

# AcWing-4633：学生和导师

有 $N$ 个学生（编号 $1 \sim N$ ）正在一起准备编程竞赛。			
为了帮助彼此做好准备，每个学生都要选择一个其他学生作为他的导师，帮助其进步。			
每个学生只能拥有一位导师，但是一个学生可以成为多个学生的导师。			
第 $i$ 个学生的实力评分为 $R_i$ 。			
我们认为，导师不能比其受指导者强太多，所以只有当 $R_j \leq 2 \times R_i$ 时，学生 $j$ 才能成为学生 $i$ 的导师。			
请注意，导师的评分可以小于或等于其受指导者的评分。			
毫不奇怪，每个学生都希望自己的导师尽可能强，所以对于每个学生，请你找出他们可以选择的导师的最高评分。			
<b>输入格式</b> 第一行包含整数 $T$ ，表示共有 $T$ 组测试数据。			
每组数据第一行包含整数 $N$ 。			
第二行包含 $N$ 个整数 $R_1, R_2, \dots, R_N$ 。			
<b>输出格式</b> 每组数据输出一个结果，每个结果占一行。			
结果表示为 <code>Case #x: M_1 M_2 ... M_N</code> ，其中 $x$ 为组别编号（从 1 开始），			

时/空限制:	2s / 64MB
总通过数:	1044
总尝试数:	2396
来源: Google Kickstart2022 Round E Problem B	
算法标签 ▾	

$M_i$  为第

$i$  个学生可以选择的导师的最高评分，如果该学生无法选择任何导师，则输出  $-1$ 。

数据范围

$1 \leq T \leq 100,$

$2 \leq N \leq 10^5,$

$1 \leq R_i \leq 10^6$

输入样例：

```
3
3
2000 1500 1900
5
1000 600 1000 2300 1800
2
2500 1200
```

输出样例：

```
Case #1: 1900 2000 2000
Case #2: 1800 1000 1800 1800 2300
Case #3: 1200 -1
```

样例解释

在 Case 1 中，三个学生的评分分别为 2000, 1500, 1900。

每个学生都可以选择任何其他学生作为他们的导师，因此他们都会选择评分最高的导师。

所以，他们分别选择评分为 1900, 2000, 2000 的导师。

需要注意的是，评分

2000 的学生不能选择自己作为自己的导师，所以只能选择评分

1900 的学生作为导师。

在 Case 2 中，五个学生的评分分别为 1000, 600, 1000, 2300, 1800（请注意，有些学生的评分可能相同）。

对于评分为

1000 的两个学生，他们能够选择的导师的最高评分为

1800, 他们不能选择评分为  
2300 的导师, 因为  
 $2300 > 2 \times 1000$ 。

对于评分为  
600 的学生, 他不能选择评分为  
1800 或  
2300 的导师, 他能够选择的导师的最高评分为  
1000。

对于评分为  
2300 的学生, 他可以选择任何其他学生作为他的导师, 因此他选择评分为  
1800 的导师。

对于评分为  
1800 的学生, 他可以选择任何其他学生作为他的导师, 因此他选择评分为  
2300 的导师。

所以, 五个学生分别选择评分为 1800, 1000, 1800, 1800, 2300 的导师。

在 Case 3 中, 两个学生的评分分别为 2500, 1200。

对于评分为  
2500 的学生, 他可以选择另一个评分为  
1200 的学生作为导师。

对于评分为  
1200 的学生, 他无法选择评分为  
2500 的学生作为导师, 因为  
 $2500 > 2 \times 1200$ 。

所以, 输出结果应该是  
1200 和  
-1。

根据题意, 就是要找出分数小于等于两倍自己分数的同学作为自己的导师, 如果找不到就输出 -1。需要注意的是不可以自己做自己的导师, 导师分数可以小于自己。

那么问题就转化为一个排序后的查找问题。根据数据量, 算法的时间复杂度需要控制在  $O(n \log n)$ , 而排序的时间复杂度已经达到上限, 故可以选择同样为  $n \log n$  的二分作为查找算法。

查找完成后需要判定当前找到的是否大于两倍分数, 如果大于就要向左移一项。同时“判定是否为自己”也可以用左移一项的方式, 即如果找到的导师分数和自己一样, 那么需要判断是否只有自己一个人“占有”这个分数。这里通过统一左移一项避免讨论——如果是自己, 那么真正需要找的是恰好比自己分数少的那一个人; 如果不是自己, 那么左移一项仍然是这个分数数值, 不会影响。

Code:

```
//  
//  main.cpp  
//  4333-学生和导师  
//  
//  Created by MacBook Pro on 2023/8/5.  
//  
  
#include<iostream>  
#include<algorithm>  
#include<cstring>  
using namespace std;
```

```

const int N=100010;
int a[N],s[N];
int main()
{
    int T;
    cin>>T;
    for(int t=1;t<=T;t++)
    {
        int n;
        scanf("%d",&n);
        for(int i=1;i<=n;i++)scanf("%d",&a[i]);
        memcpy(s,a,sizeof a);
        sort(s+1,s+1+n);
        printf("Case #%d: ",t);
        for(int i=1;i<=n;i++)
        {
            int l=1,r=n;
            while(l<r)
            {
                int mid=(l+r)/2;
                if(s[mid]>2*a[i])r=mid;
                else l=mid+1;
            }

            //都进行左移一位的操作（避免临界大于or自己）
            if(s[r]>2*a[i])r--;
            if(a[i]==s[r])r--;
            if(r!=0)
                printf("%d ",s[r]);
            else
                printf("-1 ");
        }
        printf("\n");
    }
    return 0;
}

```

代码提交状态: **Accepted**