### Input:

N: số thao tác, N <= 2.5 \* 10^5

Thao tác 1: 1 x, x <= 10^9

Thao tác 2: 2

### Output:

Với mỗi thao tác 2

### Sample test:

10

1 1

1 7

2

1 9

1 21

1 8

1 5

2

1 9

2

list: 1 5 7 8 9 9 21

2/3 = 0

6/3 = 2

7/3 = 2

output:

no reviews yet

9

9

### Ý tưởng:

minHeap: lưu lại 1/3 số lớn nhất

maxHeap: 2/3 số nhỏ nhất

Thao tác 2: In ra minHeap.top()

Thao tác 1:

Bước 1:

1.1: Thêm x vào minHeap

1.2: Lấy top của minHeap đưa vào maxHeap

Bước 2:

Lấy top của maxHeap đưa vào minHeap đến khi độ dài minHeap = (độ dài minHeap + maxHeap) / 3

### Code:

priority\_queue<int, vector<int>, greater<int>> minHeap;

priority\_queue<int, vector<int>, less<int>> maxHeap;

function main() {

input N;

int m = 0;

while(N--) {

int query;

input query;

if(query == 1) {

intput x;

m++;

// Buoc 1 O(3\*logM)

// Tổng O(N\* 3\*logN)

minHeap.push(x);

int minVal = minHeap.top();

minHeap.pop();

maxHeap.push(minVal);

// Buoc 2 O(MlogM)

// Tổng N bước O(N/3\*logN)

// while(minHeap.size() < (minHeap.size() + maxHeap.size()) / 3) {

// m = 1 => minHeap chứa 0 phần tử

// m = 2 => minHeap chứa 0 phần tử

// m = 3 => minHeap chứa 1 phần tử

// m = 4 => minHeap chứa 1 phần tử

// m = 5 => minHeap chứa 1 phần tử

// m = 6 => minHeap chứa 2 phần tử

if(m % 3 == 0) {

int maxVal = maxHeap.top();

maxHeap.pop();

minHeap.push(maxVal);

}

} else {

if(minHeap.empty()) {

output “No reviews yet”;

} else {

output minHeap.top();

}

}

}

}

O(N\*logN)

6/3 =2

7/3 = 2

9/3 = 3

9%3 = 0

Bước thứ 6

minHeap 2

maxHeap 4

Bước thứ 7

minHeap 2

maxHeap 5

2 < (2 + 5) / 3

2 < 7/3

Bước thứ 8

minHeap 2

maxHeap 6

Bước thứ 9

2 < (2 + 7) / 3

2 < 3