Code -collections

**1. Two Sum**

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

**You may assume that each input would have exactly one solution, and you may not use the same element twice.**

**You can return the answer in any order.**

**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]**

**Example 3:**

**Input: nums = [3,3], target = 6**

**Output: [0,1]**

**Solution:**

**class Main {**

**public static void main(String[] args) {**

**int[] arr = {2, 7, 11, 15};**

**int target = 9;**

**for (int i = 0; i < arr[i]; i++) {**

**for (int j = i + 1; j < arr[j]; j++)**

**{**

**if (arr[i] + arr[j] == target) {**

**System.out.println("\n hence the Array will be : [" + arr[i] + ", " + arr[j] + "]");**

**System.out.println("AT Index: [" + i + ", " + j + "]");**

**return;**

**}**

**}**

**}**

**System.out.println("No solution found.");**

**}**

**}**

**---------------------------------------------------------------------------------------------------------**

**Given an integer x, return true if x is a**

**palindrome**

**, and false otherwise.**

**Example 1:**

**Input: x = 121**

**Output: true**

**Explanation: 121 reads as 121 from left to right and from right to left.**

**Example 2:**

**Input: x = -121**

**Output: false**

**Explanation: From left to right, it reads -121. From right to left, it becomes 121-. Therefore it is not a palindrome.**

**Example 3:**

**Input: x = 10**

**Output: false**

**Explanation: Reads 01 from right to left. Therefore it is not a palindrome.**

[**https://leetcode.com/problems/palindrome-number**](https://leetcode.com/problems/palindrome-number)**.**

**Solution:**

**class Solution {**

**public boolean isPalindrome(int x) {**

**if (x < 0) {**

**return false;**

**}**

**int temp = x, rev = 0;**

**while (x > 0) {**

**int rem = x % 10;**

**rev = rev \* 10 + rem;**

**x = x / 10;**

**}**

**return temp == rev;**

**}**

**}**

**You are given a string s consisting of lowercase English letters, and an integer k. Your task is to convert the string into an integer by a special process, and then transform it by summing its digits repeatedly k times. More specifically, perform the following steps:**

**1. Convert s into an integer by replacing each letter with its position in the alphabet (i.e. replace 'a' with 1, 'b' with 2, ..., 'z' with 26).**

**2. Transform the integer by replacing it with the sum of its digits.**

**3. Repeat the transform operation (step 2) k times in total.**

**For example, if s = "zbax" and k = 2, then the resulting integer would be 8 by the following operations:**

**1. Convert: "zbax" ➝ "(26)(2)(1)(24)" ➝ "262124" ➝ 262124**

**2. Transform #1: 262124 ➝ 2 + 6 + 2 + 1 + 2 + 4 ➝ 17**

**3. Transform #2: 17 ➝ 1 + 7 ➝ 8**

**Return the resulting integer after performing the operations described above.**

**Example :**

**Input: s = "zbax", k = 2**

**Output: 8**

**class Main {**

**public static void main(String[] args) {**

**String[] str = {"zbax"};**

**int k = 2;**

**int result = trString(str[0], k);**

**System.out.println("The sumof (53=5+3) is: "+result);**

**}**

**public static int trString(String str, int k) {**

**int n = 0;**

**for (int i = 0; i < str.length(); i++) {**

**n += (str.charAt(i) - 'a' + 1);**

**}**

**for (int i = 0; i < k; i++) {**

**n = digitSum(n);**

**}**

**return n;**

**}**

**public static int digitSum(int n) {**

**int sum = 0;**

**while (n > 0) {**

**sum += n % 10;**

**n /= 10;**

**}**

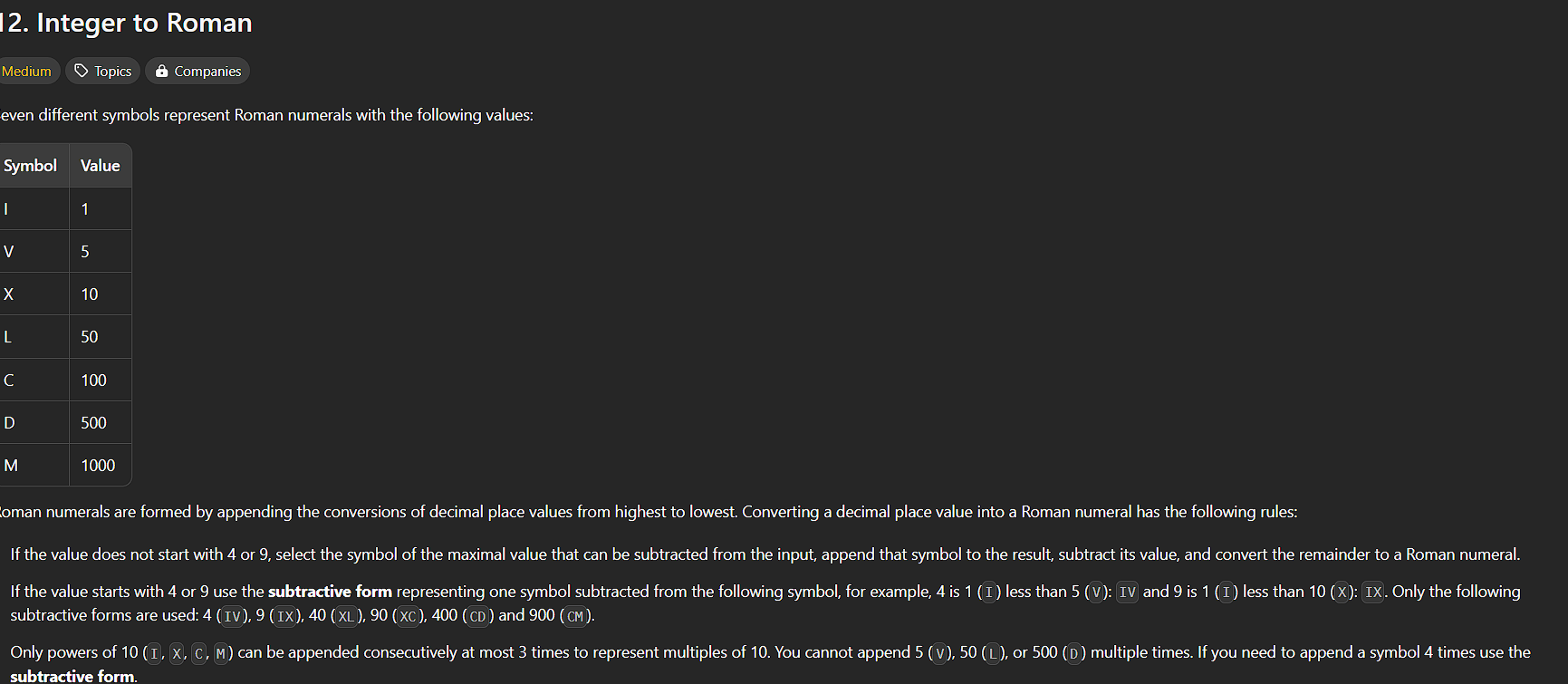
**return sum;**

**}**

**}**

**4**

|  |
| --- |
|  |

****

<https://leetcode.com/problems/integer-to-roman>

**https://leetcode.com/problems/integer-to-roman**

**Solution:**

**public class Solution {**

**public String intToRoman(int num) {**

**int[] val = {1000, 900, 500, 400, 100, 90, 50, 40, 10, 9, 5, 4, 1};**

**String[] symbols = {"M", "CM", "D", "CD", "C", "XC", "L", "XL", "X", "IX", "V", "IV", "I"};**

**StringBuilder roman = new StringBuilder();**

**for (int i = 0; i < val.length; i++) {**

**while (num >= val[i]) {**

**roman.append(symbols[i]);**

**num -= val[i];**

**}**

**}**

**return roman.toString();**

**}**

**}**

**Note: had referred gpt in order to take help to append the string without creating the new function for it .**