|  | doublerAppend() | doublerInsert() |
| --- | --- | --- |
| tinyArray = 10 | 62.0 us | 27.5 us |
| smallArray = 100 | 66.7 us | 34.5 us |
| mediumArray = 1000 | 120.5 us | 173.3 us |
| largeArray = 10,000 | 377.9 us | 7.2781 ms |
| extraLargeArray = 100,000 | 2.3815 ms | 714.0814 ms |

While using small data sets, unshift() performs better than push(). However, once the data sets become much larger, push() far outperforms unshift(). This difference is because as the data sets reach “infinity”, or as they become larger, the push() method runs at constant time, while the unshift() method runs in linear time.

The main difference in performance lies in how each method processes the data. The push() method simply adds the item to the end of the array, only requiring the length of the array to access the index at the end. Side note: If the array size was already predetermined, and the push() method would cause the array to go over that memory allocation, then it would require new memory to be allocated and copied over. While the unshift() method requires inserting the item into the 0 index, and then subsequently moving the other items up one index and finally returning the length of the array. Essentially the easiest way to perform the unshift() method is by allocating new memory and copying over the new array. In summary, the push() method rarely needs to allocate new memory, and the unshift() method always allocates new memory.