Level: Easy

Given an array of numbers, replace each even number with two of the same number. e.g, [1,2,5,6,8, , , ,] -> [1,2,2,5,6,6,8,8].

Assume that the array has the exact amount of space to accommodate the result.

Questions to Clarify:
Q. How do you want to return the result?
A. Modify the input array.

Q. What would an empty element contain?
A. -1

Solution:
We use reverse traversal. We keep 2 pointers, one that loops through the existing numbers, and one that keeps track of the end. When we encounter an even number, we move two instances of it to the end. When we encounter an odd number, we move only one instance to the end.

Pseudocode:

```
(Note: Never write pseudocode in an actual interview. Unless you're writing a few lines quickly to plan out your solution. Your actual solution should be in a real language and use good syntax.)


end = a.length - 1
current -> getLastNumber(a) to 0
   if even, copy value to end, decrease end, do this twice
   if odd, copy value to end, decrease end
```

Test Cases:
Corner Cases - null, empty array, array with only blanks
Base Cases - one odd number, one even number
Regular Cases - only odd numbers, only even numbers, both odd and even numbers

Time Complexity: O(n) aka linear time

Space Complexity: O(1) aka constant space

```
public static int[] cloneEvenNumbers(int[] a) {
    if (a == null || a.length == 0)
        return a;

    int end = a.length, i = getLastNumber(a);
    while (i >= 0) {
```

```
        if (a[i] % 2 == 0) {
            a[--end] = a[i];
        }
        a[--end] = a[i];
        i--;
    }
    return a;
}

public static int getLastNumber(int[] a) {
    int i = a.length - 1;
    while (i >= 0 && a[i] == -1) {
        i--;
    }
    return i;
}
```