Rolling windows over streams of numbers

Engineering Coding Question

Problem Statement

Given a stream of numbers, write code to calculate the rolling average and rolling maximum over two different window sizes. We provide an illustrative example below. It illustrates the expected output for window sizes of 3 and 5. For your solution please implement a solution that uses window sizes of 3 and 20.

For example, given a stream containing the following values:

[1, 2, 3, 4, 5, 6]

the following tuples would be expected, where 'None' indicates that a value is not available, and would be 'NaN' in some languages:

(None, None, None, None)
(None, None, None, None)
(2, 3, None, None) # 2 = average(1, 2, 3), 3 = max(1,2,3)
(3, 4, None, None)
(4, 5, 3, 5) # 4 = average(3,4,5), 5 = max(3,4,5), 3 = average(1,2,3,4,5), 5 = max(1,2,3,4,5)
(5, 6, 4, 6)

Problem Solving Guidance

You may solve this in C++, Python, Java, or C#. There is some language-specific guidance below.

You may use any publicly available libraries you wish, but please include a list of the projects' homepages in your response.

Python guidance

You should write a class which accepts an iterator and acts as a generator which yields tuples as shown above.

C++ guidance

This is a sample interface for C++; you may use any functionally equivalent interface that you wish. Your implementation would accept an InputStream reference and implement StatisticsGenerator.

```
struct Statistics {
    bool valid;        // true only if mean and max fields are valid
    double mean;
    double max;
};
class StatisticsGenerator {
public:
    // Returns true if more elements are available
    bool HasNext();
    // Get the current statistics.
    std::pair<Statistics, Statistics> GetNext();
};
class InputStream {
public:
    // Returns true if more elements are available
    bool HasNext();
    // Get the next element in the stream
    double GetNext();
};
```

Other languages

For Java or C# use an interface that you feel is idiomatic for the language.

Your Response

Your response should consist of source code that will solve the problem described above. In addition, please provide a write-up discussing of the following topics in your response:

- Any assumptions you made
- The space and time complexity of your algorithm and the trade-offs you made between the two
- Improvements you would make given more time to work on the problem
- Please discuss any change in your approach if instead of 3 and 20 elements, we want to calculate over the last 10,000 or 1 Million elements
- Please discuss how you would extend your code to handle finding the median over these window sizes in addition to the mean and max
- Please discuss how you validated your solution is correct, and how you would do so for a production application at scale

Please note while we do take into consideration the time taken to complete the test, we also value a thoughtful write-up describing your solution.

Your solution will be judged on

1. Correctness
2. Clarity of the code and documentation
3. Efficiency of the implementation
4. Extensibility

Thank you for taking the time to conduct this Engineering Coding Question.