3月22日第1次课笔记

- 非降序序列二分查找等于 x 的数下标

```
int find(int x, int l, int r) {
    while(l < r) {
        int mid = (l + r) / 2;
        if(x <= a[mid]) r = mid;
        else l = mid + 1;
    }
    return l;
}
```

- 非降序可重序列下标最小$\geq x$ 的元素

```
int find(int x, int l, int r) {
    while(l < r) {
        int mid = (l + r) / 2;
        if(a[mid] >= x) r = mid;
        else l = mid + 1;
    }
    return l;
}
```

- STL 的使用

```
vector<int> a(n);
int id = lower_bound(a.begin(), a.end(), x) - a.begin();
int id = upper_bound(a.begin(), a.end(), x) - a.begin();

int a[N];
int id = lower_bound(a + 1, a + n + 1, x) - a;
int id = upper_bound(a + 1, a + n + 1, x) - a;
```

- a中元素 x 出现的次数 $\mathcal{O}(\log n)$

```
int count = upper_bound(a.begin(),a.end(),x) -
lower_bound(a.begin(),a.end(),x);
```

- A - B 数对 code

```
#include <bits/stdc++.h>
using namespace std;
const int N = 2e5 + 10;
int a[N];
signed main() {
    ios::sync_with_stdio(false);
    cin.tie(NULL);
    int n, C;
    cin >> n >> C;
    for(int i = 1; i <= n; ++ i) cin >> a[i];
    sort(a + 1, a + n + 1);
```

```
    long long ans = 0;
    for(int i = 1; i <= n; ++ i) {
        ans += upper_bound(a + 1, a + n + 1, a[i] - c) - lower_bound(a + 1, a
+ n + 1, a[i] - c);

    }
    cout << ans << endl;
    return 0;
}
```

洛谷首页 https://www.luogu.com.cn/user/88735