Callstats.io server team interview assignment: Get Median

Problem Set

Conference calls are happening over the Internet and participants are reporting metrics from the endpoints to callstats.io in real-time. The conference call duration can be anything from a few minutes long (industry average for call centers) to a few hours (typical for team meetings) or even longer. In some cases the conference calls last for days (between sites for example). Network engineers rely on particular metrics and look for specific statistical trends to identify calls with bad quality and furthermore, to make decisions. For example, a network engineer might want to identify a bad call based on common statistical measurements: average, max, min, median, standard deviation, skew, etc.

For this problem set, let's assume that we get just one metric, the network delay between participants, reported at regular intervals, typically once every 100ms. At the end of a 5 minute call there would be: 3000 measurements, in 30 minutes: 18000, and in 6 hours: 216000 measurements.

We need to calculate the median over a set of measurements for providing reliable information about the network delay. The measurements are stored in a sliding window, which limits the number of items and the intervals between the first and last element in it. For an instance, a sliding window can contains maximum 5 received items and the maximum interval is 25s.

Your task is to implement a sliding window, which contains a limited amount of items and provides addDelay and getMedian interfaces. The former interface adds a delay value to the sliding window, the latter interface returns the median of the delays calculated over the items from the sliding window.

You must limit your sliding window regarding the number of items it can contain.

Test 1

An example is given below, using a sliding window with length of 3.

The delay measurement arrive one-by-one (iteration) in the following order:

100, 102, 101, 110, 120, 115,

The sliding window should look like this at each iteration:
100
100, 102,
100, 102, 101,
102, 101, 110,
101, 110, 120,
110, 120, 115,

Output: after each iteration (use \r\n delimiter)
-1
101
101
102
110
115

Help:
If only one element available in the sliding window the answer is -1.
If n is odd then Median (M) = value of ((n + 1)/2)th item from a sorted array of length n.
If n is even then Median (M) = value of [((n)/2)th item term + ((n)/2 + 1)th item term] /2

Hint:
If you have a version, which works, try to improve it by focusing the time complexity.

Attached within are additional test vectors:

- Sliding window size for Test 2 is 100
- Sliding window size for Test 3 is 1000
- Sliding window size for Test 4 is 10000

The input file contains a value on each line.

The output file your program generates should have on each line the median of the sliding window.

Test 2

What is the time and space complexity of your solution? If you had more time, how would you improve it?

Deliverables and assignment evaluation

- Please spend no more than 5 hours on this task
- You can choose the language and approach
- In addition to the code, include documentation on what is happening (as well as running instructions, if needed) in whatever format you prefer
- Note that this assignment is a development skills test, so your code should look as mature as possible and be ready to be used in a live system
- Try to find a solution with good time-efficiency
- Please don't go with the most basic/simplistic route with the data structure Share the code e.g via GitHub, GitLab or email attachment, as you wish
- You don't need to include the output files in the assignment submission
- Extra: Some of the following would be considered a plus
    - Comments on how to handle millions of measurements in real-time
    - Having tests and benchmarks
    - Containerized application (Docker)
    - Code formatted according to language style guide