二分查找

TEACHING COURSEWARE POWPOINT

授课时间：2025.08.04

目录

01

概念引入

TEACHING
COURSEWARE

TEACH

概念引入

有一个1-100的数字，现在开始猜

02

概念分析

TEACHING
COURSEWARE

TEACH

概念分析

概念
二分查找是一种高效的查找算法，适用于有序数组（或有序序列）。其核心思想是通过不断将查找区间一分为二，缩小目标元素的可能范围，直到找到目标或确定目标不存在。

03

经典例题

TEACHING
COURSEWARE

TEACH

经典例题

题目描述

复制 Markdown 展开 进入 IDE 模式

输入 n 个不超过 10^9 的单调不减的（就是后面的数字不小于前面的数字）非负整数 $a_1, a_2, \ldots, a_n$，然后进行 m 次询问。对于每次询问，给出一个整数 q，要求输出这个数字在序列中第一次出现的编号，如果没有找到的话输出 -1 。

输入格式

第一行 2 个整数 n 和 m，表示数字个数和询问次数。

第二行 n 个整数，表示这些待查询的数字。

第三行 m 个整数，表示询问这些数字的编号，从 1 开始编号。

输出格式

输出一行，m 个整数，以空格隔开，表示答案。

输入输出样例

输入 #1 复制

```
11 3
1 3 3 3 5 7 9 11 13 15 15
1 3 6
```

输出 #1 复制

```
1 2 -1
```

说明/提示

数据保证，$1 \le n \le 10^6$，$0 \le a_i, q \le 10^9$，$1 \le m \le 10^5$

本题输入输出量较大，请使用较快的 IO 方式。

经典例题

解题思路
核心算法：二分查找
利用数组 "单调不减" 的特性，通过二分查找快速定位目标值。
关键是找到第一个等于目标值的位置，而非任意位置。
查找逻辑：
初始化左右边界 l=1，r=n，答案 ans=-1（默认未找到）。
当 l ≤ r 时，计算中间位置 mid：
若 a[mid] ≥ x：说明目标可能在左半区间（包括 mid），需向左收缩范围（r=mid-1）。
若 a[mid] == x，则更新 ans=mid（记录当前找到的位置）。
若 a[mid] < x：说明目标在右半区间，向右收缩范围（l=mid+1）。
最终 ans 即为第一个等于 x 的位置，若始终未找到则保持 -1。

经典例题

轮次	l	r	mid	a [mid] 值	条件判断（a [mid] >= 3？）	ans 变化	r/l 调整	范围变化
1	1	11	6	7	是（7≥3）	ans 仍为 - 1（7≠3）	r=6-1=5	[1,5]
2	1	5	3	3	是（3≥3）	ans=3（3==3）	r=3-1=2	[1,2]
3	1	2	1	1	否（1<3）	ans 不变	l=1+1=2	[2,2]
4	2	2	2	3	是（3≥3）	ans=2（3==3）	r=2-1=1	[2,1]（循环结束）

```cpp
#include<iostream>
using namespace std;
const int N=1e6+10;
int a[N];
int main()
{
    int n,m;
    cin>>n>>m;
    for(int i=1;i<=n;i++)
    {
        cin>>a[i];
    }
while(m--)
{
    int x;
    cin>>x;
    int l = 1;
    int r = n ;
    int ans = -1;  // 初始化为-1（未找到状态）
    while(l<=r)
    {
        int mid = (l+r)/2;
        if(a[mid] >= x)
        {
            // 先记录可能的位置
            if(a[mid] == x)  // 只有当相等时才更新有效答案
                ans = mid;
            r = mid - 1;  // 继续向左查找更早出现的位置
        }
        else
            l = mid + 1;  // 当前值小于目标，向右查找
    }
    // 输出结果，未找到则保持-1
    cout<<ans<<" ";
}
return 0;
```

04

总结

TEACHING
COURSEWARE

TEACH