1. Explain the difference between an array size and capacity

Although the array size and its capacity are similar, there is a small difference between the two. The size of an array represents the number of elements currently stored in the array, while the capacity of the array represents the maximum number of elements the array can hold without having to allocate more memory.

2. What happens when an array needs to grow beyond its current capacity? Explain and produce a diagram showing the memory layout before and after expansion

As explained in the earlier question, to expand the array's capacity, it must allocate more memory.

i) First consider the case where there is space in memory after the end of the array

When there is space in memory after the end of the array, Python grows the array by allocating a larger chunk of memory and copies the content while adding the new elements to the end of the array. In terms of Python, the array grows by a factor of 1.125 in case further elements are added. After this process finishes, the old array is then handled by Python's garbage collection system.

ii) Then consider the case where the memory after the end of the array is occupied by another variable. What happens in that case?

In the case where the space in memory after the end of the array is occupied, Python must allocate a new larger chunk of memory elsewhere in the memory space. Similar to before, Python copies the elements from the old array to the new array, but the variable that was occupying the space after the old array is left untouched.

3. Discuss one or more techniques real-world array implementations use to amortize the cost of array expansion

One of the real-world techniques to amortize the cost of array expansion is changing the factor at which the capacity of the array is expanded. For example, pretend that the factor is currently expanding the array by 1.5x but you notice that even after this resizing, it is resized again due to the capacity of the array being exceeded. Because of this, you decide to instead double the array’s capacity when it needs to be expanded. Because of this, only one resizing is needed. This reduces the frequency at which an array is expanded which will amortize the cost of array expansion.