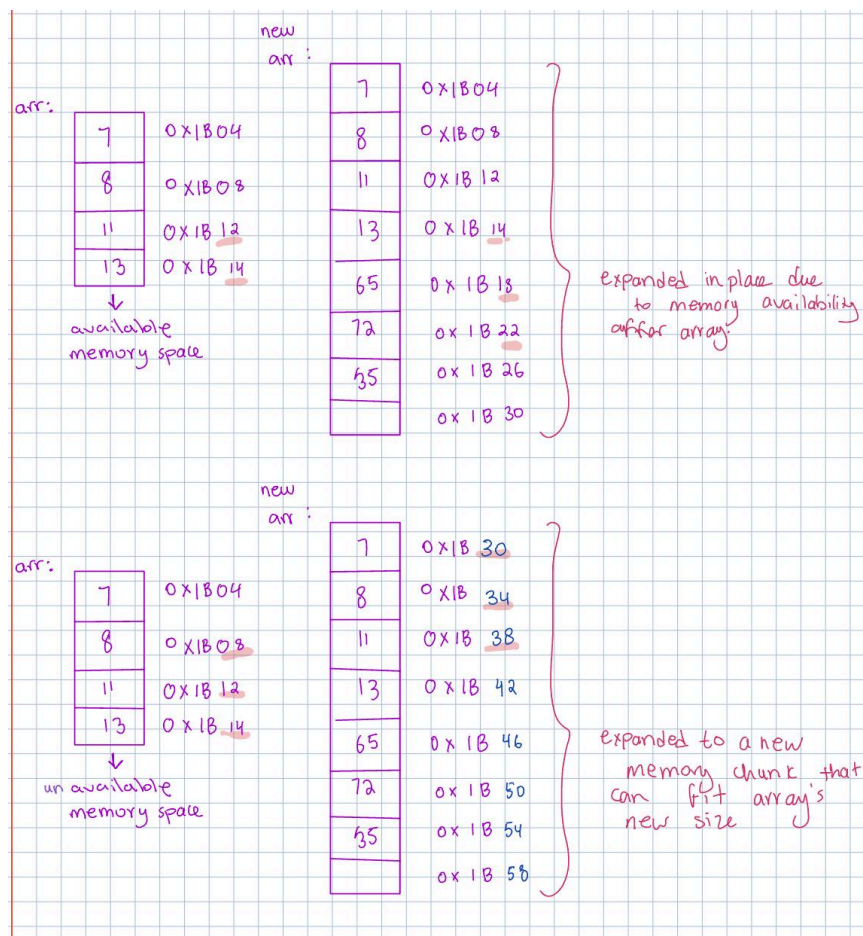**Question 1:**
- Array Size:
    - Size is the number of elements that are currently being stored in an array.
- Capacity:
    - Capacity is the max number of elements an array can hold without resizing/ allocating a new chunk memory.

Question2:

- In Case 1 - When there is memory available after the array, the array can expand in place without needing to copy over all of its elements to a new location. This is efficient as the array can avoid relocation.
- In Case 2 - when there is no available memory immediately after the array, the array will need to be relocated to a new area in the memory that can accommodate its elements and new additions. This is done through copying the array's elements to the new memory space and allocating extra room for the growth.

arr:

7	0X1B04
8	0X1B08
11	0X1B 12
13	0X1B 14

↓
available memory space

new arr:

7	0X1B04
8	0X1B08
11	0X1B 12
13	0X1B 14
65	0X1B 18
72	0X1B 22
35	0X1B 26
	0X1B 30

expanded in place due to memory availability after array.

arr:

7	0X1B04
8	0X1B08
11	0X1B 12
13	0X1B 14

↓
un available memory space

new arr:

7	0X1B 30
8	0X1B 34
11	0X1B 38
13	0X1B 42
65	0X1B 46
72	0X1B 50
35	0X1B 54
	0X1B 58

expanded to a new memory chunk that can fit array's new size

Question 3:

1. One way would be to double the size of the array every time it's expanded. This will allow for extra space for the array to grow in if needed, without needing to expand again.
2. Another way would be to use a specific growth factor for each expansion, allowing for extra space for growths without needing expansion, this is already used, such as by python. .
3. Another way would be to reduce the size of the array size when elements are removed to ensure no memory waste.