Assessment 7 Responses

Function	extra-large 100,000	large 10,000	medium 1,000	small 100	tiny 10
insert	513.4346 ms	5.3448 ms	108.3 µs	29.5 µs	22 µs
append	2.1947 ms	379.5 µs	92.3 µs	64.6 µs	63.3 µs

The insert (unshift) function has a faster runtime than the append (push) function for arrays of less than 1000 items. Around and above 1000 items, the insert function scales to a runtime considerably longer than the append function, making the append function the faster option for these larger arrays. The append function scales more linearly as the input array size increases. Append's runtime seems more resilient to throttling by input size as compared to insert. Consequently, the append/push function scales better for increased processing and input demand.

Extra Credit:
Unshift and push seemingly perform the same operation of adding one new item, the difference being unshift adds to the front of the stack and push adds to the end of the stack. However, in terms of memory allocation, push only needs to create new memory to append to the end of the original array. The insert function is much slower because unshift not only needs to add the new element to the beginning of the array, but also shift every single other element from its original index + 1. Reassigning every other element from its previous location adds more time complexity to unshift compared to push.