1) The size of an array represents the actual number of elements that are stored in an array. Essentially, it is the count of elements in an array. For example, if we are given an array of integers [1,2,3,4,5,6]. In this case, the size of the array is 6. On the other hand, the capacity of an array refers to the total amount of space that has been allocated for the array. It is basically the maximum number of elements that the array can currently hold.
In the case of dynamic arrays, the capacity is often bigger than or equal the size of the array.

2) In dynamic arrays, expanding an array beyond its present size usually requires reallocating memory to accommodate additional elements. A combination of a fixed-size array and a dynamically allocated buffer is often implemented to build dynamic arrays. The resizing process is typically handled by the inner functions responsible oof the dynamic array.

When there is space available after the end of the array, the inner function resizing the dynamic array allocates a new, larger memory block. Then it copies the existing elements from the old memory block to the new one. The array's internal pointers r metadata are updated to reflect the new memory block and increased capacity. Finally, the old memory block is deallocated to prevent memory leaks.

In the case of space occupied by another variable after the end of the array, just like before it resizes and allocates a larger memory block, copies the existing elements from the old array to the new one. Then the array's internal pointers are updated. The variable occupying the space is also copied to the new memory. After it's done, the old memory block and the meory occupied by the other variable are deallocated.