## 题目描述：

## 把一个数组最开始的若干个元素搬到数组的末尾，我们称之为数组的旋转。 输入一个非递减排序的数组的一个旋转，输出旋转数组的最小元素。 例如数组{3,4,5,1,2}为{1,2,3,4,5}的一个旋转，该数组的最小值为1。 NOTE：给出的所有元素都大于0，若数组大小为0，请返回0。

## 解题思路：

采用二分法解答这个问题，

mid = low + (high - low)/2

需要考虑三种情况：

(1)array[mid] > array[high]:

出现这种情况的array类似[3,4,5,6,0,1,2]，此时最小数字一定在mid的右边。

low = mid + 1

(2)array[mid] == array[high]:

出现这种情况的array类似 [1,0,1,1,1] 或者[1,1,1,0,1]，此时最小数字不好判断在mid左边

还是右边,这时只好一个一个试 ，

high = high - 1

(3)array[mid] < array[high]:

出现这种情况的array类似[2,2,3,4,5,6,6],此时最小数字一定就是array[mid]或者在mid的左

边。因为右边必然都是递增的。

high = mid

**注意这里有个坑：如果待查询的范围最后只剩两个数，那么mid** **一定会指向下标靠前的数字**

比如 array = [4,6]

array[low] = 4 ;array[mid] = 4 ; array[high] = 6 ;

如果high = mid - 1，就会产生错误， 因此high = mid

但情形(1)中low = mid + 1就不会错误

## 算法描述：

// sort(rotateArray.begin(),rotateArray.end()); //直接排序最方便

// return rotateArray[0];

class Solution {

public:

int minNumberInRotateArray(vector<int> rotateArray) {

int low =0, high = rotateArray.size()-1;

while (low < high)

{

int mid = low + (high-low)/2;

if (rotateArray[mid]>rotateArray[high])

low = mid + 1;

else if (rotateArray[mid] == rotateArray[high])

high--;

else if (rotateArray[mid] < rotateArray[high])

high = mid;

}

return rotateArray[low];

}

};