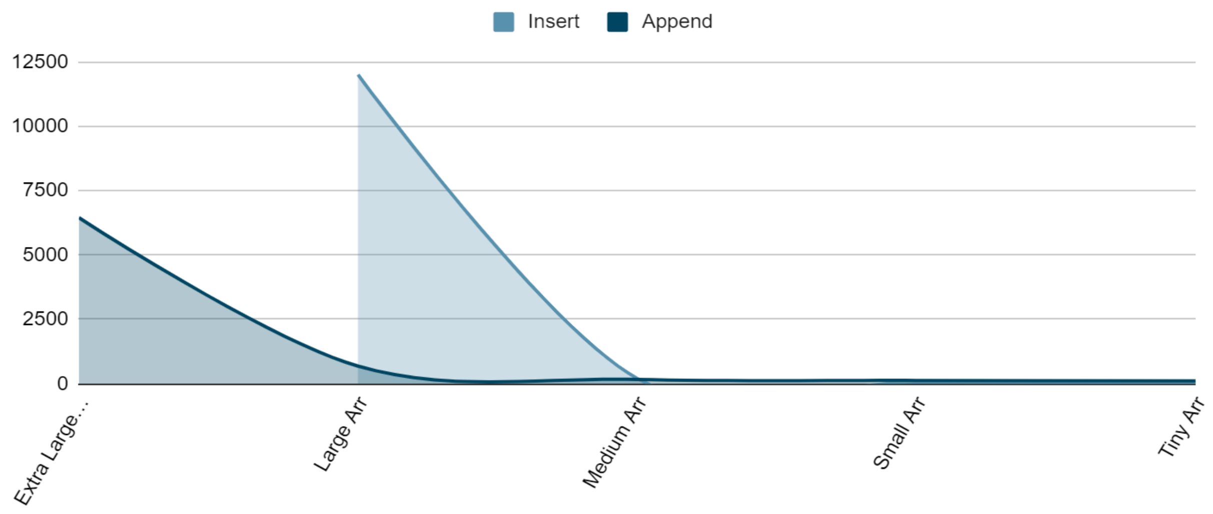
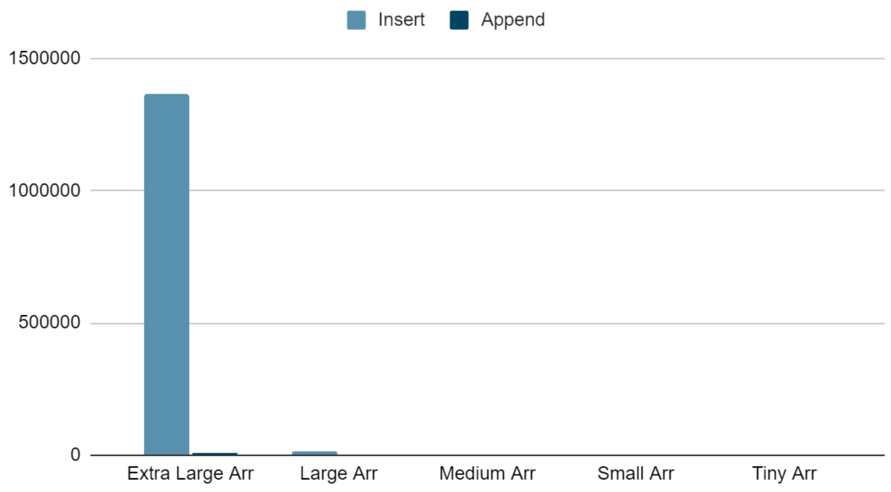
Graphs
The area chart displays the trend difference, where insert increases exponentially to Extra large array(Extra array removed from this graph for clarity, see next graph for scale inference) All times converted to μs.



The scale of Extra Large array time:



Summary:
The function Append is much more efficient for large scale data time complexity based on the dramatic time change.

Review/research why is this so:
Both functions use a single for loop to go over the data and return an altered array based on the loop findings, but use different methods of .push and .unshift.

Append uses .push and .push default time complexity is O(1). This is because it only needs to insert an element to the end of the array, without editing any of the other indexes.

Insert uses .unshift and .unshift default time complexity is O(n). This is because it adds the element to index 0, and thereby must increment all the elements present in the array. Although it is worth noting if used with a linked list which can bring it's complexity to O(1), but this is less commonly done and not the case in these functions.

Timing results:

Results for the extraLargeArray
insert 1.3629594 s
append 6.455 ms

Results for the tinyArray
insert 43.7 μs
append 116.2 μs

Results for the smallArray
insert 60.8 μs
append 133.7 μs

Results for the mediumArray
insert 212 μs
append 172.9 μs

Results for the largeArray
insert 12.0084 ms
append 689.2 μs