题目描述：

现有两组服务器 A 和 B，每组有多个算力不同的 CPU，其中 A[i]是 A 组第 i 个 CPU 的运算能力，B[i]是 B 组第 i 个 CPU 的运算能力。一组服务器的总算力是各 CPU 的算力之和。为了让两组服务器的算力相等，允许从每组各选出一个 CPU 进行一次交换，求两组服务器中，用于交换的 CPU 的算力，并且要求从 A 组服务器中选出的 CPU，算力尽可能小。

输入描述：

第一行输入为 L1 和 L2，以空格分隔，L1 表示 A 组服务器中的 CPU 数量，L2 表示 B 组服务器中的 CPU 数量。

第二行输入为 A 组服务器中各个 CPU 的算力值，以空格分隔。

第三行输入为 B 组服务器中各个 CPU 的算力值，以空格分隔。

1 <= L1 <= 10000

1 <= L2 <= 10000

1 <= A[i] <= 100000

1 <= B[i] <= 100000

输出描述：

对于每组测试数据，输出两个整数，以空格分隔，依次表示 A 组选出的 CPU 算力、B 组选出的 CPU 算力。

要求从 A 组选出的 CPU 的算力尽可能小。

补充说明：

保证两组服务器的初始总算力不同。
答案肯定存在。

示例 1

输入：

```
2 2

1 1

2 2
```

输出：

```
1 2
```

说明：

从 A 组中选出算力为 1 的 CPU，与 B 组中算力为 2 的进行交换，使两组服务器的算力都等于 3。

示例 2

输入：

```
2 2

1 2

2 3
```

输出：

```
1 2
```

说明：

示例 3

输入：

```
1 2

2

1 3
```

输出：

```
2 3
```

说明：

示例 4

输入：

```
3 2
1 2 5
2 4
```

输出：

```
5 4
```

说明：

```
import java.util.HashSet;
import java.util.Scanner;
import java.util.Set;

// 注意类名必须为 Main, 不要有任何 package xxx 信息
public class Main {
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        int count1 = 0;
        int count2 = 0;
        if (sc.hasNextInt()) {
            count1 = sc.nextInt();
        }
        if (sc.hasNextInt()) {
            count2 = sc.nextInt();
        }
        int sumA = 0;
        int sumB = 0;
        Set<Integer> setA = new HashSet<>();
```

```
            Set<Integer> setB = new HashSet<>();
            for (int i = 0; i < count1 && sc.hasNextInt(); i++) {
                int a = sc.nextInt();
                sumA += a;
                setA.add(a);
            }
            for (int i = 0; i < count2 && sc.hasNextInt(); i++) {
                int b = sc.nextInt();
                sumB += b;
                setB.add(b);
            }
            int distance = (sumA - sumB) / 2;
            int[] res = new int[]{100001, 100001};
            for (Integer a : setA) {
                if (setB.contains(a - distance)) {
                    if (a < res[0]) {
                        res[0] = a;
                        res[1] = a - distance;
                    }
                }
            }
            System.out.printf("%d %d\n", res[0], res[1]);
        }
    }
```