Running Median [Medium]

You will be given C commands. Each command can either add a number to the current sequence S, or choose to print the current Median of the sequence.

Note: input and output formats are given below, but the code template already has input parsing and output routines. You may recode these yourself, but make sure to use fast I/O since input and output files are 26 MB each: recommended are `BufferedReader` / `BufferedWriter`.

Code Template Function APIs and Variables

```
void init();       //initialize variables; this function is called every test case
void add(int num); //add new integer to the sequence
int query();       //return the current median of the sequence
```

Time Limit:

5 seconds

Input:

The first line has a single number T ($T \leq 10$), the number of test cases. T test cases follow.
On the first line of each test case will be a single number C ($C \leq 100{,}000$) which denotes the number of commands to be called. C lines follow.
The i^{th} line has an integer c_i. If the command is add ($c_i = 0$), a new integer a_i ($a_i \leq 10{,}000$) is added to the sequence. If the command is query ($c_i = 1$), your program should output the current Median in the sequence. The Median is the middle element (for odd-length sequences), or the larger element of the middle pair (for even-length sequences). There is always at least 1 element in the sequence before query is called.

Output:

For each test case, output a single line containing the string "`Case #x:` " (where `x` is the case number) and space-separated integers corresponding to the Median queries.

Sample Input:

```
2
6
0 10
0 15
0 12
1
0 6
1
2
0 5
1
```

Sample Output:

```
Case #1: 12 12
Case #2: 5
```