Sort Colors

Given an array `nums` with `n` objects colored red, white, or blue, sort them **in-place** so that objects of the same color are adjacent, with the colors in the order red, white, and blue.

We will use the integers `0`, `1`, and `2` to represent the color red, white, and blue, respectively.

You must solve this problem without using the library's sort function.

Example 1:

```
Input: nums = [2,0,2,1,1,0]

Output: [0,0,1,1,2,2]
```

Example 2:

```
Input: nums = [2,0,1]

Output: [0,1,2]
```

Note: this problem is variation of Dutch national flag problem

```
// two pass O(m+n) space
void sortColors(int A[], int n) {
    int num0 = 0, num1 = 0, num2 = 0;

    for(int i = 0; i < n; i++) {
        if (A[i] == 0) ++num0;
        else if (A[i] == 1) ++num1;
        else if (A[i] == 2) ++num2;
    }

    for(int i = 0; i < num0; ++i) A[i] = 0;
    for(int i = 0; i < num1; ++i) A[num0+i] = 1;
    for(int i = 0; i < num2; ++i) A[num0+num1+i] = 2;
```

```
}
// one pass in place solution=
void sortColors(int A[], int n) {
    int n0 = -1, n1 = -1, n2 = -1;
    for (int i = 0; i < n; ++i) {
        if (A[i] == 0)
        {
            A[++n2] = 2; A[++n1] = 1; A[++n0] = 0;
        }
        else if (A[i] == 1)
        {
            A[++n2] = 2; A[++n1] = 1;
        }
        else if (A[i] == 2)
        {
            A[++n2] = 2;
        }
    }
}


// one pass in place solution
void sortColors(int A[], int n) {
    int j = 0, k = n - 1;
    for (int i = 0; i <= k; ++i){
        if (A[i] == 0 && i != j)
            swap(A[i--], A[j++]);
        else if (A[i] == 2 && i != k)
            swap(A[i--], A[k--]);
```

```
    }
}


// one pass in place solution
void sortColors(int A[], int n) {
    int j = 0, k = n-1;
    for (int i=0; i <= k; i++) {
        if (A[i] == 0)
            swap(A[i], A[j++]);
        else if (A[i] == 2)
            swap(A[i--], A[k--]);
    }
}
```