**Array Assignments**

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

1. **Given an array of integers nums and an integer target, return indices of the two numbers such that they add up to target.**

**Example 1:**

Input: nums = [2,7,11,15], target = 9

Output: [0,1]

Explanation: Because nums[0] + nums[1] == 9, we return [0, 1].

**Example 2:**

Input: nums = [3,2,4], target = 6

Output: [1,2]

============================================================

1. **Given two sorted arrays nums1 and nums2 of size m and n respectively, return the median of the two sorted arrays.**

Example 1:

Input: nums1 = [1,3], nums2 = [2]

Output: 2.00000

Explanation: merged array = [1,2,3] and median is 2.

Example 2:

Input: nums1 = [1,2], nums2 = [3,4]

Output: 2.50000

===============================================================================

1. **Write a Java program to find the smallest and second smallest elements of a given array.**

**==========================================================**

1. **Write a Java program to cyclically rotate a given array clockwise by n.**

**Input- 1 2 3 4 5 6**

Rotate array by 1

**Output-6 1 2 3 4 5**

Rotate array by 3

**Output-4 5 6 1 2 3**

1. **Write a Java program to cyclically rotate a given array anti-clockwise by n.**

**Input- 1 2 3 4 5 6**

Rotate array by 1

**Output- 2 3 4 5 6 1**

Rotate array by 2

**Output- 5 6 1 2 3 4**

**============================================================**

1. **Write a Java program to separate even and odd numbers from a given array of integers. Put all even numbers first, and then odd numbers.**

**Input – 20 12 23 17 7 8 10 2 1 0**

**Output- 20 12 0 2 10 8 7 17 1 23**

**============================================================**

1. **Write a Java program to replace every element with the next greatest element (from the right side) in a given array of integers.  
   There is no element next to the last element, therefore replace it with -1.**

**Input-** **Original Array**

**[45, 20, 100, 23, -5, 2, -6]**

**Output- The modified array:**

**[100, 100, 23, 2, 2, -6, -1]**

**============================================================8) Write a Java program to sort a binary array in linear time.    
From Wikipedia,  
Linear time: An algorithm is said to take linear time, or O(n) time, if its time complexity is O(n). Informally, this means that the running time increases at most linearly with the size of the input. More precisely, this means that there is a constant c such that the running time is at most cn for every input of size n. For example, a procedure that adds up all elements of a list requires time proportional to the length of the list, if the adding time is constant, or, at least, bounded by a constant.  
Linear time is the best possible time complexity in situations where the algorithm has to sequentially read its entire input. Therefore, much research has been invested into discovering algorithms exhibiting linear time or, at least, nearly linear time. This research includes both software and hardware methods. There are several hardware technologies which exploit parallelism to provide this. An example is content-addressable memory. This concept of linear time is used in string matching algorithms such as the Boyer–Moore algorithm and Ukkonen's algorithm.**

**Example:  
Input :  
b\_nums[] = { 0, 1, 1, 0, 1, 1, 0, 1, 0, 0 }  
Output:  
After sorting: [0, 0, 0, 0, 0, 1, 1, 1, 1, 1]**

1. **Write a Java program to find the maximum product of two integers in a given array of integers.**

**Example:  
Input :  
nums = { 2, 3, 5, 7, -7, 5, 8, -5 }  
Output:  
Pair is (7, 8), Maximum Product: 56**

===========================================================

1. **Write a Java program to shuffle a given array of integers. (to shuffle an array using the Fisher-Yates algorithm also you can use Random class from java.util.\*; package and their methods)**

**Example:  
Input :  
nums = { 1, 2, 3, 4, 5, 6 }  
Output:  
Shuffle Array: [4, 2, 6, 5, 1, 3]**

**============================================================**

**11)  Write a Java program to replace each element of the array with the product of every other element in a given array of integers.**

**Example:  
Input :  
nums1 = { 1, 2, 3, 4, 5, 6, 7}  
Output:  
Array with product of every other element:  
[5040, 2520, 1680, 1260, 1008, 840, 720]**

**Input :  
nums2 = {0, 1, 2, 3, 4, 5, 6, 7}  
Array with product of every other element:  
[5040, 0, 0, 0, 0, 0, 0, 0]**

**============================================================**

1. **Write a Java program to calculate the largest gap between sorted elements of an array of integers.**

**Example:  
Original array: [23, -2, 45, 38, 12, 4, 6]  
Largest gap between sorted elements of the said array: 15**

1. **Given an array of N integers where each value represents the number of chocolates in a packet. Each packet can have a variable number of chocolates. There are m students, the task is to distribute chocolate packets such that:**

**Each student gets one packet.**

**The difference between the number of chocolates in the packet with maximum chocolates and the packet with minimum chocolates given to the students is minimum.**

**Examples:**

***Input : arr[] = {7, 3, 2, 4, 9, 12, 56} , m = 3   
Output: Minimum Difference is 2   
Explanation:  
We have seven packets of chocolates and we need to pick three packets for 3 students   
If we pick 2, 3 and 4, we get the minimum difference between maximum and minimum packet sizes.***

***Input : arr[] = {3, 4, 1, 9, 56, 7, 9, 12} , m = 5   
Output: Minimum Difference is 6***

***Input : arr[] = {12, 4, 7, 9, 2, 23, 25, 41, 30, 40, 28, 42, 30, 44, 48, 43, 50} , m = 7   
Output: Minimum Difference is 10***

1. **Given an array prices[] of size N denoting the cost of stock on each day, the task is to find the maximum total profit if we can buy and sell the stocks any number of times.**

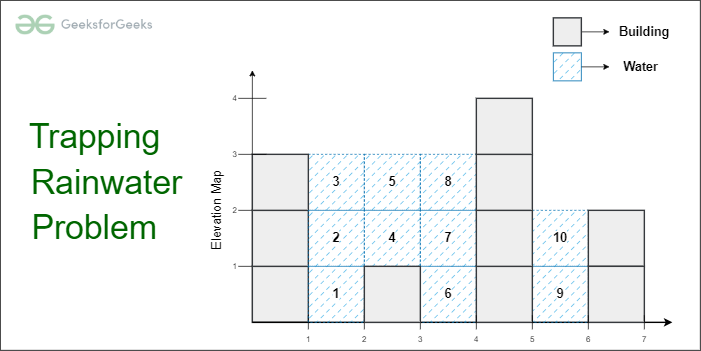
**Note: We can only sell a stock which we have bought earlier and we cannot hold multiple stocks on any day.**

**Examples:**

***Input: prices[] = {100, 180, 260, 310, 40, 535, 695}  
Output: 865  
Explanation: Buy the stock on day 0 and sell it on day 3 => 310 – 100 = 210  
                       Buy the stock on day 4 and sell it on day 6 => 695 – 40 = 655  
                       Maximum Profit  = 210 + 655 = 865***

***Input: prices[] = {4, 2, 2, 2, 4}  
Output: 2  
Explanation: Buy the stock on day 4 and sell it on day 5 => 4 – 2 = 2  
                       Maximum Profit  = 2***

***15* ) Trapping Rainwater Problem states that given an array of N non-negative integers arr[] representing an elevation map where the width of each bar is 1, compute how much water it can trap after rain.**



Trapping Rainwater Problem

**Examples:**

**Let us understand Trapping Rainwater problem with the help of some examples:**

**Input: arr[] = {3, 0, 1, 0, 4, 0, 2}  
Output: 10  
Explanation: The expected rainwater to be trapped is shown in the above image.**

**Input: arr[]   = {3, 0, 2, 0, 4}  
Output: 7  
Explanation: Structure is like below.  
We can trap “3 units” of water between 3 and 2,  
“1 unit” on top of bar 2 and “3 units” between 2 and 4.**

**============================================================**

1. **Write a java programs to print all primes smaller than or equal to n using Sieve of Eratosthenes algorithm**

**Input – n=10**

**Output- 2 3 5 7**

**Input- n=20**

**Output- 2 3 5 7 11 13 17 19**