



实验舱  
青少年编程  
走近科学 走进名校

# 挑战信息学奥林匹克

## C++程序设计 (8)

### 二分查找

# 查找

- 顺序查找（从数组第一个元素开始向后查找）

- ◆ 查找特定的值
- ◆ 查找最大值
- ◆ 查找最小值



特点：

1. 数据不需要排序
2. 查找的效率低

- 二分查找

---

# 二分查找

1	2	3	4	5	6	7	8	9	10	11
5	13	19	21	37	56	64	75	80	88	92

L=1

R=11


```
mid = (L + R) / 2;  
if (a[mid] == x) return mid;  
if (x < a[mid]) R = mid - 1;  
else L = mid + 1;
```

• 查找21

L	R	mid
1	11	6
1	5	3
4	5	4

# 二分查找

二分查找的数据必须是有序的！！！！



1	2	3	4	5	6	7	8	9	10	11
5	13	19	21	37	56	64	75	80	88	92

L=1

R=11

```
mid = (L + R) / 2;  
if (a[mid] == x) return mid;  
if (x < a[mid]) R = mid - 1;  
else L = mid + 1;
```

• 查找85

L	R	mid
1	11	6
7	11	9
10	11	10
10	10	10
10	9	

R < L时，循环结束。

## 二分查找算法框架（1）

```
int bfind_1(int L, int R, int x)
{
    while ( L <= R )
    {
        int mid = ( L + R ) / 2;
        if ( x == a[mid] ) return mid;
        if ( x < a[mid] ) R = mid - 1;
        else L = mid + 1;
    }
    return -1;
}
```

查到x返回下标

查不到x返回-1



## 二分查找算法框架（2）

```
int bfind_R(int L, int R, int x)
{
    while ( L <= R )
    {
        int mid = ( L + R ) / 2;
        if ( x < a[mid] ) R = mid - 1;
        else L = mid + 1;
    }
    return R;
}
```



返回小于x的最大值

# 例题-1：查找最接近的元素

## 描述

在一个非降序列中，查找与给定值最接近的元素。

## 输入

第一行包含一个整数  $n$ ，为非降序列长度。  $1 \leq n \leq 100000$ 。

第二行包含  $n$  个整数，为非降序列各元素。所有元素的大小均在  $0 \sim 10^9$  之间。

第三行包含一个整数  $m$ ，为要询问的给定值个数。  $1 \leq m \leq 10000$ 。

接下来  $m$  行，每行一个整数，为要询问最接近元素的给定值。所有给定值的大小均在  $0 \sim 10^9$  之间。

## 输出

$m$  行，每行一个整数，为最接近相应给定值的元素值，保持输入顺序。

若有多个值满足条件，输出最小的一个。

## 样例输入

```
3
2 5 8
2
10
5
```

## 样例输出

```
8
5
```

# 算法分析

1	2	3	4	5	6	7	8	9	10	11
5	13	19	21	37	56	64	75	80	88	92

L=1 R=11

```
int bfind_R(int L, int R, int x)
{
    while ( L <= R )
    {
        int mid = ( L + R ) / 2;
        if ( x < a[mid] ) R = mid - 1;
        else L = mid + 1;
    }
    return R;
}
```

t = bfind\_R(1, n, x);

1. x比最小值还小, t = 0;
2. x比最大值还大, t = n;
3. x不存在, a[t] <= x ;



## 参考代码

```
for ( int i = 1; i <= m; i++ )
{
    cin >> x;
    t = bfind_R(1, n, x);
    if (t == n || t != 0 && x - a[t] <= a[t+1] - x )
    {
        cout << a[t] << endl;
    }
    else
    {
        cout << a[t+1] << endl;
    }
}
```

---



# 库函数 `sort()`

## ■ 用法

`sort(a + 1, a + n + 1);` //数组下标从1开始

`sort(a, a + n);` //数组下标从0开始

## ◆ 样例说明

- `a`是数组名称;

- 函数将数组`a`的`n`个元素从小到大排序，默认是升序排序;



## 例题-2：和为给定数

### 【描述】

给出若干个整数，询问其中是否有一对数的和等于给定的数。

### 【输入】

共三行：

第一行是整数 $n$  ( $0 < n \leq 100,000$ )，表示有 $n$ 个整数。

第二行是 $n$ 个整数。整数的范围是在0到 $10^8$ 之间。

第三行是一个整数 $m$  ( $0 \leq m \leq 2^{30}$ )，表示需要得到的和。

### 【输出】

若存在和为 $m$ 的数对，输出两个整数，小的在前，大的在后，中间用单个空格隔开。若有多个数对满足条件，选择数对中较小的数更小的。若找不到符合要求的数对，输出一行No。

### 【样例输入】

4

2 5 1 4

6

### 【样例输出】

1 5

---

# 算法分析

1. 读取数据
  2. 数组排序（升序）
  3. 枚举数组元素 $a[i]$ ，二分法查找 $m - a[i]$ ，查到则输出，并退出程序。
  4. 全部数据枚举后没查到 $m - a[i]$ ，循环结束后输出“No”。
-

# 算法分析

```
sort(a, a + n);
cin >> m;
for ( int i = 0; i < n; i++)
{
    j = bfind_1(0, n - 1, m - a[i]);
    if ( j >= 0 && j != i)
    {
        cout << a[i] << ' ' << m - a[i] << endl;
        return 0;
    }
}
cout << "No" << endl;
```

---

## 例题-3：渔民

有  $n$  户渔民住在海边，他们都整齐的排列成一条直线。

每户渔民用一个坐标  $p_i$  表示，每个渔民活动半径为  $d$ 。如果两户渔民之间的距离不超过  $d$ ，那么这两户渔民就相互认识。请你帮忙统计一下有多少户渔民相互认识。

### 输入格式

第一行两个正整数  $n, d$

第二行  $n$  个整数  $p_i$ ，代表每户渔民的坐标。

### 输出格式

输出一个整数，代表有多户渔民相互认识

对于 50% 的数据  $2 \leq n \leq 10^3$

对于 100% 的数据  $2 \leq n \leq 10^5, 1 \leq d \leq 10^4, 1 \leq p_i \leq 10^9$

### 输入样例

```
5 10
10 12 16 37 40
```

### 输出样例

```
4
```

# 算法分析

## ■ 穷举法

二重循环穷举所有的整数对，检查它们的差值是否小于等于 $d$ ，如果小于等于 $d$ ，则计数。本算法对于大数据会超时。

## ■ 二分查找算法

一重循环枚举数组，用二分查找算法（ $j = \text{bfind\_R}()$ ）查找 $a[i]+d$ ，函数返回小于等于 $a[i]+d$ 的值，从 $i$ 到 $j$ 的范围内，所有的渔民皆与第 $i$ 个渔民相识，累计求和。

## 参考程序

```
for (int i = 0; i < n; i++)
{
    j = bfind_R(0, n - 1, a[i] + m);
    s += j - i;
}
cout << s << endl;
```

---



## 例题-4：接水问题

### 【描述】

学校里有一个水房，水房里一共装有  $m$  个龙头可供同学们打开水，每个龙头每秒钟的供水量相等，均为 1。

现在有  $n$  名同学准备接水，他们的初始接水顺序已经确定。将这些同学按接水顺序从 1 到  $n$  编号， $i$  号同学的接水量为  $w_i$ 。接水开始时，1 到  $m$  号同学各占一个水龙头，并同时打开水龙头接水。当其中某名同学  $j$  完成其接水量要求  $w_j$  后，下一名排队等候接水的同学  $k$  马上接替  $j$  同学的位置开始接水。这个换人的过程是瞬间完成的，且没有任何水的浪费。即  $j$  同学第  $x$  秒结束时完成接水，则  $k$  同学第  $x+1$  秒立刻开始接水。若当前接水人数  $n'$  不足  $m$ ，则只有  $n'$  个龙头供水，其它  $m-n'$  个龙头关闭。

现在给出  $n$  名同学的接水量，按照上述接水规则，问所有同学都接完水需要多少秒。

### 【输入】

第 1 行 2 个整数  $n$  和  $m$ ，用一个空格隔开，分别表示接水人数和龙头个数。

第 2 行  $n$  个整数  $w_1$ 、 $w_2$ 、.....、 $w_n$ ，每两个整数之间用一个空格隔开， $w_i$  表示  $i$  号同学的接水量。

$1 \leq n \leq 10000$ ， $1 \leq m \leq 100$  且  $m \leq n$ ；

$1 \leq w_i \leq 100$ 。

### 【输出】

输出只有一行，1 个整数，表示接水所需的总时间。

样例一

【样例输入】

5 3

4 4 1 2 1

【样例输出】

4

样例二

【样例输入】

8 4

23 71 87 32 70 93 80 76

【样例输出】

163

# 问题分析

■ 样例一

【样例输入】

5 3

4 4 1 2 1

【样例输出】

4

■ 样例二

【样例输入】

8 4

23 71 87 32 70 93 80 76

【样例输出】

163

tap1	tap2	tap3	tap4
23	71	87	32
70			93
	80		
		76	
93	151	163	125

模拟接水过程

# 算法分析

## ■ 数据存储结构

- ◆ 一维数组 $a[101]$ :存储接水时间
- ◆ 下标: 接水龙头编号

## ■ 算法过程（模拟接水过程）

1. 初始化水龙头: 依次读取 $M$ 个人接水数据
2. 枚举数组, 查找最小值的位置 $j$
3. 读取下一个排队的接水数据并加入 $a[j]$
4. 重复2、3步骤, 直到数据处理完毕
5. 在数组中查找最大值
6. 输出结果

tap1	tap2	tap3	tap4
23	71	87	32
70			93
	80	76	
93	151	163	125

$a[1]$	$a[2]$	$a[3]$	$a[4]$
23	71	87	32
93	71	87	32
93	71	87	125
93	151	87	125
93	151	163	125

## 模拟接水参考代码

```
int x, t;  
for ( int i = m + 1; i <= n; i++ )  
{  
    t = Findmin(a, m);  
    cin >> x;  
    a[t] += x;  
}  
t = Findmax(a, m);  
cout << a[t] << endl;
```

---



# 查找最小值函数

```
int Findmin( int p[], int n )
{
    int k = 1;
    int mmin = p[1];
    for ( int i = 2; i <= n; i++ )
    {
        if ( p[i] < mmin )
        {
            mmin = p[i];
            k = i;
        }
    }
    return k;
}
```

数组中最小值的下标

# 查找最大值

```
int Findmax(int p[], int n)
{
    int k = 1;
    int mmax = p[1];
    for ( int i = 2; i <= n; i++ )
    {
        if ( p[i] > mmax )
        {
            mmax = p[i];
            k = i;
        }
    }
    return k;
}
```

数组中最大值的下标