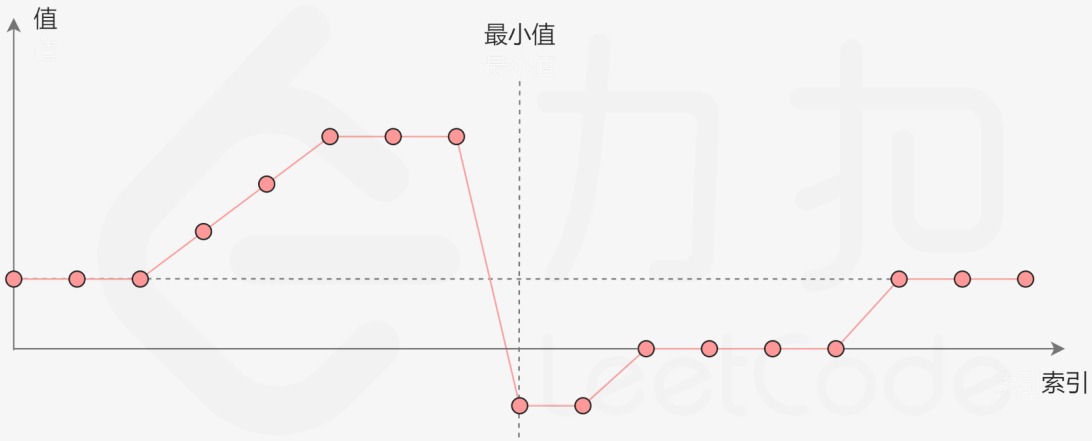
剑指11 旋转数组的最小数字

我是二分法，但是其实是个傻逼。完全等于遍历。

特殊之处在于，mid点和right比较，而不是mid和相邻元素比较。

得好好分析一下边界
当中间一段的左节点大于右节点时，可以确定目标在里面。
当中间一段的左节点小于右节点时，可以确定一定不在里面。
相等的情况就只能慢慢缩小范围看，右节点--。




```
// 官方
class Solution {
public:
    int minArray(vector<int>& numbers) {
        int low = 0;
        int high = numbers.size() - 1;
        while (low < high) {
            int pivot = low + (high - low) / 2;
            if (numbers[pivot] < numbers[high]) {
                high = pivot;
            }
            else if (numbers[pivot] > numbers[high]) {
                low = pivot + 1;
            }
            else {
                high -= 1;
            }
        }
        return numbers[low];
    }
};
```

```cpp


// my 垃圾别看
class Solution {
public:

    int divide(int l, int r, vector<int>& numbers) {
        if(l >= r) {
            return INT_MAX;
        }
        int mid = (l+r)/2;
        if(numbers[mid] > numbers[mid+1]) {
            return numbers[mid+1];
        }
        else{
            return min(divide(l, mid, numbers), divide(mid + 1, r, numbers));
        }
    }
    int minArray(vector<int>& numbers) {
        int out = divide(0, numbers.size() - 1, numbers);
        return out == INT_MAX ? numbers[0] : out;
    }
};

```