Notes Document

Results for the extraLargeArray

insert 985.4587 ms

append 2.6482 ms

|  |  |  |
| --- | --- | --- |
|  | doublerAppend | doublerInsert |
| Tiny Array | 98.4 μs | 34.1 μs |
| Small Array | 96.5 μs | 42.9 μs |
| Medium Array | 133.5 μs | 165.1 μs |
| Large Array | 488.3 μs | 8.6888 ms |
| Extra Large Array | 2.6482 ms | 985.4587 ms |

While the doublerAppend function starts slower than the insert function, a larger input makes it take more and more time to complete, putting it behind the Append function. As the inputs grow larger, the time the insert function takes grows quite quickly, becoming incredibly inefficient and time consuming. The append function, on the other hand, seems to take a consistently longer amount of time, scaling its runtime at a predictably manageable rate.

This is because the append function adds the new values to the end of the array, needing only to find the end of the array on every iteration and put the value into memory at that location, while the insert function puts the new item at the beginning of the array, pushing all the items back by 1 in every iteration. This movement doesn’t take much effort in the case of a small array, but once an array gets big enough it takes longer and longer to push every item down by 1, and having to do so over and over again with more and more numbers, that quickly adds up.