**DSA Coding Problems**

**1.Next Permutation:**

Given an array of integers **arr[]** representing a permutation, implement the **next permutation** that rearranges the numbers into the lexicographically next greater permutation. If no such permutation exists, rearrange the numbers into the lowest possible order (i.e., sorted in ascending order).

Note - A permutation of an array of integers refers to a specific arrangement of its elements in a sequence or linear order.

**Examples:**

**Input:** arr = [2, 4, 1, 7, 5, 0]

**Output:** [2, 4, 5, 0, 1, 7]

**Explanation:** The next permutation of the given array is {2, 4, 5, 0, 1, 7}.

**Input:** arr = [3, 2, 1]

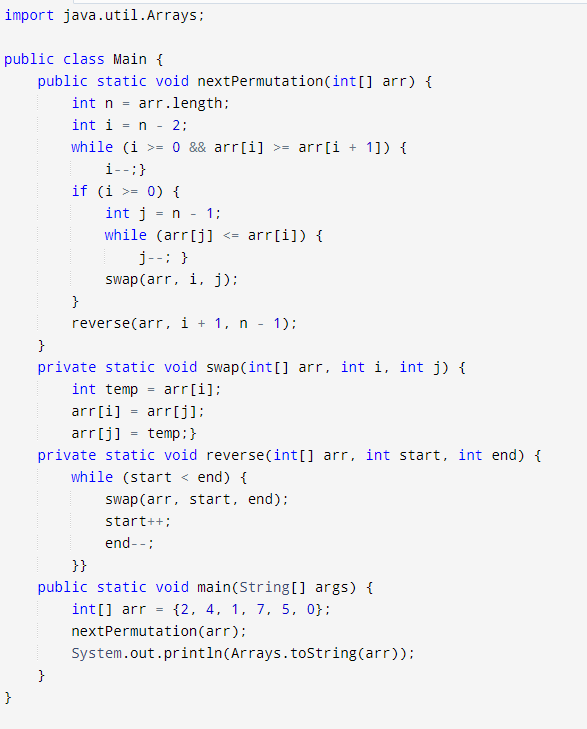
**Output:** [1, 2, 3]

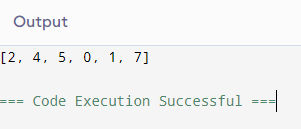
**Explanation:** As arr[] is the last permutation, the next permutation is the lowest one.

**Input:** arr = [3, 4, 2, 5, 1]

**Output:** [3, 4, 5, 1, 2]

**Explanation:** The next permutation of the given array is {3, 4, 5, 1, 2}.





**2.Spiral Matrix**

Given a matrix of size**N x M**. You have to find the **Kth** element which will obtain while traversing the matrix **spirally** starting from the top-left corner of the matrix.

**Example 1:**

**Input:**

N = 3, M = 3, K = 4

A[] = {{1, 2, 3},

{4, 5, 6},

{7, 8, 9}}

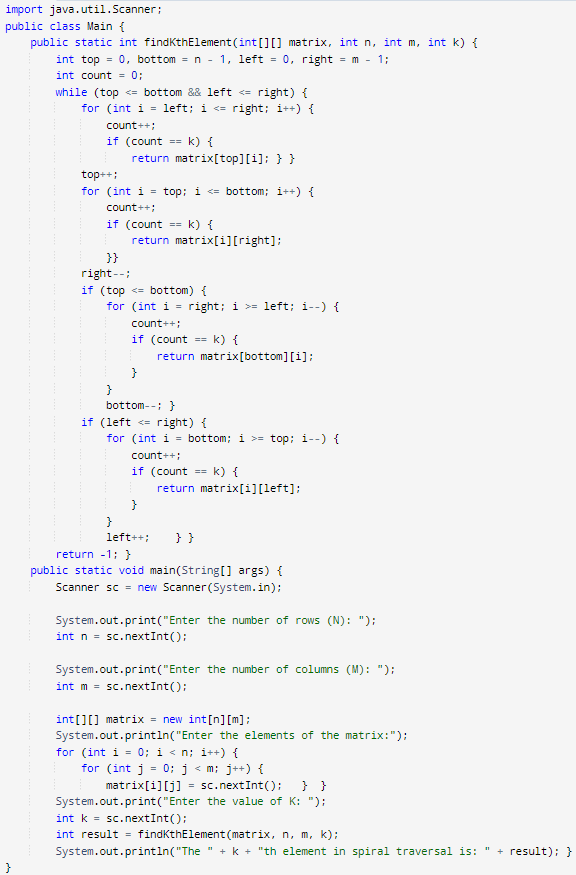
**Output:**

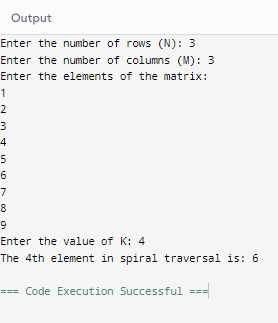
6

**Explanation:** Spiral traversal of matrix:

{1, 2, 3, 6, 9, 8, 7, 4, 5}. Fourth element

is 6.





**3. Longest substring without repeating characters**

Given a string **s**, you need to print the size of the longest possible substring with exactly**k unique** characters. If no possible substring exists, print -1.

**Examples:**

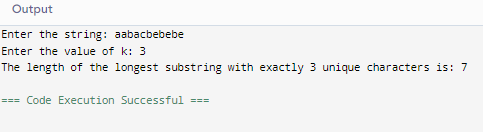
**Input:**

s = "aabacbebebe", k = 3

**Output:** 7

**Explanation**:   
"cbebebe" is the longest substring with 3 distinct characters.





**4. Remove linked list elements**

Given a singly linked list, delete the **middle**of the linked list.

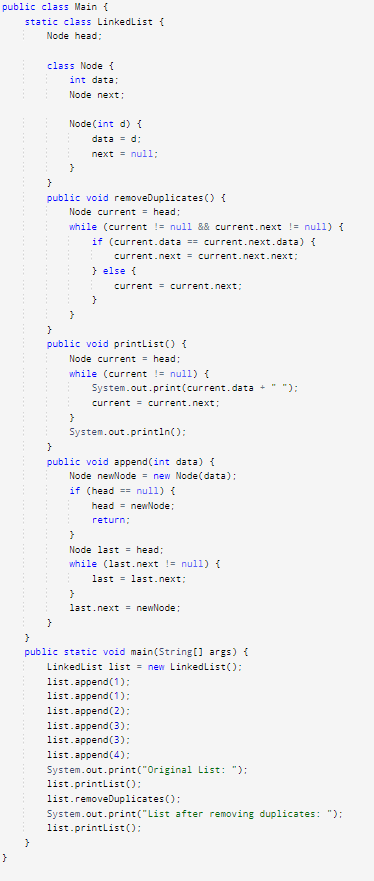
Note:

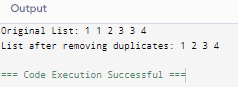
* If there are **even** nodes, then there would be two middle nodes, we need to delete the **second**middle element.
* If the input linked list has a single node, then it should return **NULL.**

**Examples:**

**Input:** LinkedList: 1->2->3->4->5

**Output:** 1->2->4->5





**5.Palindrome linked List**

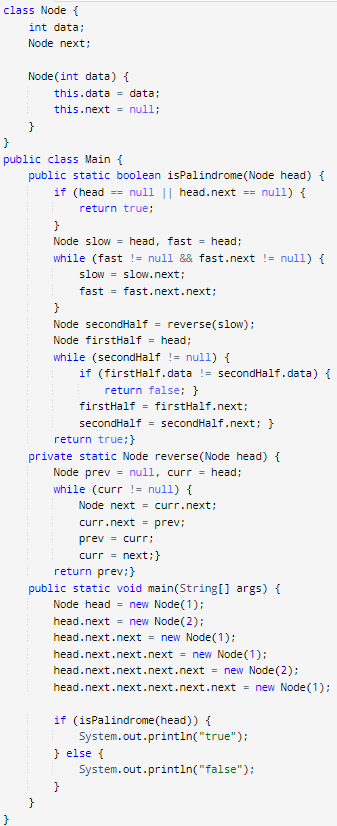
Given a singly linked list of integers. The task is to check if the given linked list is palindrome or not.

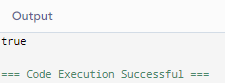
Examples:

Input: LinkedList: 1->2->1->1->2->1

Output: true

Explanation: The given linked list is 1->2->1->1->2->1 , which is a palindrome and Hence, the output is true.

****

****

**6.** **Minimum path sum**

Given an array **arr[]** such that each element is in the range [0 - 9], find the minimum possible sum of two numbers formed using the elements of the array. All digits in the given array must be used to form the two numbers. Return a string without leading zeroes.

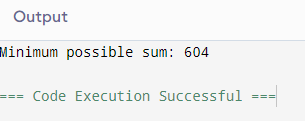
**Examples :**

**Input:** arr[] = [6, 8, 4, 5, 2, 3]

**Output:** 604

**Explanation:** The minimum sum is formed by numbers 358 and 246.





**7. Validate binary search tree**

Given an array of integers **arr[]** representing **inorder traversal** of elements of a binary tree. Return true if the given inorder traversal can be of a valid Binary Search Tree.

Note - In a valid Binary Search Tree all keys are unique.

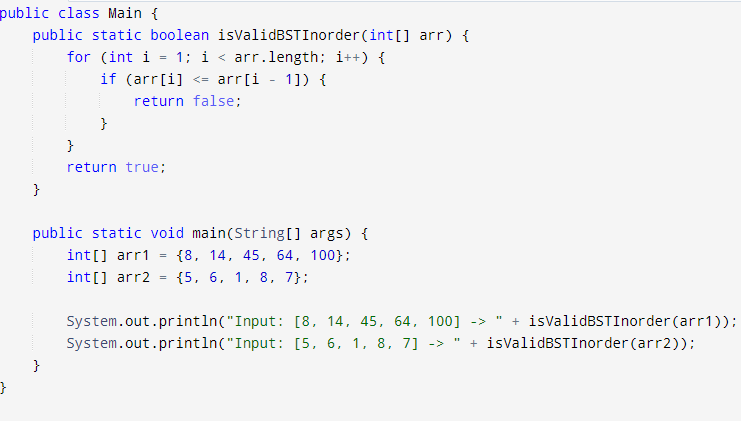
**Examples :**

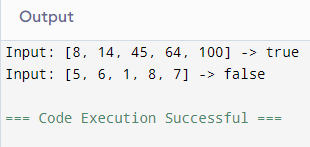
**Input:** arr[] = [8, 14, 45, 64, 100]

**Output:** True

**Input:** arr[] = [5, 6, 1, 8, 7]

**Output:** False





**8. Word ladder**

Given two distinct words **startWord** and **targetWord**, and a list denoting **wordList** of unique words of equal lengths. Find the length of the shortest transformation sequence from startWord to targetWord.  
Keep the following conditions in mind:

* A word can only consist of lowercase characters.
* Only one letter can be changed in each transformation.
* Each transformed word must exist in the wordList including the targetWord.
* startWord may or may not be part of the wordList

The second part of this problem can be found [here](https://practice.geeksforgeeks.org/problems/word-ladder-ii/1/).

**Note**: If no possible way to transform sequence from startWord to targetWord **return 0**

**Example 1:**

**Input:**

wordList = {"des","der","dfr","dgt","dfs"}

startWord = "der", targetWord= "dfs",

**Output:**

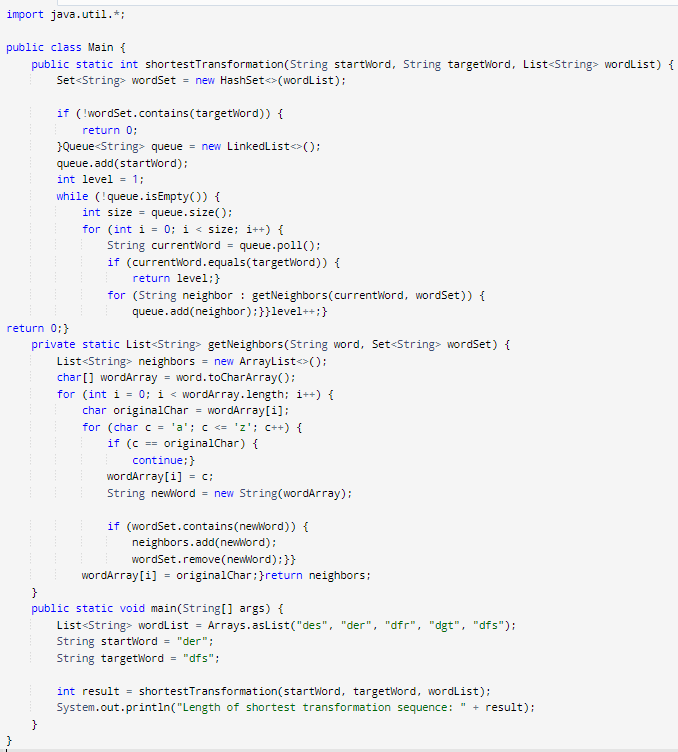
3

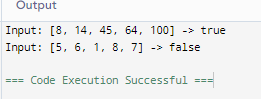
**Explanation:**

The length of the smallest transformation

sequence from "der" to "dfs" is 3

i,e "der" -> "dfr" -> "dfs".





**9. Word ladder -II**

Given a 2D board of letters and a word. Check if the word exists in the board. The word can be constructed from letters of adjacent cells only. ie - horizontal or vertical neighbors. The same letter cell can not be used more than once.

Example 1:

Input: board = {{a,g,b,c},{q,e,e,l},{g,b,k,s}},

word = "geeks"

Output: 1

Explanation: The board is-

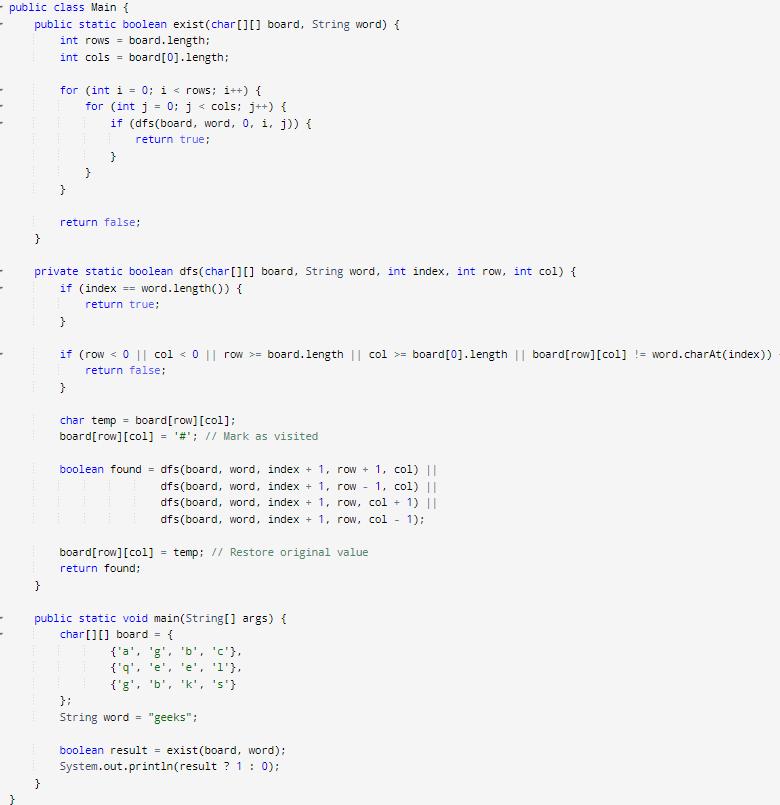
a g b c

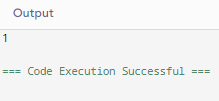
q e e l

g b k s

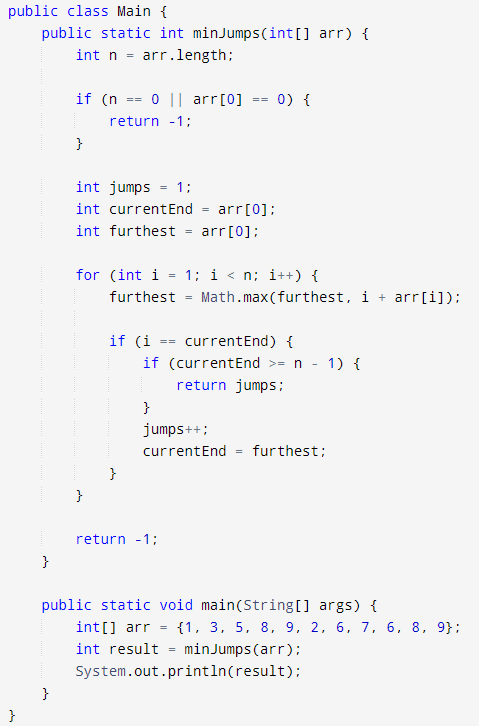
The letters which are used to make the

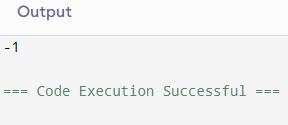
"geeks" are colored.

****

****

**10.** **Course schedule**

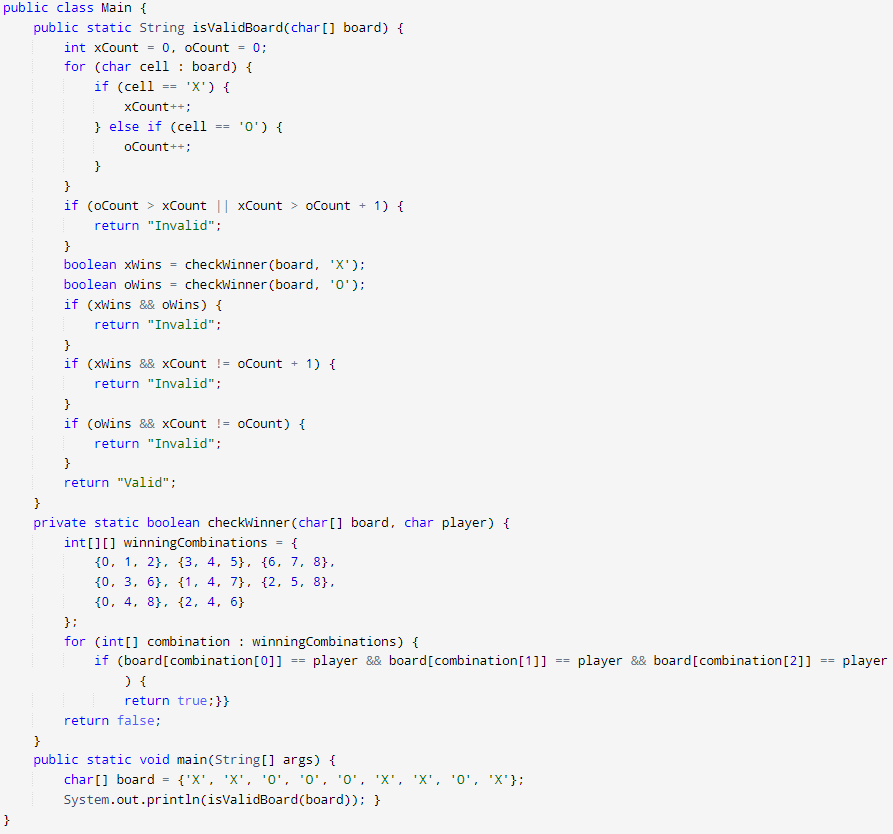
****

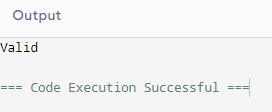
****

**11.** **Design tic tac toe**

A Tic-Tac-Toe board of size 3X3 is given after all the moves are played, i.e., all nine spots are filled. Find out if the given board is valid, i.e., is it possible to reach this board position after a set of moves or not.  
Note that every arbitrarily filled grid of 9 spaces isn’t valid, e.g., a grid filled with 3 **X** and 6 **O** isn’t a valid situation because each player needs to take alternate turns.

**Note:**The game starts with X

****

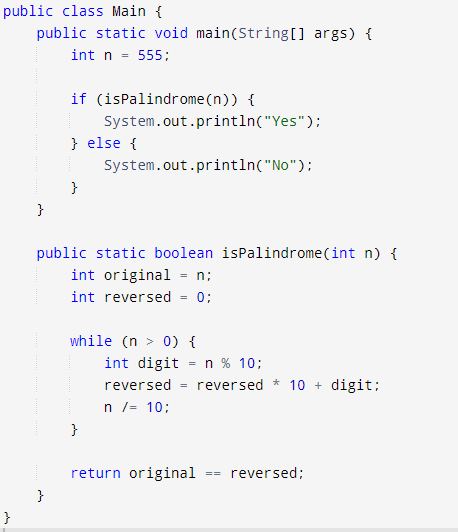
****

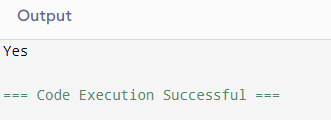
**12.Palindrome**

Note : Strings which read the same backwards as forwards , eg - MADAM, MOM etc.  
  
Example 1:

Input: n = 555

Output: Yes

****

****

**13.Is Subsquence**

Given two strings, **s1** and **s2**, count the number of subsequences of string **s1** equal to string **s2**.

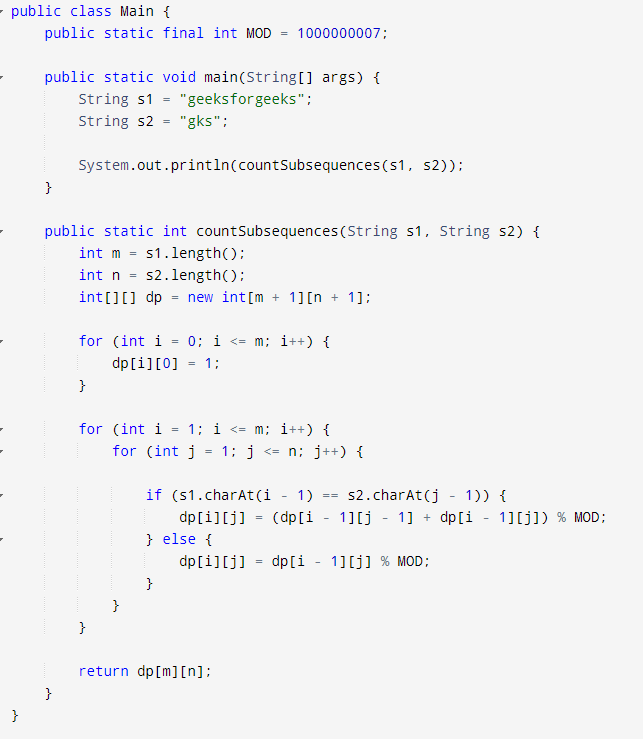
Return the total count modulo **1e9+7.**

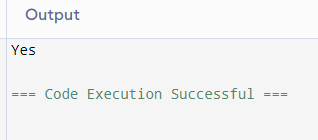
**Example 1:**

**Input:** s1 = geeksforgeeks, s2 = gks

**Output:** 4

**Explaination:** We can pick characters from s1 as a subsequence from indices {0,3,4}, {0,3,12}, {0,11,12} and {8,11,12}.So total 4 subsequences of s1 that are equal to s2.

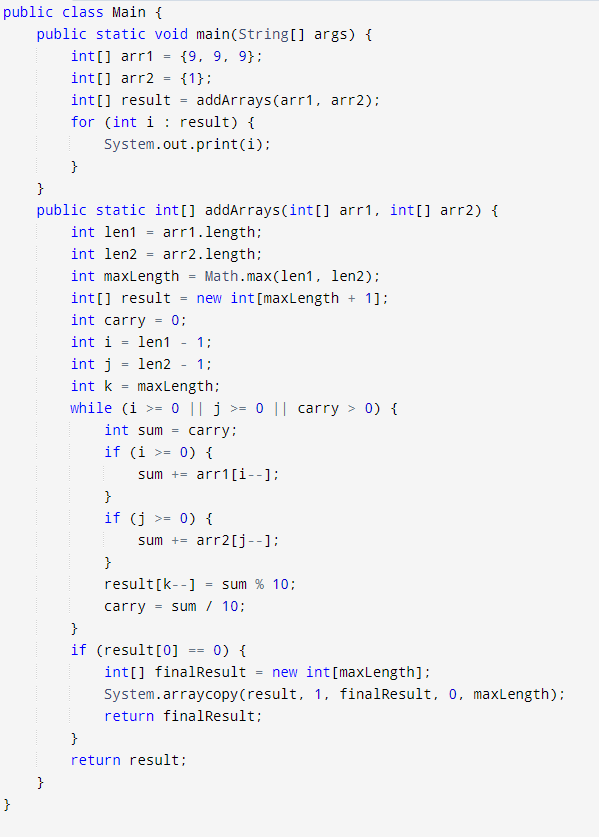


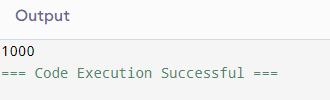


**14.Sum of 2 Number**

Given two numbers represented by two different arrays, **arr1[]** and **arr2[]**, the task is to find their sum as a new array. Each array represents a number where each element corresponds to a digit in that number. The resulting sum array should also represent the sum of the two numbers in the same digit-by-digit format.

**Note**: No leading zeroes in array arr1 and arr2.





**15**.**Container with Most Water**

Given non-negative integers arr1,arr2,....arrn where each represents a point at coordinate (i, arri). For each i vertical lines are drawn such that the two endpoints of line**i**is at (i, arri) and (i,0). Find two lines, which together with x-axis form a container, such that it contains the most water.

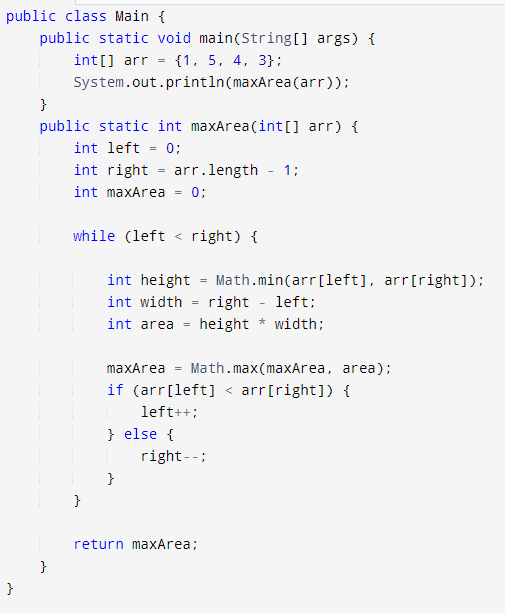
Note: In the case of a single verticle line it will not be able to hold water.

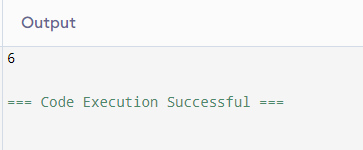
**Examples:**

**Input:** arr[] = [1, 5, 4, 3]

**Output:** 6

**Explanation:** 5 and 3 are distance 2 apart. So the size of the base = 2. Height of container = min(5, 3) = 3. So total area = 3 \* 2 = 6.





**16.3Sum**

Given an array **A[]** of **N** integers and an integer **X**. The task is to find the sum of three integers in **A[]** such that it is closest to **X**.

**Example 1:**

**Input:**

N = 4

A[] = {-1 , 2, 1, -4}

X = 1

**Output:** 2

**Explaination:**

Sums of triplets:

(-1) + 2 + 1 = 2

(-1) + 2 + (-4) = -3

2 + 1 + (-4) = -1

2 is closest to 1.

