# 移动石子直到连续

三枚石子放置在数轴上，位置分别为 a，b，c。

每一回合，我们假设这三枚石子当前分别位于位置 x, y, z 且 x < y < z。从位置 x 或者是位置 z 拿起一枚石子，并将该石子移动到某一整数位置 k 处，其中 x < k < z 且 k != y。

当你无法进行任何移动时，即，这些石子的位置连续时，游戏结束。

要使游戏结束，你可以执行的最小和最大移动次数分别是多少？ 以长度为 2 的数组形式返回答案：answer = [minimum_moves, maximum_moves]

**示例 1：**

**输入：**a = 1, b = 2, c = 5

**输出：**[1, 2]

**解释：**将石子从 5 移动到 4 再移动到 3，或者我们可以直接将石子移动到 3。

**示例 2：**

**输入：**a = 4, b = 3, c = 2

**输出：**[0, 0]

**解释：**我们无法进行任何移动。

```
/**
 * Note: The returned array must be malloced, assume caller calls free().
 */

 /*
 首先先把 a, b, c 排排序

最大值就是把 a1, b1, c1 移到一起所需要的移动步数，也就是 a1 和 b1 之间的距离

把 a1 移向 b1 需要 b1 - a1 - 1

把 c1 移向 b1 需要 c1 - b1 - 1

所以 maxAns = c1 - a1 - 2

那么最少移动步数呢？
```

```
首先如果 a1, b1, c1 已经连续了，也就是 c1 - b1 == 1 && b1 - a1 == 1，那么 minAns = 0；

如果 a1, b1, c1 之间有一个间隔小于等于 2：

比如 1 2 5, 5 移到 3 就结束了，1 3 5，5 移到 2 就结束了，此时 minAns = 1

如果 a1, b1, c1 之间的间隔都很大，比如 1 4 7

那么很明显最少要移动两次，则 minAns = 2

 */
void swap(int* a,int* b,int* c)
{
    if(*a > *b)
    {
        int tmp=*a;
        *a=*b;
        *b=tmp;
    }
    if(*a > *c)
    {
        int tmp=*a;
        *a=*c;
        *c=tmp;
    }
    if(*b > *c)
    {
        int tmp=*b;
        *b=*c;
        *c=tmp;
    }
}
int* numMovesStones(int a, int b, int c, int* returnSize){
    swap(&a,&b,&c);
    int maxans=c-a-2;
    int minans;
    if(c-b==1&&b-a==1)
        minans=0;
    else if(b-a<=2||c-b<=2)
        minans=1;
    else
        minans=2;
    *returnSize=2;
```

```
    int* ans=(int*)malloc(sizeof(int)*2);
    ans[0]=minans;
    ans[1]=maxans;
    return ans;
}
```