ 来源：AcWing

单调栈/AcWing 830. 单调栈

```
#include <iostream>
using namespace std;
const int N = 100010;
int stk[N], tt;
int main()
{
    int n;
    cin >> n;
    while (n -- )
    {
        int x;
        scanf("%d", &x);
        while (tt && stk[tt] >= x) tt -- ;
        if (!tt) printf("-1 ");
        else printf("%d ", stk[tt]);
        stk[ ++ tt] = x;
    }
    return 0;
}
```

作者：yxc 链接：
https://www.acwing.com/activity/content/code/content/43105/ 来源：
AcWing

单调队列/AcWing 154. 滑动窗口

```
#include <iostream>
using namespace std;
const int N = 1000010;
int a[N], q[N];
int main()
{
    int n, k;
    scanf("%d%d", &n, &k);
    for (int i = 0; i < n; i ++ ) scanf("%d", &a[i]);
    int hh = 0, tt = -1;
    for (int i = 0; i < n; i ++ )
    {
        if (hh <= tt && i - k + 1 > q[hh]) hh ++ ;
        while (hh <= tt && a[q[tt]] >= a[i]) tt -- ;
        q[ ++ tt] = i;
        if (i >= k - 1) printf("%d ", a[q[hh]]);
    }
    puts("");
    hh = 0, tt = -1;
    for (int i = 0; i < n; i ++ )
    {
        if (hh <= tt && i - k + 1 > q[hh]) hh ++ ;
        while (hh <= tt && a[q[tt]] <= a[i]) tt -- ;
        q[ ++ tt] = i;
        if (i >= k - 1) printf("%d ", a[q[hh]]);
    }
    puts("");
    return 0;
}
```

作者：yxc 链接：
https://www.acwing.com/activity/content/code/content/43107/ 来源：

AcWing

KMP/AcWing 831. KMP字符串

```
#include <iostream>
using namespace std;
const int N = 100010, M = 1000010;
int n, m;
int ne[N];
char s[M], p[N];
int main()
{
    cin >> n >> p + 1 >> m >> s + 1;
    for (int i = 2, j = 0; i <= n; i ++ )
    {
        while (j && p[i] != p[j + 1]) j = ne[j];
        if (p[i] == p[j + 1]) j ++ ;
        ne[i] = j;
    }
    for (int i = 1, j = 0; i <= m; i ++ )
    {
        while (j && s[i] != p[j + 1]) j = ne[j];
        if (s[i] == p[j + 1]) j ++ ;
        if (j == n)
        {
            printf("%d ", i - n);
            j = ne[j];
        }
    }
    return 0;
}
```

作者：yxc 链接：https://www.acwing.com/activity/content/code/content/43108/ 来源：AcWing

AC自动机/AcWing 1285. 单词

```
#include <cstdio>
#include <cstring>
#include <iostream>
#include <algorithm>
using namespace std;
const int N = 1000010;
```

```
int n;
int tr[N][26], f[N], idx;
int q[N], ne[N];
char str[N];
int id[210];
void insert(int x)
{
    int p = 0;
    for (int i = 0; str[i]; i ++ )
    {
        int t = str[i] - 'a';
        if (!tr[p][t]) tr[p][t] = ++ idx;
        p = tr[p][t];
        f[p] ++ ;
    }
    id[x] = p;
}
void build()
{
    int hh = 0, tt = -1;
    for (int i = 0; i < 26; i ++ )
        if (tr[0][i])
            q[ ++ tt] = tr[0][i];
    while (hh <= tt)
    {
        int t = q[hh ++ ];
        for (int i = 0; i < 26; i ++ )
        {
            int &p = tr[t][i];
            if (!p) p = tr[ne[t]][i];
            else
            {
                ne[p] = tr[ne[t]][i];
                q[ ++ tt] = p;
            }
        }
    }
}
int main()
{
    scanf("%d", &n);
    for (int i = 0; i < n; i ++ )
    {
        scanf("%s", str);
        insert(i);
    }
    build();
    for (int i = idx - 1; i >= 0; i -- ) f[ne[q[i]]] += f[q[i]];
    for (int i = 0; i < n; i ++ ) printf("%d\n", f[id[i]]);
    return 0;
}
```

作者：yxc 链接：https://www.acwing.com/activity/content/code/content/169866/ 来源：AcWing

线段树/AcWing 1277. 维护序列

```
#include <cstdio>
#include <cstring>
#include <iostream>
#include <algorithm>
using namespace std;
typedef long long LL;
const int N = 100010;
int n, p, m;
int w[N];
struct Node
{
    int l, r;
    int sum, add, mul;
}tr[N * 4];
void pushup(int u)
{
    tr[u].sum = (tr[u << 1].sum + tr[u << 1 | 1].sum) % p;
}
void eval(Node &t, int add, int mul)
{
    t.sum = ((LL)t.sum * mul + (LL)(t.r - t.l + 1) * add) % p;
    t.mul = (LL)t.mul * mul % p;
    t.add = ((LL)t.add * mul + add) % p;
}
void pushdown(int u)
{
    eval(tr[u << 1], tr[u].add, tr[u].mul);
    eval(tr[u << 1 | 1], tr[u].add, tr[u].mul);
    tr[u].add = 0, tr[u].mul = 1;
}
void build(int u, int l, int r)
{
    if (l == r) tr[u] = {l, r, w[r], 0, 1};
    else
    {
        tr[u] = {l, r, 0, 0, 1};
        int mid = l + r >> 1;
        build(u << 1, l, mid), build(u << 1 | 1, mid + 1, r);
        pushup(u);
    }
}
void modify(int u, int l, int r, int add, int mul)
{
    if (tr[u].l >= l && tr[u].r <= r) eval(tr[u], add, mul);
    else
    {
        pushdown(u);
        int mid = tr[u].l + tr[u].r >> 1;
        if (l <= mid) modify(u << 1, l, r, add, mul);
        if (r > mid) modify(u << 1 | 1, l, r, add, mul);
        pushup(u);
    }
}
int query(int u, int l, int r)
{
    if (tr[u].l >= l && tr[u].r <= r) return tr[u].sum;
    pushdown(u);
    int mid = tr[u].l + tr[u].r >> 1;
    int sum = 0;
    if (l <= mid) sum = query(u << 1, l, r);
    if (r > mid) sum = (sum + query(u << 1 | 1, l, r)) % p;
    return sum;
}
int main()
{
    scanf("%d%d", &n, &p);
    for (int i = 1; i <= n; i ++ ) scanf("%d", &w[i]);
    build(1, 1, n);
    scanf("%d", &m);
    while (m -- )
    {
        int t, l, r, d;
        scanf("%d%d%d", &t, &l, &r);
        if (t == 1)
        {
            scanf("%d", &d);
            modify(1, l, r, 0, d);
        }
        else if (t == 2)
        {
            scanf("%d", &d);
            modify(1, l, r, d, 1);
        }
        else printf("%d\n", query(1, l, r));
    }
    return 0;
}
```

作者：yxc 链接：
https://www.acwing.com/activity/content/code/content/167944/ 来源：
AcWing

主席树/AcWing 255. 第K小数

```
#include <cstdio>
#include <cstring>
#include <iostream>
#include <algorithm>
#include <vector>
using namespace std;
const int N = 100010, M = 10010;
int n, m;
int a[N];
vector<int> nums;
struct Node
{
    int l, r;
    int cnt;
}tr[N * 4 + N * 17];
int root[N], idx;
int find(int x)
{
    return lower_bound(nums.begin(), nums.end(), x) - nums.begin();
}
int build(int l, int r)
{
    int p = ++ idx;
    if (l == r) return p;
    int mid = l + r >> 1;
    tr[p].l = build(l, mid), tr[p].r = build(mid + 1, r);
    return p;
}
int insert(int p, int l, int r, int x)
{
    int q = ++ idx;
    tr[q] = tr[p];
    if (l == r)
    {
        tr[q].cnt ++ ;
        return q;
    }
    int mid = l + r >> 1;
    if (x <= mid) tr[q].l = insert(tr[p].l, l, mid, x);
    else tr[q].r = insert(tr[p].r, mid + 1, r, x);
    tr[q].cnt = tr[tr[q].l].cnt + tr[tr[q].r].cnt;
    return q;
}
int query(int q, int p, int l, int r, int k)
{
    if (l == r) return r;
    int cnt = tr[tr[q].l].cnt - tr[tr[p].l].cnt;
    int mid = l + r >> 1;
    if (k <= cnt) return query(tr[q].l, tr[p].l, l, mid, k);
    else return query(tr[q].r, tr[p].r, mid + 1, r, k - cnt);
}
int main()
{
    scanf("%d%d", &n, &m);
    for (int i = 1; i <= n; i ++ )
    {
        scanf("%d", &a[i]);
        nums.push_back(a[i]);
    }
    sort(nums.begin(), nums.end());
    nums.erase(unique(nums.begin(), nums.end()), nums.end());
    root[0] = build(0, nums.size() - 1);
    for (int i = 1; i <= n; i ++ )
        root[i] = insert(root[i - 1], 0, nums.size() - 1, find(a[i]));
    while (m -- )
    {
        int l, r, k;
        scanf("%d%d%d", &l, &r, &k);
        printf("%d\n", nums[query(root[r], root[l - 1], 0, nums.size()
    }
    return 0;
}
```

作者：yxc 链接：https://www.acwing.com/activity/content/code/content/168534/ 来源：AcWing

树套树/AcWing 2488. 树套树-简单版

```
#include <iostream>
#include <cstdio>
#include <cstring>
#include <algorithm>
#include <set>
using namespace std;
const int N = 50010, M = N * 4, INF = 1e9;
int n, m;
struct Tree
{
    int l, r;
    multiset<int> s;
}tr[M];
int w[N];
void build(int u, int l, int r)
{
    tr[u] = {l, r};
    tr[u].s.insert(-INF), tr[u].s.insert(INF);
    for (int i = l; i <= r; i ++ ) tr[u].s.insert(w[i]);
    if (l == r) return;
    int mid = l + r >> 1;
    build(u << 1, l, mid), build(u << 1 | 1, mid + 1, r);
}
void change(int u, int p, int x)
{
    tr[u].s.erase(tr[u].s.find(w[p]));
    tr[u].s.insert(x);
    if (tr[u].l == tr[u].r) return;
    int mid = tr[u].l + tr[u].r >> 1;
    if (p <= mid) change(u << 1, p, x);
    else change(u << 1 | 1, p, x);
}
int query(int u, int a, int b, int x)
{
    if (tr[u].l >= a && tr[u].r <= b)
    {
        auto it = tr[u].s.lower_bound(x);
        --it;
        return *it;
    }
    int mid = tr[u].l + tr[u].r >> 1, res = -INF;
    if (a <= mid) res = max(res, query(u << 1, a, b, x));
    if (b > mid) res = max(res, query(u << 1 | 1, a, b, x));
    return res;
}
int main()
```

```
{
    scanf("%d%d", &n, &m);
    for (int i = 1; i <= n; i ++ ) scanf("%d", &w[i]);
    build (1, 1, n);
    while (m -- )
    {
        int op, a, b, x;
        scanf("%d", &op);
        if (op == 1)
        {
            scanf("%d%d", &a, &x);
            change(1, a, x);
            w[a] = x;
        }
        else
        {
            scanf("%d%d%d", &a, &b, &x);
            printf("%d\n", query(1, a, b, x));
        }
    }
    return 0;
}
```

作者：yxc 链接：https://www.acwing.com/activity/content/code/content/488897/ 来源：AcWing

Treap/AcWing 253. 普通平衡树

```
#include <cstdio>
#include <cstring>
#include <iostream>
#include <algorithm>
using namespace std;
const int N = 100010, INF = 1e8;
int n;
struct Node
{
    int l, r;
    int key, val;
    int cnt, size;
}tr[N];
int root, idx;
void pushup(int p)
{
    tr[p].size = tr[tr[p].l].size + tr[tr[p].r].size + tr[p].cnt;
}
int get_node(int key)
{
    tr[ ++ idx].key = key;
    tr[idx].val = rand();
    tr[idx].cnt = tr[idx].size = 1;
    return idx;
}
void zig(int &p)    // 右旋
{
    int q = tr[p].l;
    tr[p].l = tr[q].r, tr[q].r = p, p = q;
    pushup(tr[p].r), pushup(p);
}
void zag(int &p)    // 左旋
{
    int q = tr[p].r;
    tr[p].r = tr[q].l, tr[q].l = p, p = q;
    pushup(tr[p].l), pushup(p);
}
void build()
{
    get_node(-INF), get_node(INF);
    root = 1, tr[1].r = 2;
    pushup(root);
    if (tr[1].val < tr[2].val) zag(root);
}
void insert(int &p, int key)
{
    if (!p) p = get_node(key);
    else if (tr[p].key == key) tr[p].cnt ++ ;
    else if (tr[p].key > key)
    {
        insert(tr[p].l, key);
        if (tr[tr[p].l].val > tr[p].val) zig(p);
    }
    else
    {
        insert(tr[p].r, key);
        if (tr[tr[p].r].val > tr[p].val) zag(p);
    }
    pushup(p);
}
void remove(int &p, int key)
{
    if (!p) return;
    if (tr[p].key == key)
    {
        if (tr[p].cnt > 1) tr[p].cnt -- ;
        else if (tr[p].l || tr[p].r)
        {
            if (!tr[p].r || tr[tr[p].l].val > tr[tr[p].r].val)
            {
                zig(p);
                remove(tr[p].r, key);
            }
            else
            {
                zag(p);
                remove(tr[p].l, key);
            }
        }
        else p = 0;
    }
    else if (tr[p].key > key) remove(tr[p].l, key);
    else remove(tr[p].r, key);
    pushup(p);
}
int get_rank_by_key(int p, int key)    // 通过数值找排名
{
    if (!p) return 0;   // 本题中不会发生此情况
    if (tr[p].key == key) return tr[tr[p].l].size + 1;
    if (tr[p].key > key) return get_rank_by_key(tr[p].l, key);
    return tr[tr[p].l].size + tr[p].cnt + get_rank_by_key(tr[p].r, key
}
int get_key_by_rank(int p, int rank)   // 通过排名找数值
{
    if (!p) return INF;     // 本题中不会发生此情况
    if (tr[tr[p].l].size >= rank) return get_key_by_rank(tr[p].l, rank
    if (tr[tr[p].l].size + tr[p].cnt >= rank) return tr[p].key;
    return get_key_by_rank(tr[p].r, rank - tr[tr[p].l].size - tr[p].cn
}
int get_prev(int p, int key)   // 找到严格小于key的最大数
{
    if (!p) return -INF;
    if (tr[p].key >= key) return get_prev(tr[p].l, key);
    return max(tr[p].key, get_prev(tr[p].r, key));
}
int get_next(int p, int key)    // 找到严格大于key的最小数
{
    if (!p) return INF;
    if (tr[p].key <= key) return get_next(tr[p].r, key);
    return min(tr[p].key, get_next(tr[p].l, key));
```

```
}
int main()
{
    build();
    scanf("%d", &n);
    while (n -- )
    {
        int opt, x;
        scanf("%d%d", &opt, &x);
        if (opt == 1) insert(root, x);
        else if (opt == 2) remove(root, x);
        else if (opt == 3) printf("%d\n", get_rank_by_key(root, x) - 1
        else if (opt == 4) printf("%d\n", get_key_by_rank(root, x + 1)
        else if (opt == 5) printf("%d\n", get_prev(root, x));
        else printf("%d\n", get_next(root, x));
    }
    return 0;
}
```

作者：yxc 链接：https://www.acwing.com/activity/content/code/content/168876/ 来源：AcWing

Splay/AcWing 2437. Splay

```
#include <cstdio>
#include <cstring>
#include <iostream>
#include <algorithm>
using namespace std;
const int N = 100010;
int n, m;
struct Node
{
    int s[2], p, v;
    int size, flag;
    void init(int _v, int _p)
    {
        v = _v, p = _p;
        size = 1;
    }
}tr[N];
int root, idx;
void pushup(int x)
{
    tr[x].size = tr[tr[x].s[0]].size + tr[tr[x].s[1]].size + 1;
}
void pushdown(int x)
{
    if (tr[x].flag)
    {
        swap(tr[x].s[0], tr[x].s[1]);
        tr[tr[x].s[0]].flag ^= 1;
        tr[tr[x].s[1]].flag ^= 1;
        tr[x].flag = 0;
    }
}
void rotate(int x)
{
    int y = tr[x].p, z = tr[y].p;
    int k = tr[y].s[1] == x;  // k=0表示x是y的左儿子；k=1表示x是y的右儿
    tr[z].s[tr[z].s[1] == y] = x, tr[x].p = z;
    tr[y].s[k] = tr[x].s[k ^ 1], tr[tr[x].s[k ^ 1]].p = y;
    tr[x].s[k ^ 1] = y, tr[y].p = x;
    pushup(y), pushup(x);
}
void splay(int x, int k)
{
    while (tr[x].p != k)
    {
        int y = tr[x].p, z = tr[y].p;
        if (z != k)
            if ((tr[y].s[1] == x) ^ (tr[z].s[1] == y)) rotate(x);
            else rotate(y);
        rotate(x);
    }
    if (!k) root = x;
}
void insert(int v)
{
    int u = root, p = 0;
    while (u) p = u, u = tr[u].s[v > tr[u].v];
    u = ++ idx;
    if (p) tr[p].s[v > tr[p].v] = u;
    tr[u].init(v, p);
    splay(u, 0);
}
int get_k(int k)
{
    int u = root;
    while (true)
    {
        pushdown(u);
        if (tr[tr[u].s[0]].size >= k) u = tr[u].s[0];
        else if (tr[tr[u].s[0]].size + 1 == k) return u;
        else k -= tr[tr[u].s[0]].size + 1, u = tr[u].s[1];
    }
    return -1;
}
void output(int u)
{
    pushdown(u);
    if (tr[u].s[0]) output(tr[u].s[0]);
    if (tr[u].v >= 1 && tr[u].v <= n) printf("%d ", tr[u].v);
    if (tr[u].s[1]) output(tr[u].s[1]);
}
int main()
{
    scanf("%d%d", &n, &m);
    for (int i = 0; i <= n + 1; i ++ ) insert(i);
    while (m -- )
    {
        int l, r;
        scanf("%d%d", &l, &r);
        l = get_k(l), r = get_k(r + 2);
        splay(l, 0), splay(r, l);
        tr[tr[r].s[0]].flag ^= 1;
    }
    output(root);
    return 0;
}
```

作者：yxc 链接：https://www.acwing.com/activity/content/code/content/478319/ 来源：AcWing