The runtime results for the extraLargeArray are different when passed to doublerAppend and doublerInsert because of how each of the functions work. doublerAppend uses the .push() method which adds to the end of an array and has a runtime of O(1). It has a set amount of steps. The doublerInsert function uses the .unshift() method which adds data to the beginning of an array and has to update the index of every following item in the array. The unshift method has a runtime of O(n) because it is dependent on how many items are in said array.

Tiny

insert 167.946 μs

append 166.216 μs

Small

insert 324.63 μs

append 125.464 μs

Medium

insert 161 μs

append 224.422 μs

Large

insert 11.71579 ms

append 1.004415 ms

Extra large

insert 1.3036408609999999 s

append 35.818205 ms

By looking at the runtime data above, we can easily tell that the insert function takes much longer to run than the append function when the data starts to scale up. Even on the extraLargeArray, the append function runs in only 35.8 ms while the insert function takes over 1.3 seconds! This is because of the reasoning mentioned above. The append function utilizes the .push() method which has a runtime time complexity of O(1). It is only dependent on how many items are being appended to the array whereas the .unshift() method used by the insert function has a runtime time complexity of O(n) because it is also dependent on how many items are in the array itself.