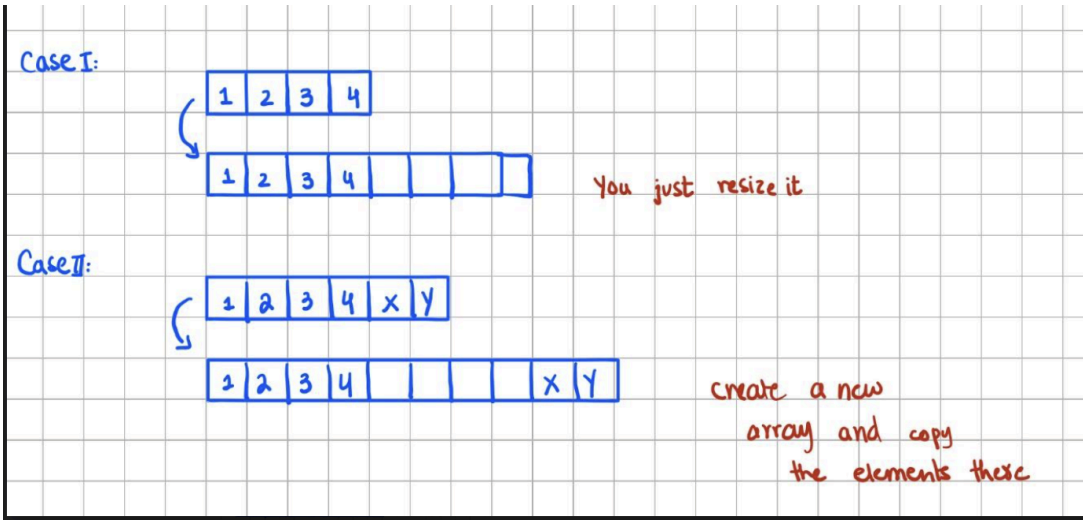
1. The size of the array refers to how many elements it has at the time and the capacity of the array is the largest number of elements it can hold (or the maximum allocated memory space).

2. If we want to grow the array beyond its current capacity, we resize it in the dynamic array.
    - If there is space in the memory after the end of the array we just add extra memory space at the end of the array (like resizing the array) and add the element there.
    - If any other variable occupies the memory then we create a new array with a larger enough capacity and copy the elements from the original array to the new larger in-size array.



3. One real-world array implementation example is to amortize the cost of array expansion used in geometric resizing. When introducing n elements into an array, using a geometric progression for the array capabilities, such as expanding it with a constant percentage 'a', assures that the entire time complexity is O(n). In this application, the phrase "amortized constant time" refers to the fact that, on average, each insertion operation takes the same amount of time despite periodic resizing actions, resulting in more predictable and efficient performance across the whole insertion sequence.
   Source:
   https://en.wikipedia.org/wiki/Dynamic_array#:~:text=Geometric%20expansion%20and%20amortized%20cost,-To%20avoid%20incurring&text=As%20n%20elements%20are%20inserted,insertion%20takes%20amortized%20constant%20time.