CodeVerse

Google Developer Groups
On Campus • RV University

Days 2-3: Array and String Notes

Arrays:-

Definition: An array is a collection of elements of the same type stored in contiguous memory locations. Arrays enable efficient access to elements using their index.

- **Access : O(1) -** Direct access using the index.
- **Insertion/Deletion:**

* **O(n) in the middle -** Requires shifting elements.

* **O(1) at the end (for dynamic arrays) -**

Amortized constant time due to resizing.

Use Cases : Storing lists of data, implementing other data structures (e.g., stacks, queues), representing matrices.

LeetCode Tip: Arrays are fundamental; master basic operations like traversal, insertion, and deletion. Pay close attention to edge cases and array bounds.

Array Operations

1. Insertion

Description : Adding an element at a specific index. Involves shifting elements to make space.

Complexity : O(n)

```
Algorithm InsertElement(A, index, value):
  // A is the array, index is the position, value is the element to insert
  For i from length(A) - 1 down to index:
    A[i + 1] = A[i]  // Shift elements to the right
  A[index] = value
```

2. **Deletion**

Description: Removing an element at a specific index. Involves shifting elements to fill the gap.

Complexity: O(n)

```
Algorithm DeleteElement(A, index):
  // A is the array, index is the position to delete
  For i from index to length(A) - 2:
    A[i] = A[i + 1]  // Shift elements to the left
```

3. **Traversal**

Description: Iterating through all elements in the array.

Complexity : O(n)

```
Algorithm TraverseArray(A):
  // A is the array
  For i from 0 to length(A) - 1:
    Process A[i]  // Access and process each element
```

Subarrays

Definition : A contiguous portion of an array.

Use Cases : Finding maximum/minimum sums, searching for patterns.

LeetCode Tip: Understand how to generate all possible subarrays. Nested loops are often used.

Kadane's Algorithm (Maximum Sum Subarray)

Description: Finds the maximum sum of any contiguous subarray efficiently.

Complexity : O(n)

Tip: This algorithm is a classic example of dynamic programming. It builds upon the optimal solution for smaller subproblems.

Algorithm Kadane(A):

```
 // A is the array
 max_ending_here = A[0]
 max_so_far = A[0]
 For i from 1 to length(A) - 1:
   max_ending_here = max(A[i], max_ending_here + A[i])  // Extend or start new
   max_so_far = max(max_so_far, max_ending_here)  // Update global max
 Return max_so_far
```

Kadane's Algorithm | Maximum Subarray Sum | Finding and Printing

Two-Pointer Technique

Description: Using two pointers to traverse an array, often from opposite ends, to find pairs or triplets that satisfy a condition.

Use Cases: Pair Sum, Triplet Sum, merging sorted arrays.

Requirement : Usually requires a sorted array.

LeetCode Tip: The Two-Pointer technique is highly efficient when used correctly. It reduces time complexity significantly compared to brute-force approaches.*

Two Pointer and Sliding Window Playlist

Strings

Definition: A sequence of characters. Strings are often immutable (cannot be changed after creation).

Common Operations: Slicing, concatenation, character access.

LeetCode Tip: Understand string manipulation techniques, including slicing, concatenation, and character-by-character processing.

Palindrome Check

```
Algorithm IsPalindrome(s):
  Left, right = 0, length(s) - 1
  While left < right:
    If s[left] != s[right]:
      Return False
  left = left + 1
  right = right - 1
Return True
```

Reverse String

```
Algorithm ReverseString(s):
  // s is the string
  new_string = ""
  For i from length(s) - 1 down to 0:
    new_string = new_string + s[i]
  Return new_string
```

Anagram Check

```
Algorithm AreAnagrams(s1, s2):
  // s1 and s2 are the strings
  If length(s1) != length(s2):
    Return False
  char_counts = new Map()  // Character counts
  For each char in s1:
    char_counts[char] = char_counts[char] + 1 or 1  // Increment count
  For each char in s2:
```

```
 If char_counts[char] exists:
   char_counts[char] = char_counts[char] - 1
   If char_counts[char] == 0:
     Remove char_counts[char]


 Else:
   Return False  // Not an anagram
Return char_counts is empty  // Check if all counts are zero
```

SUMMARY TABLE

Topic	Description	Key Operations/Algorithms	Interview Tips
Arrays	Collection of elements of the same type stored in contiguous memory.	Traversal, Insertion, Deletion, Subarray Sum, Two-Pointer Techniques, Sliding Window	Master basic operations; watch for edge cases and array bounds.
Kadane's Algorithm	Finds the maximum sum of a contiguous subarray.	Dynamic Programming approach.	Classic DP problem; understand the principles.
Two-Pointer	Uses two pointers to traverse an array (often sorted) to find pairs/triplets that meet a condition.	Pair Sum, Triplet Sum.	Requires sorted data for efficiency; reduces complexity from brute-force.
Sliding Window	Maintains a dynamic window within an array/string to solve	Fixed Size Window, Variable Size Window.	Focus on window boundaries and conditions for expanding/shrinking.

	problems related to subarrays/substrings.		
Strings	Sequence of characters; often immutable.	Palindrome Check, Reverse String, Anagram Check.	Understand string manipulation techniques (slicing, concatenation).

QUESTIONS (From Strivers)

1. **2 Sum Problem (Easy) - Arrays * : https://leetcode.com/problems/two-sum/description/**
2. **3 Sum Problem (Easy) - Arrays : https://leetcode.com/problems/3sum/description/**
3. **Longest Consecutive Sequence (Medium) - Arrays * : https://leetcode.com/problems/longest-consecutive-sequence/description/**
4. **Merge Intervals (Hard) - Arrays* : https://leetcode.com/problems/merge-intervals/description/**
5. **Roman to Integer (Medium) - Strings * : https://leetcode.com/problems/roman-to-integer/description/**
6. **String to Integer (Hard) - Strings : https://leetcode.com/problems/string-to-integer-atoi/description/**

EXTRA QUESTIONS

1. **Single Number (Easy) -** https://leetcode.com/problems/single-number/
2. **Reverse Vowels of a string (Easy) -** https://leetcode.com/problems/reverse-vowels-of-a-string/
3. **Best Time to Buy and Sell Stock with Cooldown (Medium) -** https://leetcode.com/problems/best-time-to-buy-and-sell-stock/
4. **Reconstruct Original Digits from English (Medium) -** https://leetcode.com/problems/reconstruct-original-digits-from-english/
5. **Spiral Matrix (Medium) -** https://leetcode.com/problems/spiral-matrix
6. **Substring with Concatenation of All words (Hard) -** https://leetcode.com/problems/substring-with-concatenation-of-all-words/
7. **Largest Rectangle in Histogram (Hard) -** https://leetcode.com/problems/largest-rectangle-in-histogram/