数据流中的中位数

如何得到一个数据流中的中位数？如果从数据流中读出奇数个数值，那么中位数就是所有数值排序之后位于中间的数值。如果从数据流中读出偶数个数值，那么中位数就是所有数值排序之后中间两个数的平均值。

例如，

[2,3,4] 的中位数是 3

[2,3] 的中位数是 (2 + 3) / 2 = 2.5

设计一个支持以下两种操作的数据结构：

- void addNum(int num) - 从数据流中添加一个整数到数据结构中。
- double findMedian() - 返回目前所有元素的中位数。

示例 1：

输入：

```
["MedianFinder","addNum","addNum","findMedian","addNum","findMedian"]
[[],[1],[2],[],[3],[]]
```

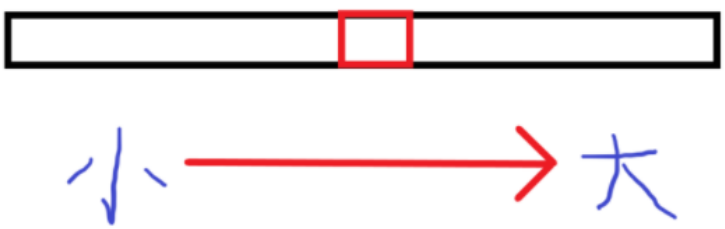
输出：`[null,null,null,1.50000,null,2.00000]`

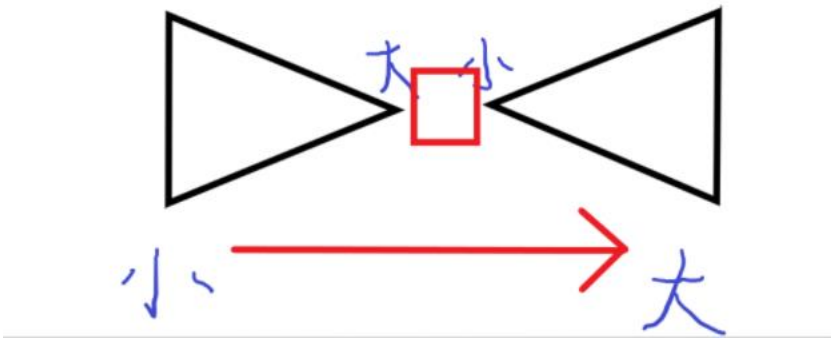
示例 2：

输入：

```
["MedianFinder","addNum","findMedian","addNum","findMedian"]
[[],[2],[],[3],[]]
```

输出：`[null,null,2.00000,null,2.50000]`






```
class MedianFinder {
public:
    //创建优先级队列
    priority_queue<int,vector<int>,less<int>> left;//大根堆（从大到小）堆顶在右边
    priority_queue<int,vector<int>,greater<int>> right;//小根堆（从小到大）堆顶在左边
    /** initialize your data structure here. */
    MedianFinder() {

    }

    void addNum(int num) {
        if(left.empty()||num<left.top())
            left.emplace(num);//没有超过左堆顶，就添加左堆
        else
            right.emplace(num);//超过左堆顶，就添加右堆
        //保证排序后中位数在中间，必须维持左右堆大小一致
        if(left.size()>right.size()+1)
        {
            right.emplace(left.top());//左堆顶划分到右堆存储
            left.pop();
        }
        else if(right.size()>left.size()+1)
        {
            left.emplace(right.top());//左堆顶划分到右堆存储
            right.pop();
        }
    }

    double findMedian() {
        if((left.size()+right.size())%2!=0)
        {
```

```
            return left.size()>right.size()?left.top():right.top();
        }
        else
            return (left.top()+right.top())/2.0;
    }
};

/**
 * Your MedianFinder object will be instantiated and called as such:
 * MedianFinder* obj = new MedianFinder();
 * obj->addNum(num);
 * double param_2 = obj->findMedian();
 */
```