295. Find Median from Data Stream

class MedianFinder {

vector<int> store;

public:

/\*\* initialize your data structure here. \*/

MedianFinder() {

}

void addNum(int num) {

if(store.empty())

store.push\_back(num);

else

/\* num ≤ x \*/

store.insert(lower\_bound(store.begin(),store.end(), num), num);

}

double findMedian() {

int n = store.size();

return n & 1 ? store[n/2] : ((double)store[n/2-1] + store[n/2])\*0.5;

}

};

/\*\*

\* Your MedianFinder object will be instantiated and called as such:

\* MedianFinder\* obj = new MedianFinder();

\* obj->addNum(num);

\* double param\_2 = obj->findMedian();

\*/

class MedianFinder {

priority\_queue<int> lo; // max heap

priority\_queue<int, vector<int>, greater<int>> hi; // min heap

public:

// Adds a number into the data structure.

void addNum(int num)

{

lo.push(num); // Add to max heap

hi.push(lo.top()); // balancing step

lo.pop();

if (lo.size() < hi.size()) { // maintain size property

lo.push(hi.top());

hi.pop();

}

}

// Returns the median of current data stream

double findMedian()

{

return lo.size() > hi.size() ? lo.top() : ((double) lo.top() + hi.top()) \* 0.5;

}

};