**Runtime Analysis**

100,000 SIZE ARRAY

Results for the extraLargeArray

insert 1.048717588 s

append 4.364951 ms

10,000 SIZE ARRAY

Results for the largeArray

insert 7.630257 ms

append 864.921 μs

1,000 SIZE ARRAY

Results for the mediumArray

insert 337.606 μs

append 214.08 μs

100 SIZE ARRAY

Results for the smallArray

insert 145.281 μs

append 132.561 μs

10 SIZE ARRAY

Results for the tinyArray

insert 132.06 μs

append 122.569 μs

**QUESTIONS:** 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?

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

**ANSWER:**

As the array gets larger in size, the insert (unshift function) time increases as well. This is due to time complexity. The unshift function requires more time to process as it must first push back the elements in each index then finally add the new element.  
  
Likewise, with the append (push function) time increases as the array gets larger in size. This is also due to time complexity.  
  
The push function scales much better which is evidently seen in the runtime results. Again, this is due to a better time complexity.