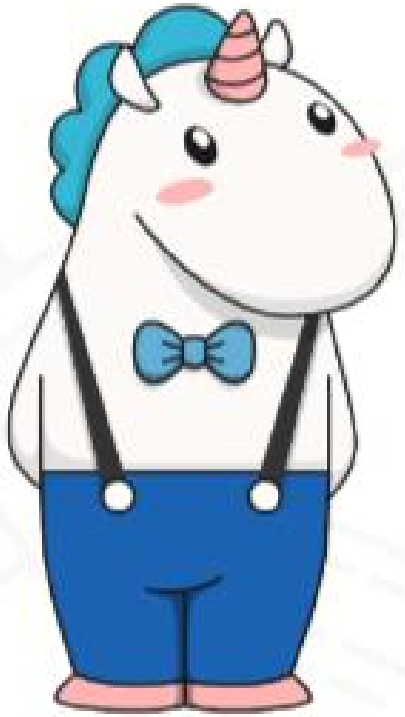
二分查找

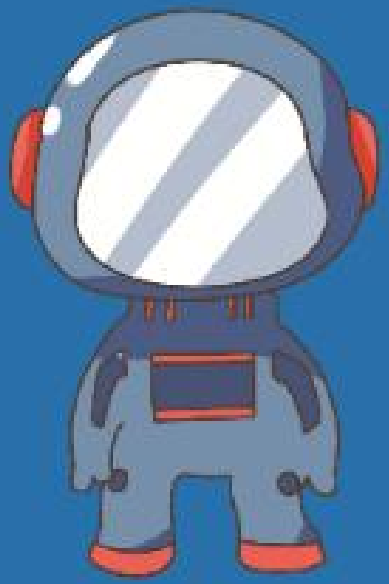
学习目标



1、了解二分法的基本概念

2、掌握二分查找的基本框架

3、了解二分查找的简单题型



知识讲授

二分查找的概念

- 二分查找又称为折半查找，主要用于查找一个有序数组中某一个数的位置。主要思想如下：

v 在一个**有序**数组中，取数组的**中间值**与要查找的数进行比较；

v 若要查找的数等于中间值，查找成功。

二分查找的步骤

1. 若要查找的数大于中间值，则在右半区间继续取中间值与要查找的数进行比较；
2. 若要查找的数小于中间值，则在左半区间继续取中间值与要查找的数进行比较；
3. 直至最后要查找的数未出现过与中间值相等的情况，查找失败

二分查找基本框架

```
int Search(int a[] , int n, int key){
    int low = 1;//左边界从1开始
    int high = n;//右边界从n开始
    while(low <= high) {//[low,high]是查找范围，存在范围则查找
            int mid = low + ((high-low)/2);    //中间下标
            if(key == a[mid])     return mid;//相等代表找到
            else if(key < a[mid])  high = mid - 1;
                                    //key比mid小，mid-1就是右边界
            else  low = mid + 1;
                                    //key比mid，mid+1就是左边界
            }
             return -1;//如果都找不到
} //适用于等值查找
```

使用自定义函数的方法，需要引入的三个参数分别是整个数组，数组长度，查找值。返回的是查找值在数组中的位置。

二分查找的优势

二分查找因为每次查找都会把数据规模折半，所以效率相对较高。如果使用普通的查找可能会消耗太多的时间。

大家可以试试看，在1-100之间随便设定一个数字，只需要最多最多7次肯定能猜对，每次问的都是这个数字和范围中间的数字比大小，每次比完都能去掉一半的数字。

示例1

在有序数组中查找某个数，找到返回数的下标，不存在重复的值，没有返回-1。

参考代码

```
#include<iostream>
using namespace std;
int Search(int a[] , int n, int key){
    int low = 1;
    int high = n;
    while(low <= high) {
            int mid = low + ((high-low)/2);
            if(key == a[mid])   return mid;
            else if(key < a[mid])high = mid - 1;
            else low = mid + 1;
    }
    return -1;
}
```

```
int main()
{
    int s[100000],n,m,b;
    cin>>n>>m;
    for(int i=1;i<=n;i++)cin>>s[i];
    for(int i=1;i<=m;i++)
    {
            cin>>b;
            cout<<Search(s,n,b)<<endl;
    }
    return 0;
}
```

此模板主要用于查找大范围，且有明确数值的元素位置

示例2

从一个有序的整数序列中查找第一个大于整数k的数，如果存在输出出现位置，否则输出-1。序列有重复元素，并且单调递增。

参考代码

```
#include<iostream>
using namespace std;
int Search(int a[] , int n, int key){
    int low = 1;
    int high = n;
    while(low <= high) {
            int mid = low + ((high-low)/2);
            if(key < a[mid])high = mid - 1;
            else low = mid + 1;//包括大于和相等的情况
    }
    if(low<=n) return low;
    //判断左边界是否越界，越界则表示查找不到
    else return -1;
}
```

```
int main()
{
    int s[100000],n,m,b;
    cin>>n>>m;
    for(int i=1;i<=n;i++)cin>>s[i];
    for(int i=1;i<=m;i++)
    {
            cin>>b;
            cout<<Search(s,n,b)<<endl;
    }
    return 0;
}
```

如果相等，左边界仍然会收缩一个单位，最终左边界停在第一个大于这个数的数字上

示例3

从一个有序的整数序列中查找第一个大于或等于整数k的数，如果存在输出出现位置，否则输出-1。序列有重复元素，并且单调递增。

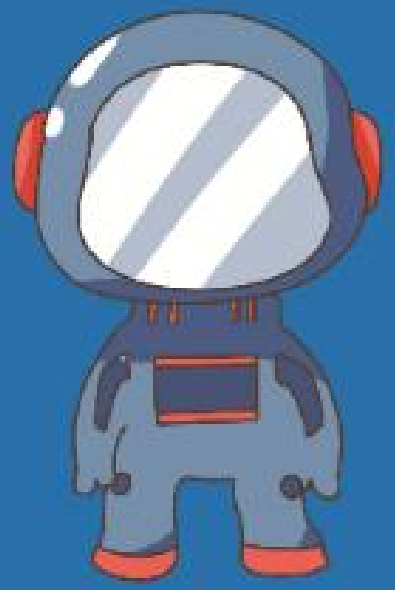
参考代码

```
#include<iostream>
using namespace std;
int Search(int a[] , int n, int key){
    int low = 1;
    int high = n;
    while(low <= high) {
            int mid = low + ((high-low)/2);
            if(key <= a[mid])   high = mid - 1;
            //相等的情况high会一直左移直到错位
            else low = mid + 1;
    }
    if(low<=n) return low;
    else return -1;
}
```

```
int main()
{
    int s[100000],n,m,b;
    cin>>n>>m;
    for(int i=1;i<=n;i++)cin>>s[i];
    for(int i=1;i<=m;i++)
    {
            cin>>b;
            cout<<Search(s,n,b)<<endl;
    }
    return 0;
}
```

low和high错位时，查找结束。low=high+1。最终high指向最后一个小于key的数。

思考：相等时为什么不能直接返回这个位置？



课堂练习

查找特定元素第一次出现的位置

【描述】 从一个有序的整数序列中查找整数k，如果存在输出第一次出现位置，否则输出-1。序列有重复元素，并且单调递增。

【输入】第一行是两个整数n和m；n为序列中整数的个数，m为询问次数；第二行是n个递增的整数；第三行是m个整数，为查找的目标；

【输出】m行；m个查询结果。

【样例输入】

10 3

1 2 3 3 3 4 4 6 6 6

3 5 6

样例输出】

3

-1

8

【提示】参考查找第一个大于等于key的元素的程序，需要修改什么？

参考代码

```
#include<iostream>
using namespace std;
int Search(int a[] , int n, int key){
    int low = 1;
    int high = n;
    while(low <= high) {
            int mid = low + ((high-low)/2);
            if(key <= a[mid])high = mid - 1;
            else low = mid + 1;
    }
    if(low<=n&&a[low] == key)return low;
//增加条件
    else return -1;
}
```

```
int main()
{
    int s[100000],n,m,b;
    cin>>n>>m;
    for(int i=1;i<=n;i++)cin>>s[i];
    for(int i=1;i<=m;i++)
    {
            cin>>b;
            cout<<Search(s,n,b)<<endl;
    }
    return 0;
}
```

和寻找第一个大于等于某数字方法比，增加一个条件，就是确定左边界必须和找的数字一样

查找特定元素最后一次出现的位置

【描述】 从一个有序的整数序列中查找整数k，如果存在输出最后一次出现位置，否则输出-1。序列有重复元素，并且单调递增。

【输入】第一行是两个整数n和m；n为序列中整数的个数，m为询问次数；第二行是n个递增的整数；第三行是m个整数，为查找的目标；

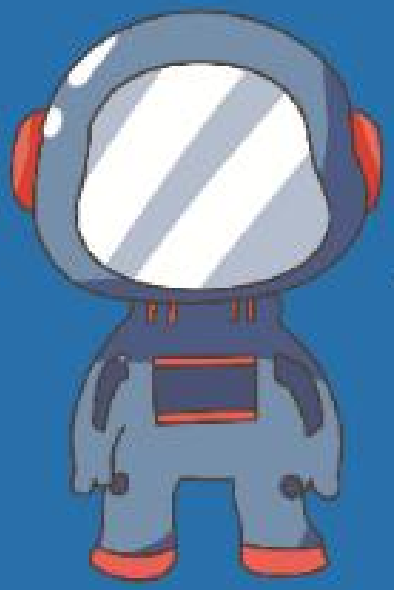
【输出】m行；m个查询结果。

【样例输入】
10 3
1 2 3 3 3 4 4 6 6 6
3 5 6
样例输出】
5
-1
10
【提示】参考查找第一个大于key的元素的程序，需要修改什么？

参考代码

```
#include<iostream>
using namespace std;
int Search(int a[] , int n, int key){
    int low = 1;
    int high = n;
    while(low <= high) {
            int mid = low + ((high-low)/2);
            if(key < a[mid])high = mid - 1;
            else low = mid + 1;
    }
    if(low-1>=1&&a[low-1] == key)return low-1;
    //找到low的位置向前一个
    else return -1;
}
```

```
int main()
{
    int s[100000],n,m,b;
    cin>>n>>m;
    for(int i=1;i<=n;i++)cin>>s[i];
    for(int i=1;i<=m;i++)
    {
            cin>>b;
            cout<<Search(s,n,b)<<endl;
    }
    return 0;
}
```



课堂练习

小结

1. 二分查找基本思路是什么？

2. 二分查找怎么找某一个不重复数字的位置？

3. 怎么知道一个数字重复出现多少次？

作业

1、查找最接近的元素　http://noi.openjudge.cn/ch0111/01/

2、二分法求函数的零点　http://noi.openjudge.cn/ch0111/02/
提示：浮点数数据；hight-low<精确度时则停止搜索即可；

3、不重复地输出数　http://noi.openjudge.cn/ch0111/08/
因为本题数据不强，所以顺序查找也能过。

4、和为给定数　http://noi.openjudge.cn/ch0111/07/
提示：先从小到大排序。设和为sum，顺序搜索第一个数x，二分查找sum-x，如果sun-x存在那么就break；O(nlogn)的算法复杂度，可以应付十万规模数据。