Beau Enslow

Module 7 Project

29th November 2023

tinyArray: Append/Push - 197.541 µs Insert/Unshift - 88.098 µs

smallArray: Append/Push - 50.46 µs Insert/Unshift - 17.275 µs

mediumArray: Append/Push - 60.508 µs Insert/Unshift - 154.098 µs

largeArray: Append/Push - 570.685 µs Insert/Unshift - 10.325073 ms

extraLargeArray: Append/Push - 5.050672 ms Insert/Unshift - 1.32502762 s

Read over the results, and write a paragraph that explains the pattern you see. How does each function "scale"? Which of the two functions scales better? How can you tell?

The larger the array, the slower the insert function takes to complete its calculations. The append function scales better based on the time differential. It's clear that the Append/Push function would be far better for scale. Especially if you were dealing with a list exponentially larger than the ones we are seeing in this example.

For extra credit, do some review / research on why the slower function is so slow, and summarize the reasoning for this.

From my research I have learned that the array.push() time complexity is shorter because the process is always 1. Meaning that every time you use push you add 1 to the end of the number in the array. That is what is meant by 0(1). It doesn't matter if there are 10 elements or 20,000 elements, the process is always array number + 1. The array indexes stay the same, we just add one index to the end. Whereas array.unshift() adds the item to the beginning of the array giving it an index of 0. Every other item in the array is assigned a new number as they shift over 1 or 0(n).

Imagine a room of people sitting in chairs lined up in order. Push is similar to a new person grabbing a new chair and putting it at the end of the row of people. Unshift is like taking that chair and putting it at the end of the line but instead of sitting- that person goes to the very first seat and asks everyone to move over one seat. Which process takes more effort? Which process takes more time? It becomes clear.