# Sort Colors (Dutch National Flag Problem) - Notes

This problem is known as the **Dutch National Flag Problem**. You are given an array `nums` with values `0`, `1`, and `2`. The task is to sort the array in-place so that all `0`s come first, followed by all `1`s, then all `2`s. ### Key Idea: - Use three pointers: `low`, `mid`, and `high`. - `low` → boundary for `0`s (everything before `low` is 0). - `mid` → current element being checked. - `high` → boundary for `2`s (everything after `high` is 2). - Traverse the array with the following rules: - If `nums[mid] == 0`: swap `nums[low]` and `nums[mid]`, then increment both `low` and `mid`. - If `nums[mid] == 1`: just move `mid` forward. - If `nums[mid] == 2`: swap `nums[mid]` and `nums[high]`, then decrement `high`. ### Complexity: - Time Complexity: O(n) (each element is processed once). - Space Complexity: O(1) (in-place sorting). Below is the C++ code implementation:

```
class Solution {
public:
    void sortColors(vector<int>& nums) {
        int n = nums.size();
        int low = 0;
        int mid = 0;
        int high = n - 1;

        while(mid <= high){
            if(nums[mid] == 0){
                swap(nums[low], nums[mid]);
                low++;
                mid++;
            }

            else if(nums[mid] == 1){
                mid++;
            }

            else{
                swap(nums[high], nums[mid]);
                high--;
            }
        }
    }
};
```