MOVING AVERAGES EXERCISE

This exercises evaluates critical thinking and problem solving skills, specifically **coding and algorithms**. Use any tools or resources that you normally would for programming tasks.

INSTRUCTIONS

Write an implementation of the function specification below. Use whatever mainstream programming language you are most comfortable with.

The code should be efficient and robust. Consider edge cases; what might happen with different inputs (e.g. various array lengths, data values, window sizes)? Describe any potential problems even if you can't think of a solution for a particular case. A driver program is nice, but not required.

SUBMISSIONS SHOULD CONTAIN THE FOLLOWING:

- Code that solves the problem above
- Instructions for how to compile and run your code
- A description of your solution and any assumptions it makes
- Analysis of the time complexity of your solution, preferably in terms of big-O notation
- Analysis of the space complexity of your solution, preferably in terms of big-O notation

SPECIFICATION

A function that accepts inputs for an integer window size and an array of double-precision floating point values, and returns an array of double-precision floating point values. The nth value of the output array should be the average after processing the first n elements of the input array. The average should be computed as a cumulative average until the number of elements reaches the window size, and as a simple moving average using the window size afterward.

EXAMPLES

```
compute(3, [0, 1, 2, 3]) => [0, 0.5, 1, 2]

compute(5, [0, 1, -2, 3, -4, 5, -6, 7, -8, 9]) =>
       [0, 0.5, -0.333333333333333, 0.5, -0.4, 0.6, -0.8, 1, -1.2, 1.4]
```