Problem 1: Given an integer array nums, return all the triplets [nums[i], nums[j], nums[k]] such that i != j, i != k, and j != k, and nums[i] + nums[j] + nums[k] == 0.

Notice that the solution set must not contain duplicate triplets.

Test Cases: 1
Input: nums = [-1,0,1,2,-1,-4]
Output : [[-1,-1,2],[-1,0,1]]

Test Case 2:
Input: nums = [0,1,1]
Output: []

Explanation: The only possible triplet does not sum up to 0.

Test Case 3:
Input: nums = [0,0,0]
Output: [[0,0,0]]

Explanation: The only possible triplet sums up to 0.

Problem 2: Given an array of positive and negative numbers, arrange them such that all negative integers appear before all the positive integers in the array without using any additional data structure like a hash table, arrays, etc. The order of appearance should be Maintained.

Test case 1:
Input: -12, 11, -13, -5, 6, -7, 5, -3, -6
Output: -12 -13 -5 -7 -3 -6 11 6 5

Test case 2:
Input: -12, 11, 13, -5, 6, -7, 5, -3, 8
Output: -12 -5 -7 -3 11 13 6 5 8

Problem 3: Given an array of N integers and a number K, the task is to find the number of pairs of integers in the array whose sum is equal to K.

Test Case 1:

Input: arr[] = {1, 5, 7, -1}, sum = 6
Output: 2

Explanation: Pairs with sum 6 are (1, 5) and (7, -1).

Test Case 2:

Input: arr[] = {1, 5, 7, -1, 5}, sum = 6
Output: 3

Explanation: Pairs with sum 6 are (1, 5), (7, -1) & (1, 5).

Problem 4: 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:
a. Each student gets one packet.
b. 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.

Test Case 1
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.

Test Case 2
Input: arr[] = {3, 4, 1, 9, 56, 7, 9, 12}, m = 5
Output: Minimum Difference is 6

Problem 5: Seven different symbols represent Roman numerals with the following values:

Symbol	Value
I	1
V	5
X	10
L	50
C	100
D	500
M	1000

Roman numerals are formed by appending the conversions of decimal place values from highest to lowest. Converting a decimal place value into a Roman numeral has the following rules:

- If the value does not start with 4 or 9, select the symbol of the maximal value that can be subtracted from the input, append that symbol to the result, subtract its value, and convert the remainder to a Roman numeral.
- If the value starts with 4 or 9 use the **subtractive form** representing one symbol subtracted from the following symbol, for example, 4 is 1 (I) less than 5 (V): IV and 9 is 1 (I) less than 10 (X): IX. Only the following subtractive forms are used: 4 (IV), 9 (IX), 40 (XL), 90 (XC), 400 (CD) and 900 (CM).

- Only powers of 10 (I, X, C, M) can be appended consecutively at most 3 times to represent multiples of 10. You cannot append 5 (V), 50 (L), or 500 (D) multiple times. If you need to append a symbol 4 times use the **subtractive form**

Given an integer, convert it to a Roman numeral.

Test Case 1:
Input: num = 3749
Output: "MMMDCCXLIX"

Explanation:
3000 = MMM as 1000 (M) + 1000 (M) + 1000 (M)
700 = DCC as 500 (D) + 100 (C) + 100 (C)
40 = XL as 10 (X) less of 50 (L)
9 = IX as 1 (I) less of 10 (X)

Note: 49 is not 1 (I) less of 50 (L) because the conversion is based on decimal places

Test Case 2:
Input: num = 58
Output: "LVIII"

Explanation:
50 = L
8 = VIII

Test Case 3:
Input: num = 1994
Output: "MCMXCIV"

Explanation:
1000 = M
900 = CM
90 = XC
4 = IV

Constraints: 1 <= num <= 3999

Problem 6: Roman numerals are represented by seven different symbols: I, V, X, L, C, D and M.

Symbol	**Value**
I	1
V	5
X	10
L	50
C	100
D	500
M	1000

For example, 2 is written as II in Roman numerals, just two ones added together. 12 is written as XII, which is simply X + II. The number 27 is written as XXVII, which is XX + V + II.

Roman numerals are usually written largest to smallest from left to right. However, the numeral for four is not IIII. Instead, the number four is written as IV. Because the one is before the five we subtract it making four. The same principle applies to the number nine, which is written as IX. There are six instances where subtraction is used:

- I can be placed before V (5) and X (10) to make 4 and 9.
- X can be placed before L (50) and C (100) to make 40 and 90. C can be placed before D (500) and M (1000) to make 400 and 900.

Given a roman numeral, convert it to an integer.

Test Case 1:
Input: s = "III"
Output: 3

Explanation: III = 3.

Test Case 2:
Input: s = "LVIII"
Output: 58

Explanation: L = 50, V= 5, III = 3.

Test Case 3:
Input: s = "MCMXCIV"
Output: 1994

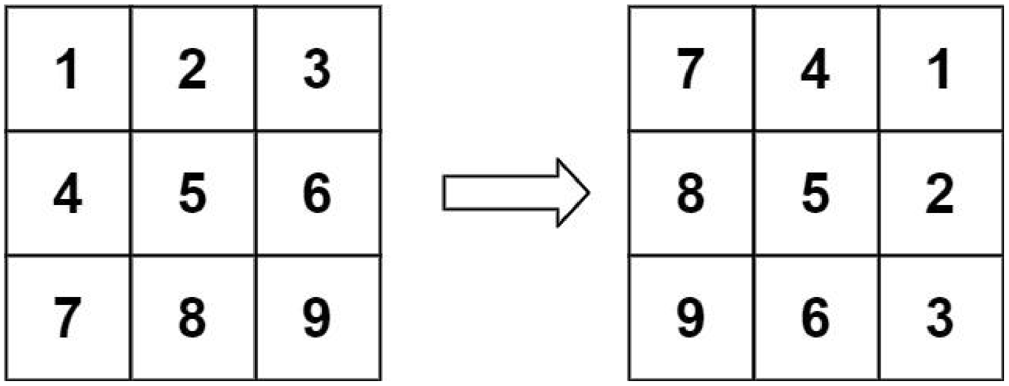
Explanation: M = 1000, CM = 900, XC = 90 and IV = 4.

Constraints:

- 1 <= s.length <= 15
- s contains only the characters ('I', 'V', 'X', 'L', 'C', 'D', 'M').
- It is **guaranteed** that s is a valid roman numeral in the range [1, 3999]

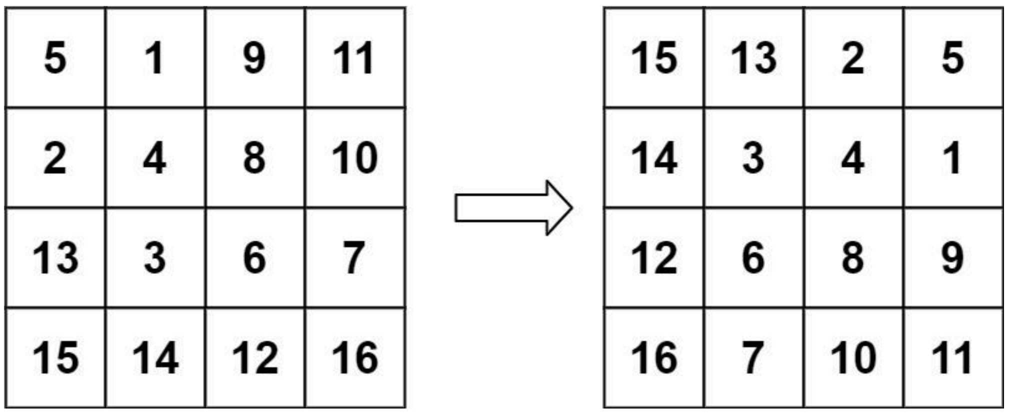
Problem 7: You are given an n x n 2D matrix representing an image, rotate the image by **90** degrees (clockwise). You have to rotate the image **in-place**, which means you have to modify the input 2D matrix directly.**DO NOT** allocate another 2D matrix and do the rotation.

Example 1:



Input: matrix = [[1,2,3],[4,5,6],[7,8,9]]
Output: [[7,4,1],[8,5,2],[9,6,3]]

Example 2:



Input: matrix = [[5,1,9,11],[2,4,8,10],[13,3,6,7],[15,14,12,16]]
Output: [[15,13,2,5],[14,3,4,1],[12,6,8,9],[16,7,10,11]]

Constraints:

- n == matrix.length == matrix[i].length
- 1 <= n <= 20
- -1000 <= matrix[i][j] <= 1000