Results of runtime.js for different arrays

- extraLargeArray: insert = 670.2537 ms, append = 2.3862 ms
- largeArray: Insert = 6.7492 ms, append = 394.1 us
- mediumArray: Insert = 154.3 us, append = 112.2 us
- smallArray: Insert = 46.7 us, append = 77.4 us
- tinyArray: Insert = 36 us, append 79.1 us

	Insert	Append	Time diff
Extra Large Array	670.2537 ms	2.3862 ms	667.8675 ms
Large Array	6.7492 ms	394.1 us	6.3551 ms
Medium Array	154.3 us	112.2 us	42.1 us
Small Array	46.7 us	77.4 us	30.7 us
Tiny Array	36 us	79.1 us	43.1 us
Time diff avg	135447.98 us	609.8 us	

This is interesting, each array scaled differently. With a variety of time differences depending on how long each array was. The Insert array struggled with larger arrays, but shined with smaller arrays. While the Append function was shining with larger arrays, but got slightly slower with the tiny array.

Overall, the Append function scaled better than the Insert function. With the Insert function, they had the biggest time difference between the different sizes of arrays, while the Append function had

the smallest time difference between the sizes. This tells us that the Append function will be more stable to use, no matter what size of the array we are pushing.

Overall the Append function is more desirable, since it uses the .push method, which simplys adds elements to the end of the array then returns the new length of the array. While the Insert function uses the unshift method, which adds the new element in front of the array, moving the entire array over. Creating more time delay.