runTime.js

```
Results for the tinyArray
insert 85.113 µs
append 104.702 µs

Results for the smallArray
insert 91.559 µs
append 215.443 µs

Results for the mediumArray
insert 165.369 µs
append 152.454 µs

Results for the largeArray
insert 7.636429 ms
append 645.715 µs


Results for the extraLargeArray
insert 1.111336524 s
append 18.996483 ms
```

1. 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?

As far as time the fastest function has the input of tinyArray, this is because it has the smallest input. Since time complexity is related to how big of an input a function has, the larger input will take longer for a function to execute. Between the two functions the function called doublerApprend is more efficient because it uses the .push() method opposed to the other function called doubleInsert() that uses the .upshift() method. The Reason .push() is faster than .Unshift() is because .push() only has to add an element to the end of the array, .unshift() has to check the first item in a array then shift it over and finally add what's needed to the array, this i terms takes longer.