# Dutch National Flag Algorithm - Explanation & Code

The Dutch National Flag Algorithm efficiently sorts an array containing only three distinct values (0s, 1s, and 2s).

Pattern: Three-Way Partitioning

---

This algorithm partitions the array into three sections using three pointers:

- `low`: Boundary for 0s (left side)
- `mid`: Current element being processed
- `high`: Boundary for 2s (right side)

Algorithm Steps:

1. If arr[mid] == 0: Swap arr[mid] with arr[low], then move both low and mid forward.
2. If arr[mid] == 1: Just move mid forward.
3. If arr[mid] == 2: Swap arr[mid] with arr[high], then decrease high (but don't move mid immediately).

Time Complexity: O(n) (single pass through the array)

Space Complexity: O(1) (in-place sorting)

```
public void sortColors(int[] arr) {
    int low = 0, mid = 0, high = arr.length - 1;

    while (mid <= high) {
        if (arr[mid] == 0) {
            swap(arr, low, mid);
            low++;
            mid++;
        } else if (arr[mid] == 1) {
            mid++;
        } else { // arr[mid] == 2
            swap(arr, mid, high);
            high--;
        }
    }
```

```
}

private void swap(int[] arr, int i, int j) {
    int temp = arr[i];
    arr[i] = arr[j];
    arr[j] = temp;
}
```