DSA Revision Notes

1. Best Time to Buy and Sell Stock (Greedy)

- Goal: Max profit with only one transaction.
- Track minimum price seen so far.
- At each day, compute profit = current price - min_price.
- Update max_profit accordingly.
- Time: O(n), Space: O(1).

2. Maximum Subarray (Kadane's Algorithm)

- Goal: Find maximum sum of contiguous subarray.
- Maintain current_sum and max_sum.
- current_sum = max(nums[i], current_sum + nums[i]).
- Update max_sum each step.
- Handles negative arrays correctly.
- Time: O(n), Space: O(1).

3. Move Zeroes (Two Pointers)

- Goal: Move all zeros to end while maintaining order.
- Use pointer k for position of next non-zero.
- Swap or overwrite non-zero elements forward.
- Time: O(n), Space: O(1).

4. Remove Duplicates from Sorted Array (Two Pointers)

- Array is sorted, so duplicates are adjacent.
- Use slow pointer k to track unique placement.
- If nums[i] != nums[i-1], place at nums[k] and increment k.
- Return k as new length.

- Time: O(n), Space: O(1).

5. Merge Sorted Array (Three Pointers from End)

- Merge nums2 into nums1 in-place.
- Use pointers i = m-1, j = n-1, k = m+n-1.
- Compare from back and place larger element at k.
- Continue until nums2 is exhausted.
- Time: O(m+n), Space: O(1).

6. Majority Element (Boyer-Moore Voting)

- Goal: Find element appearing > n/2 times.
- Maintain candidate and count.
- If count == 0, set candidate = current.
- If current == candidate, count++ else count--.
- Works because majority > sum of all others.
- Time: O(n), Space: O(1).