Step 1.

Input	Runtime append	Runtime insert
extraLargeArray (100000)	2.6676 ms	1.072026 s
largeArray (10000)	209.4 µs	8.8642 ms
mediumArray (1000)	889.7 µs	1.0315 ms
smallArray (100)	2.7 µs	11.7 µs
tinyArray (10)	2.3 µs	19.3 µs

The append function runtime is growing at a slower rate than insert function runtime. The scale on append seems to be growing linearly, whereas the scale on the insert function seems to be growing much more quickly. The scale seems to be roughly linear for the Append function, as if you compare extraLarge to large they're off by a factor of 10. The scale of insert looks quadratic, to avoid inconsistencies in the dat, the best data are probably from comparing largeArray and extra Large array, and in that instance the difference is on the order of 100 fold increase. If the difference in data size is 10, and the output growth is 100 that suggests quadratic growth.

The insert function is shifting a value off the beginning of the array, this forces all the elements array to be moved one spot forward in the array. What this does is cause n swaps on an array where n is the length of the array. If we are doing n insertions this means we will be doing n*n or n^2 operations, causing O(n^2) complexity.