

4.二分

一些参考文章

- [极客时间](#)
- https://blog.csdn.net/qq_41221520/article/details/108277801

两种模板、几点区分

- 两种情况

```
1 闭区间
2 <=
3 mid + 1, mid - 1;
4
5 int binarySearch(vector<int> nums, int target){
6     //闭区间[]
7     int left = 0;
8     int right = nums.size() - 1;
9
10    // <=
11    while(left <= right){
12        int mid = left + (right - left) / 2;
13        if(nums[mid] == target) return mid;
14        else if(nums[mid] < target) left = mid + 1;
15        else if(nums[mid] > target) right = mid - 1;
16    }
17    return -1
18 }
```

```
1 开区间
2 <
3 第一个mid, 第二个mid + 1/ mid - 1
4
5 int binarySearch(vector<int> nums, int target){
6     int left = 0;
7     int right = nums.size();
8     while(left < right){
9         int mid = left + (right - left) / 2;
10        if(nums[mid] == target) return mid;
11        else if(nums[mid] < target) left = mid;
12        else if(nums[mid] > target) right = mid - 1;
13
14        else if(nums[mid] > target) right = mid;
15        else if(nums[mid] < target) left = mid + 1;
16    }
17 }
```

二分查找(找一个确定的值)

基本

```
1 int binarySearch(vector<int> nums, int target){
2     int l = 0;
3     int r = nums.size() - 1;
4     while(l <= r){
5         int mid = l + (r - l) / 2;
6         if(nums[mid] == target){
7             return ans;
8         }else if(mid > target) {
9             r = mid - 1;
10        }else if{
11            l = mid + 1;
12        }
13    }
14    return -1;
15 }
```

左边界

```
1 int binarySearch(vector<int> nums, int target){
2     int ans = -1;
3     int l = 0;
4     int r = nums.size() - 1;
5     while(l <= r){
6         int mid = l + (r - l) / 2;
7         if(nums[mid] == target){
8             r = mid - 1;
9             ans = mid;
10        }else if(nums[mid] > target) {
11            r = mid - 1;
12        }else if(nums[mid] < target){
13            l = mid + 1;
14        }
15    }
16    return ans;
17 }
```

右边界

```
1 int binarySearch(vector<int> nums, int target){
2     int ans = -1;
3     int l = 0;
4     int r = nums.size() - 1;
5     while(l <= r){
6         int mid = l + (r - l) / 2;
7         if(nums[mid] == target){
8             l = mid + 1;
9             ans = mid;
10        }else if(nums[mid] > target) {
11            r = mid - 1;
12        }else if(nums[mid] < target){
13            l = mid + 1;
14        }
15    }
16    return ans;
17 }
```

```

14         }
15     }
16     return ans;
17 }

```

STL中的模板库

```

1 //第一个位置
2 lower_bound(a.begin(), a.end(), target) - a.begin();
3 //找第一个大于等于target的位置
4 upper_bound(a.begin(), a.end(), target) - a.begin();
5 //找第一个大于target的位置
6
7
8 //最后一个位置
9 lower_bound(a.begin(), a.end(), target + 1) - a.begin();
10 //找第一个大于等于target+1的位置，然后-1
11
12 在从小到大的排序数组中，
13
14 lower_bound( begin,end,num)：从数组的begin位置到end-1位置二分查找第一个大于或等于num的
15
16 upper_bound( begin,end,num)：从数组的begin位置到end-1位置二分查找第一个大于num的数字，
17
18 在从大到小的排序数组中，重载lower_bound()和upper_bound()
19
20 lower_bound( begin,end,num,greater<type>() ):从数组的begin位置到end-1位置二分查找第-
21
22 upper_bound( begin,end,num,greater<type>() ):从数组的begin位置到end-1位置二分查找第-

```

二分答案(找一个可能满足的值)

```

1 //插入的位置、第一个>=target的位置
2 int searchInsert(vector<int>& nums, int target) {
3     int n = nums.size();
4     int left = 0, right = n - 1;
5     while(left <= right){
6         int mid = (right - left) / 2 + left;
7         if(nums[mid] == target) return mid;
8         else if(nums[mid] > target){
9             right = mid - 1;
10        }
11        else{
12            left = mid + 1;
13        }
14    }
15    return left;
16    //return right + 1 (退出条件是left = right + 1)
17    //如果while(left < right) return left/right都可以
18    //>=插入位置时候返回left，像69题向下取整意思时候返回right
19 }

```

```
1 //某种要具体check里面算的逻辑
2 int binarySearch(vector<int> nums, int target){
3     int l = 0;
4     int r = nums.size() - 1;
5     int ans = 0;
6     while(l <= r){
7         int mid = l + (r - l) / 2;
8         if(!check()){
9             l = mid + 1;
10        }
11        else if(check()){
12            ans = mid;
13            r = mid - 1;
14        }
15    }
16    return ans;
17 }
```

二分查找题目

- [剑指 Offer II 069. 山峰数组的顶部](#)
- [240. 搜索二维矩阵 II](#)

二分答案题目 275、69、410、441

- [475. 供暖器](#)
- [剑指 Offer II 068. 查找插入位置](#)
- [剑指 Offer II 071. 按权重生成随机数](#)

二分题目

[33. 搜索旋转排序数组](#)

[153. 寻找旋转排序数组中的最小值](#)

[162. 寻找峰值](#)