Module 7 Assessment

In your notes document, take note of the timing result for the extraLargeArray results– comparing when the extraLargeArray is passed to doublerAppend and doublerInsert.

Results for the extraLargeArray
  insert 1.097346084 s
  append 18.883583 ms

Runtime Analysis:

Array	Insert	Append
tinyArray	5.583 μs	5.583 μs
smallArray	17.167 μs	7.709 μs
mediumArray	154.292 μs	52.584 μs
largeArray	8.491459 ms	504.792 μs
extraLargeArray	1.082402833 S	3.284208 ms

When running the two functions it is evident that the Append function scales much better than the Insert function. With our largest array, the append function only took 3.28 milliseconds, while the Insert function's runtime was slightly over a full second. It is quite interesting that at the start there is zero difference in runtime between the two functions. The reason for the slower runtime all lies within the functions themselves. Both functions are utilizing array methods to insert numbers in an array. In the Insert function, .unshift is used, and in Append .push is used. The .unshift array method inserts numbers to the front of the array, while .push simply places the new number at the end of the array. This extra step is what is causing the added runtime for the Insert function. Every time a new number is added the insert function has to move the index of the added number how many places as the index is. In the case of the ExtraLargeArray, that means the final number added will have its index changed 100000, and every number before it will have its index changed as well. Whereas in the append function, the number is simply added to the end.