假设按照升序排序的数组在预先未知的某个点上进行了旋转。

(例如，数组 `[0,1,2,4,5,6,7]` 可能变为 `[4,5,6,7,0,1,2]`)。

请找出其中最小的元素。

你可以假设数组中不存在重复元素。

示例 1:

输入： `[3,4,5,1,2]`
输出： `1`

示例 2:

输入： `[4,5,6,7,0,1,2]`
输出： `0`

思路，该题为2分查找，首先判断一个数组是否旋转了，应对比其头尾，若是头大于尾，那么就是旋转的数组，二分查找，用i，j分别记录头尾，若是中点大于等于头，那么i=中点+1，若是中点小于头，则j=中点-1，因为第二种j-1可能是j=最小值减一，那么我们需判断，j是否是最初的边（j有改动，代表循环启用过），并且j是否比头大于或者等于，大于等于，最小值在j右边，不等于，最小值为i。

代码

```
class Solution {
public:
    int findMin(vector<int>& nums) {
        int minNum=nums[0];
        int i=0,j=nums.size()-1;
        while(i<j&&nums[i]>nums[j]){
            int wz=(i+j)/2;
            if(nums[wz]>=minNum){
                i=wz+1;
                minNum=nums[i];
            }else{
                j=wz-1;
            }
        }
        if(j!=nums.size()-1&&nums[j]>=nums[0]){
            return nums[j+1];
        }
        return nums[i];
    }
};
```

附录一个能过但是特傻的方法，直接快排，取最初的数

```
public:
    int findMin(vector<int>& nums) {
        sort(nums.begin(),nums.end());
        return nums[0];
    }
};
```

自定义测试用例 （贡献给我们）

提交结果：通过 更多明细 >

进行下一个挑战：搜索旋转排序数组 寻找旋转排序数组中的最小值 II