MasterClass in DSA

Exercise Set - 1

Topic: Arrays

Total Exercises: 30
Easy: 10 | Intermediate: 10 | Difficult: 10

CoreCode Programming Academy
An Academy by Yogeshwar Shukla

General Instructions

1. Solve each exercise on your PC using your preferred programming language (C, C++, or Python recommended).
2. Unless specified otherwise, do not use built-in sorting functions or additional data structures.
3. Focus on writing clean, efficient code with proper variable naming.
4. Test your solutions with the provided sample inputs and create additional test cases.
5. Exercises are arranged in increasing order of difficulty within each section.

Section A: Easy Exercises

Exercise 1: Sum of All Elements EASY

Write a program that takes an array of integers and calculates the sum of all elements in the array.

Sample Input:

```
arr = [4, 7, 2, 9, 5]
```

Sample Output:

```
27
```

Exercise 2: Count Even Numbers EASY

Write a program that counts how many even numbers are present in an array of integers.

Sample Input:

```
arr = [3, 8, 12, 5, 7, 10, 2]
```

Sample Output:

```
4
```

Exercise 3: Find the Largest Element EASY

Write a program that finds and returns the largest element in an array of integers. Do not use any built-in max function.

Sample Input:

```
arr = [45, 12, 78, 34, 56]
```

Sample Output:

```
78
```

Exercise 4: Find the Smallest Element EASY

Write a program that finds and returns the smallest element in an array of integers. Do not use any built-in min function.

Sample Input:

```
arr = [23, 7, 45, 12, 89, 3]
```

Sample Output:

```
3
```

Exercise 5: Check if Element Exists EASY

Write a program that checks whether a given target element exists in an array. Return "Found" if present, otherwise "Not Found".

Sample Input:

```
arr = [5, 12, 8, 3, 17], target = 8
```

Sample Output:

```
Found
```

Exercise 6: Count Occurrences of a Value EASY

Write a program that counts how many times a specific value appears in an array.

Sample Input:

```
arr = [4, 2, 7, 2, 9, 2, 5], value = 2
```

Sample Output:

```
3
```

Exercise 7: Print Array in Reverse Order EASY

Write a program that prints all elements of an array in reverse order (from last to first). The original array should remain unchanged.

Sample Input:

```
arr = [10, 20, 30, 40, 50]
```

Sample Output:

```
50 40 30 20 10
```

Exercise 8: Find Index of Element EASY

Write a program that finds and returns the index of the first occurrence of a target element in an array. Return -1 if the element is not found.

Sample Input:

```
arr = [15, 8, 22, 9, 17], target = 22
```

Sample Output:

```
2
```

Exercise 9: Calculate Average EASY

Write a program that calculates and returns the average of all elements in an array. Return the result as a floating-point number.

Sample Input:

```
arr = [10, 20, 30, 40, 50]
```

Sample Output:

```
30.0
```

Exercise 10: Copy Array Elements EASY

Write a program that copies all elements from one array to another array (of the same size) using a loop. Do not use any built-in copy functions.

Sample Input:

```
source = [1, 2, 3, 4, 5]
```

Sample Output:

```
destination = [1, 2, 3, 4, 5]
```

Section B: Intermediate Exercises

Exercise 11: Reverse Array In-Place INTERMEDIATE

Write a program to reverse an array without using any additional array. The reversal must happen in the original array itself using element swapping.

Sample Input:

```
arr = [1, 2, 3, 4, 5]
```

Sample Output:

```
arr = [5, 4, 3, 2, 1]
```

Exercise 12: Find Second Largest Element INTERMEDIATE

Write a program to find the second largest element in an array. If all elements are the same, return -1. Do not sort the array.

Sample Input:

```
arr = [12, 35, 1, 10, 34, 1]
```

Sample Output:

```
34
```

Exercise 13: Remove Duplicates (Keep First Occurrence) INTERMEDIATE

Write a program that removes duplicate elements from an array, keeping only the first occurrence of each element. Use only the original array (no additional arrays for storage). Return the new effective length.

Sample Input:

```
arr = [1, 2, 2, 3, 4, 4, 4, 5]
```

Sample Output:

```
arr = [1, 2, 3, 4, 5, ...], new length = 5
```

Exercise 14: Rotate Array Left by K Positions INTERMEDIATE

Write a program to rotate an array to the left by K positions. After rotation, the first K elements should move to the end. Do this in-place without using additional arrays.

Sample Input:

```
arr = [1, 2, 3, 4, 5, 6, 7], K = 3
```

Sample Output:

```
arr = [4, 5, 6, 7, 1, 2, 3]
```

Exercise 15: Find All Pairs with Given Sum INTERMEDIATE

Write a program that finds and prints all pairs of elements in an array whose sum equals a given target value. Each pair should be printed only once.

Sample Input:

```
arr = [2, 4, 3, 5, 7, 8, 1], target = 9
```

Sample Output:

```
(2, 7), (4, 5), (8, 1)
```

Exercise 16: Merge Two Sorted Arrays INTERMEDIATE

Write a program that takes two sorted arrays and merges them into a single sorted array. You may use a third array to store the result.

Sample Input:

```
arr1 = [1, 3, 5, 7], arr2 = [2, 4, 6, 8]
```

Sample Output:

```
merged = [1, 2, 3, 4, 5, 6, 7, 8]
```

Exercise 17: Find Missing Number in Sequence INTERMEDIATE

An array contains N-1 distinct integers in the range 1 to N. Exactly one number is missing. Write a program to find the missing number.

Sample Input:

```
arr = [1, 2, 4, 5, 6], N = 6
```

Sample Output:

```
3
```

Exercise 18: Separate Odd and Even Numbers INTERMEDIATE

Write a program to rearrange an array so that all even numbers appear before all odd numbers. The relative order within even or odd groups does not matter. Do this in-place.

Sample Input:

```
arr = [12, 7, 3, 8, 15, 4, 9, 2]
```

Sample Output:

```
arr = [12, 2, 4, 8, 15, 3, 9, 7] (one possible output)
```

Exercise 19: Check if Array is Sorted INTERMEDIATE

Write a program that checks whether an array is sorted in non-decreasing order. Return "Sorted" or "Not Sorted" accordingly.

Sample Input 1:

```
arr = [1, 2, 2, 4, 5]
```

Sample Output 1:

```
Sorted
```

Sample Input 2:

```
arr = [1, 3, 2, 4, 5]
```

Sample Output 2:

```
Not Sorted
```

Exercise 20: Find Intersection of Two Arrays INTERMEDIATE

Write a program that finds all common elements between two arrays. Each common element should appear in the result only once, even if it appears multiple times in both arrays.

Sample Input:

```
arr1 = [1, 2, 2, 3, 4], arr2 = [2, 2, 3, 5]
```

Sample Output:

```
[2, 3]
```

Section C: Difficult Exercises

Exercise 21: Move All Zeros to End DIFFICULT

Write a program that moves all zeros in an array to the end while maintaining the relative order of non-zero elements. Do this in-place without using any additional array.

Sample Input:

```
arr = [0, 1, 0, 3, 12, 0, 5]
```

Sample Output:

```
arr = [1, 3, 12, 5, 0, 0, 0]
```

Exercise 22: Find the Majority Element DIFFICULT

Write a program to find the majority element in an array. A majority element appears more than N/2 times where N is the array size. If no majority element exists, return -1. Do not use any additional data structures.

Sample Input:

```
arr = [2, 2, 1, 1, 2, 2, 1, 2, 2]
```

Sample Output:

```
2
```

Exercise 23: Rotate Array Right by K Positions (In-Place, Efficient) DIFFICULT

Write a program to rotate an array to the right by K positions using the reversal algorithm. Do this in-place without using additional arrays. Handle the case where K is greater than the array length.

Sample Input:

```
arr = [1, 2, 3, 4, 5, 6, 7], K = 3
```

Sample Output:

```
arr = [5, 6, 7, 1, 2, 3, 4]
```

Exercise 24: Find Maximum Subarray Sum DIFFICULT

Write a program to find the contiguous subarray (containing at least one element) which has the largest sum and return that sum. The array may contain negative numbers.

Sample Input:

```
arr = [-2, 1, -3, 4, -1, 2, 1, -5, 4]
```

Sample Output:

```
6 (subarray [4, -1, 2, 1])
```

Exercise 25: Rearrange Array in Wave Form DIFFICULT

Write a program to arrange an array in wave form such that arr[0] >= arr[1] <= arr[2] >= arr[3] <= arr[4]... and so on. Do this in-place.

Sample Input:

```
arr = [1, 2, 3, 4, 5, 6]
```

Sample Output:

```
arr = [2, 1, 4, 3, 6, 5] (one possible output)
```

Exercise 26: Find First Non-Repeating Element DIFFICULT

Write a program to find the first element in an array that does not repeat. If all elements repeat, return -1. Do not use any additional data structures (use nested loops).

Sample Input:

```
arr = [9, 4, 9, 6, 7, 4, 8, 7]
```

Sample Output:

```
6
```

Exercise 27: Find Equilibrium Index DIFFICULT

Write a program to find the equilibrium index of an array. An equilibrium index is one where the sum of elements on its left equals the sum of elements on its right. Return -1 if no such index exists.

Sample Input:

```
arr = [-7, 1, 5, 2, -4, 3, 0]
```

Sample Output:

```
3 (left sum = -7+1+5 = -1, right sum = -4+3+0 = -1)
```

Exercise 28: Find Longest Consecutive Sequence Length DIFFICULT

Given an unsorted array of integers, find the length of the longest sequence of consecutive integers. The sequence need not be contiguous in the array. You may use nested loops.

Sample Input:

```
arr = [100, 4, 200, 1, 3, 2]
```

Sample Output:

```
4 (sequence: 1, 2, 3, 4)
```

Exercise 29: Dutch National Flag Problem DIFFICULT

An array contains only 0s, 1s, and 2s. Write a program to sort this array in-place so that all 0s come first, then all 1s, then all 2s. Do this in a single pass through the array.

Sample Input:

```
arr = [2, 0, 2, 1, 1, 0, 1, 0, 2]
```

Sample Output:

```
arr = [0, 0, 0, 1, 1, 1, 2, 2, 2]
```

Exercise 30: Find Triplets with Zero Sum DIFFICULT

Write a program to find all unique triplets in an array that sum to zero. Each triplet should be printed only once, and within a triplet, elements should be in non-decreasing order.

Sample Input:

```
arr = [-1, 0, 1, 2, -1, -4]
```

Sample Output:

```
[-1, -1, 2], [-1, 0, 1]
```