1. Find the number of digits in a number

Description: Find how many digits are in a given number.

Example:

Input: 12345

Output: 5

1. Print the Diagonal Elements of a Matrix:

Description: Extract the diagonal elements of a square matrix.

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

Output: [1, 5, 9]

1. Write a function to find the longest common prefix string amongst an array of strings.

If there is no common prefix, return an empty string "".

Example 1:

Input: strs = ["flower","flow","flight"]

Output: "fl"

Example 2:

Input: strs = ["dog","racecar","car"]

Output: ""

Explanation: There is no common prefix among the input strings.

1. **Find the number of digits in a number**

**Description: Find how many digits are in a given number.**

**Example:**

**Input: 12345**

**Output: 5**

1. **Print the Diagonal Elements of a Matrix:**

**Description: Extract the diagonal elements of a square matrix.**

**Input: [[1, 2, 3], [4, 5, 6], [7, 8, 9]]**

**Output: [1, 5, 9]**

1. **Write a function to find the longest common prefix string amongst an array of strings.**

**If there is no common prefix, return an empty string "".**

**Example 1:**

**Input: strs = ["flower","flow","flight"]**

**Output: "fl"**

**Example 2:**

**Input: strs = ["dog","racecar","car"]**

**Output: ""**

**Explanation: There is no common prefix among the input strings.**

**14. Find the first non-repeated character in a string    25**

**Description: Find the first character in a string that does not repeat.**

**Example:**

**Input: "swiss"**

**Output: 'w'**

1. **Given an array arr[ ] of integers, the task is to find the next greater element for each element of the array in order of their appearance in the array. Next greater element of an element in the array is the nearest element on the right which is greater than the current element.  
   If there does not exist next greater of current element, then next greater element for current element is -1. For example, next greater of the last element is always -1.**

**Examples**

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

**Output: [3, 4, 4, -1]**

**Explanation: The next larger element to 1 is 3, 3 is 4, 2 is 4 and for 4, since it doesn't exist, it is -1.**

**Input: arr[] = [6, 8, 0, 1, 3]**

**Output: [8, -1, 1, 3, -1]**

**Explanation: The next larger element to 6 is 8, for 8 there is no larger elements hence it is -1, for 0 it is 1 , for 1 it is 3 and then for 3 there is no larger element on right and hence -1.**

**Input: arr[] = [10, 20, 30, 50]**

**Output: [20, 30, 50, -1]**

**Explanation: For a sorted array, the next element is next greater element also exxept for the last element.**

**Input: arr[] = [50, 40, 30, 10]**

**Output: [-1, -1, -1, -1]**

**Explanation: There is no greater element for any of the elements in the array, so all are -1.**

1. **Check if a number is a factor of another number    15**

**Description: Check if one number is a factor of another number.**

**Example:**

**Input: 10, 20**

**Output: True (10 is a factor of 20)**

1. **Find the product of digits of a number  15**

**Description: Calculate the product of the digits of a given number.**

**Example:**

**Input: 123**

**Output: 6 (1 \* 2 \* 3 = 6)**

1. **Sum of perfect squares in a range:  25**

**Description: Calculate the sum of all perfect squares between two numbers.**

**Example:**

**Input: 1 to 10**

**Output: 1 + 4 + 9 = 14**

1. **Given an array of integers nums sorted in non-decreasing order, find the starting and ending position of a given target value.**

**If target is not found in the array, return [-1, -1].**

**You must write an algorithm with O(log n) runtime complexity.**

**Example 1:**

**Input: nums = [5,7,7,8,8,10], target = 8**

**Output: [3,4]**

**Example 2:**

**Input: nums = [5,7,7,8,8,10], target = 6**

**Output: [-1,-1]**

**Example 3:**

**Input: nums = [], target = 0**

**Output: [-1,-1]**

1. **Check if a number is a factor of another number**

**Description: Check if one number is a prime factor of another number.**

**Example:**

**Input: 10, 20**

**Output: True (10 is a factor of 20)**

1. **Remove duplicates from an array:**

**Description: Remove all duplicate elements from an array.**

**Example:**

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

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

1. **Given an array of integers nums sorted in non-decreasing order, find the starting and ending position of a given target value.**

**If target is not found in the array, return [-1, -1].**

**You must write an algorithm with O(log n) runtime complexity.**

**Example 1:**

**Input: nums = [5,7,7,8,8,10], target = 8**

**Output: [3,4]**

**Example 2:**

**Input: nums = [5,7,7,8,8,10], target = 6**

**Output: [-1,-1]**

**Example 3:**

**Input: nums = [], target = 0**

**Output: [-1,-1]**

1. **Find the length of a string without using built-in functions   15**

**Description: Manually count the number of characters in a string.**

**Example:**

**Input: "hello"**

**Output: 5**

1. **Group All Zeroes to the End of an Array:   25**

**Description: Move all zero elements in an array to the end while maintaining the order of non-zero elements.**

**Input: [0, 1, 2, 0, 3]**

**Output: [1, 2, 3, 0, 0]**

1. **Write a function to find the longest common prefix string amongst an array of strings.**

**If there is no common prefix, return an empty string "".**

**Example 1:**

**Input: strs = ["flower","flow","flight"]**

**Output: "fl"**

**Example 2:**

**Input: strs = ["dog","racecar","car"]**

**Output: ""**

**Explanation: There is no common prefix among the input strings.**

1. **Reverse a string  15**

**Description: Write a program to reverse the characters of a given string.**

**Example:**

**Input: "hello"**

**Output: "olleh"**

1. **Find the Number of Unique Elements in an Array:   25**

**Description: Count the distinct elements in an array.**

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

1. **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]**

1. **Check if a string is a palindrome  15**

**Description: Check if a given string is the same when read forward and backward.**

**Example:**

**Input: "madam"**

**Output: True**

1. **Calculate the Sum of Alternating Elements in an Array:  25**

**Description: Add all elements at odd indices and subtract all elements at even indices.**

**Input: [10, 20, 30, 40]**

**Output: -20 (10 - 20 + 30 - 40)**

1. **https://leetcode.com/problems/last-stone-weight/description/**
2. **Count the number of words in a string   15**

**Description: Count the number of words in a string. A word is defined as a sequence of characters separated by spaces.**

**Example:**

**Input: "hello world"**

**Output: 2**

1. **Given an array of integers nums, return *the number of good pairs*.**

**A pair (i, j) is called *good* if nums[i] == nums[j] and i < j.**

**Example 1:**

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

**Output: 4**

**Explanation: There are 4 good pairs (0,3), (0,4), (3,4), (2,5) 0-indexed.**

**Example 2:**

**Input: nums = [1,1,1,1]**

**Output: 6**

**Explanation: Each pair in the array are *good*.**

**Example 3:**

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

**Output: 0**

1. **Given an array of integers nums sorted in non-decreasing order, find the starting and ending position of a given target value.**

**If target is not found in the array, return [-1, -1].**

**You must write an algorithm with O(log n) runtime complexity.**

**Example 1:**

**Input: nums = [5,7,7,8,8,10], target = 8**

**Output: [3,4]**

**Example 2:**

**Input: nums = [5,7,7,8,8,10], target = 6**

**Output: [-1,-1]**

**Example 3:**

**Input: nums = [], target = 0**

**Output: [-1,-1]**

1. **Median of an array:  15**

**Description: Find the median value of an array (middle element when sorted).**

**Example:**

**Input: [5, 2, 9, 1, 4, 10]**

**Output: 4 (after sorting: [1, 2, 4, 5, 9, 10])**

1. **Find the Length of the Longest Word in a Sentence:  25**

**Description: Determine the length of the longest word in a sentence.**

**Input: "I love programming"**

**Output: 11 (programming)**

1. **Given an array of strings, return all groups of strings that are anagrams. The strings in each group must be arranged in the order of their appearance in the original array. Refer to the sample case for clarification.**

**Examples:**

**Input: arr[] = ["act", "god", "cat", "dog", "tac"]**

**Output: [["act", "cat", "tac"], ["god", "dog"]]**

**Explanation: There are 2 groups of anagrams "god", "dog" make group 1. "act", "cat", "tac" make group 2.**

**Input: arr[] = ["no", "on", "is"]**

**Output: [["is"], ["no", "on"]]**

**Explanation: There are 2 groups of anagrams "is" makes group 1. "no", "on" make group 2.**

**Input: arr[] = ["listen", "silent", "enlist", "abc", "cab", "bac", "rat", "tar", "art"]**

**Output: [["abc", "cab", "bac"], ["listen", "silent", "enlist"], ["rat", "tar", "art"]]**

**Explanation:**

**Group 1: "abc", "bac", and "cab" are anagrams.**

**Group 2: "listen", "silent", and "enlist" are anagrams.**

**Group 3: "rat", "tar", and "art" are anagrams.**

1. **Check if a number is a perfect square**

**Description: Check whether a given number is a perfect square, i.e., its square root is an integer. 15 marks**

**Example:**

**Input: 25**

**Output: True (since √25 = 5)**

1. **Group All Zeroes to the End of an Array: du**

**Description: Move all zero elements in an array to the end while maintaining the order of non-zero elements.**

**Input: [0, 1, 2, 0, 3]**

**Output: [1, 2, 3, 0, 0]**

1. **an unsorted array arr[] of integers and an integer x, find the floor and ceiling of x in arr[]. 40 marks**

**Floor of x is the largest element which is smaller than or equal to x. Floor of x doesn’t exist if x is smaller than smallest element of arr[].  
Ceil of x is the smallest element which is greater than or equal to x. Ceil of x doesn’t exist if x is greater than greatest element of arr[].**

**Return an array of integers denoting the [floor, ceil]. Return -1 for floor or ceiling if the floor or ceiling is not present.**

**Examples:**

**Input: x = 7 , arr[] = [5, 6, 8, 9, 6, 5, 5, 6]**

**Output: 6, 8**

**Explanation: Floor of 7 is 6 and ceil of 7 is 8.**

**Input: x = 10 , arr[] = [5, 6, 8, 8, 6, 5, 5, 6]**

**Output: 8, -1**

**Explanation: Floor of 10 is 8 but ceil of 10 is not possible.**

1. **Binary search:**

**Description: A search algorithm that finds the position of a target value within a sorted array.**

**Example:**

**Input: [1, 2, 5, 9], target = 9**

**Output: Index 3**

1. **Description: Determine if a string contains every letter of the English alphabet at least once.**

**Input: "The quick brown fox jumps over the lazy dog"**

**Output: True**

1. **Given a string s and an integer k, return *the maximum number of vowel letters in any substring of* s *with length* k.**

**Vowel letters in English are 'a', 'e', 'i', 'o', and 'u'.**

**Example 1:**

**Input: s = "abciiidef", k = 3**

**Output: 3**

**Explanation: The substring "iii" contains 3 vowel letters.**

**Example 2:**

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

**Output: 2**

**Explanation: Any substring of length 2 contains 2 vowels.**

**Example 3:**

**Input: s = "leetcode", k = 3**

**Output: 2**

**Explanation: "lee", "eet" and "ode" contain 2 vowels.**

1. **Given a sorted array arr with possibly some duplicates, the task is to find the first and last occurrences of an element x in the given array.  
   Note: If the number x is not found in the array then return both the indices as -1.**

**Examples:**

**Input: arr[] = [1, 3, 5, 5, 5, 5, 67, 123, 125], x = 5**

**Output: [2, 5]**

**Explanation: First occurrence of 5 is at index 2 and last occurrence of 5 is at index 5**

**Input: arr[] = [1, 3, 5, 5, 5, 5, 7, 123, 125], x = 7**

**Output: [6, 6]**

**Explanation: First and last occurrence of 7 is at index 6**

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

**Output: [-1, -1]**

**Explanation: No occurrence of 4 in the array, so, output is [-1, -1]**

1. **First n numbers in reverse order:  15**

**Description: Print the first n numbers in reverse order.**

**Example:**

**Input: 5**

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

1. **Check if Two Numbers Are Co-Prime: 25**

**Description: Determine if two numbers have no common factors other than 1.**

**Input: 14, 15**

**Output: True**

1. **Given an unsorted array arr[] of integers and an integer x, find the floor and ceiling of x in arr[]. 40**

**Floor of x is the largest element which is smaller than or equal to x. Floor of x doesn’t exist if x is smaller than smallest element of arr[].  
Ceil of x is the smallest element which is greater than or equal to x. Ceil of x doesn’t exist if x is greater than greatest element of arr[].**

**Return an array of integers denoting the [floor, ceil]. Return -1 for floor or ceiling if the floor or ceiling is not present.**

**Examples:**

**Input: x = 7 , arr[] = [5, 6, 8, 9, 6, 5, 5, 6]**

**Output: 6, 8**

**Explanation: Floor of 7 is 6 and ceil of 7 is 8.**

**Input: x = 10 , arr[] = [5, 6, 8, 8, 6, 5, 5, 6]**

**Output: 8, -1**

**Explanation: Floor of 10 is 8 but ceil of 10 is not possible.**

**Constraints :**

**1 ≤ arr.size ≤ 105**

**1 ≤ arr[i], x ≤ 106**

[**https://leetcode.com/problems/points-that-intersect-with-cars/description/**](https://leetcode.com/problems/points-that-intersect-with-cars/description/)

1. **Sort an array:**

**Description: A sorting algorithm that repeatedly steps through the list, compares adjacent items, and swaps them if necessary.**

**Example:**

**Input: [5, 2, 9, 1]**

**Output: [1, 2, 5, 9]     {marks=15}**

1. **Reverse the Words in a Sentence:                     {marks=25}**

**Description: Reverse the order of words in a given sentence.**

**Input: "I love coding"**

**Output: "coding love I"**

1. **You are given an array prices where prices[i] is the price of a given stock on the ith day.**

**You want to maximize your profit by choosing a single day to buy one stock and choosing a different day in the future to sell that stock.**

**Return *the maximum profit you can achieve from this transaction*. If you cannot achieve any profit, return 0.**

**Example 1:**

**Input: prices = [7,1,5,3,6,4]**

**Output: 5**

**Explanation: Buy on day 2 (price = 1) and sell on day 5 (price = 6), profit = 6-1 = 5.**

**Note that buying on day 2 and selling on day 1 is not allowed because you must buy before you sell.**

**Example 2:**

**Input: prices = [7,6,4,3,1]**

**Output: 0**

**Explanation: In this case, no transactions are done and the max profit = 0.**

1. **Find the Missing Number in an Array:   15**

**Description: Identify the missing number from a sequence of numbers.**

**Input: [1, 2, 4, 5]**

**Output: 3**

1. **Group All Zeroes to the End of an Array:   25**

**Description: Move all zero elements in an array to the end while maintaining the order of non-zero elements.**

**Input: [0, 1, 2, 0, 3]**

**Output: [1, 2, 3, 0, 0]**

1. **You are given a 2D integer array logs where each logs[i] = [birthi, deathi] indicates the birth and death years of the ith person.**

**The population of some year x is the number of people alive during that year. The ith person is counted in year x's population if x is in the inclusive range [birthi, deathi - 1]. Note that the person is not counted in the year that they die.**

**Return *the earliest year with the maximum population*.**

**Example 1:**

**Input: logs = [[1993,1999],[2000,2010]]**

**Output: 1993**

**Explanation: The maximum population is 1, and 1993 is the earliest year with this population.**

**Example 2:**

**Input: logs = [[1950,1961],[1960,1971],[1970,1981]]**

**Output: 1960**

**Explanation:**

**The maximum population is 2, and it had happened in years 1960 and 1970.**

**The earlier year between them is 1960.**

1. **Check if a number is even or odd.   15**

**Example:**

**Input: 3**

**Output: odd**

1. **Remove spaces from a string   25**

**Description: Write a program to remove all spaces from a given string.**

**Example:**

**Input: "hello world"**

**Output: "helloworld"**

1. **Given an array of integers nums, return *the number of good pairs*.**

**A pair (i, j) is called *good* if nums[i] == nums[j] and i < j.**

**Example 1:**

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

**Output: 4**

**Explanation: There are 4 good pairs (0,3), (0,4), (3,4), (2,5) 0-indexed.**

**Example 2:**

**Input: nums = [1,1,1,1]**

**Output: 6**

**Explanation: Each pair in the array are *good*.**

**Example 3:**

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

**Output: 0**

1. **Description: Compute the volume of a cube given its side length.**

**Input: Side = 3**

**Output: Volume = 27**

1. **Calculate the Sum of the Diagonal Elements of a Matrix:   25**

**Description: Find the sum of all elements on the main diagonal of a square matrix.**

**Input: [[1, 2, 3], [4, 5, 6], [7, 8, 9]]**

**Output: 15 (1 + 5 + 9)**

1. **Print the ASCII Value of a Character:   15**

**Description: Find the ASCII value of a character.**

**Input: 'A'**

**Output: 65**

1. **Find the Number of Unique Elements in an Array:  25**

**Description: Count the distinct elements in an array.**

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

**40**

1. **Given a string s and an integer k, return *the maximum number of vowel letters in any substring of* s *with length* k.**

**Vowel letters in English are 'a', 'e', 'i', 'o', and 'u'.**

**Example 1:**

**Input: s = "abciiidef", k = 3**

**Output: 3**

**Explanation: The substring "iii" contains 3 vowel letters.**

**Example 2:**

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

**Output: 2**

**Explanation: Any substring of length 2 contains 2 vowels.**

1. **Harshad number check:**

**Description: A Harshad number is an integer that is divisible by the sum of its digits.**

**Example:**

**Input: 18**

**Output: True (1 + 8 = 9, and 18 is divisible by 9)**

1. **Sum of perfect squares in a range:  25**

**Description: Calculate the sum of all perfect squares between two numbers.**

**Example:**

**Input: 1 to 10**

**Output: 1 + 4 + 9 = 14**

1. **Given an array of integers nums sorted in non-decreasing order, find the starting and ending position of a given target value.**

**If target is not found in the array, return [-1, -1].**

**You must write an algorithm with O(log n) runtime complexity.**

**Example 1:**

**Input: nums = [5,7,7,8,8,10], target = 8**

**Output: [3,4]**

**Example 2:**

**Input: nums = [5,7,7,8,8,10], target = 6**

**Output: [-1,-1]**

**Example 3:**

**Input: nums = [], target = 0**

**Output: [-1,-1]**

1. **Perfect number check:**

**Description: A perfect number is a number whose sum of proper divisors (excluding the number itself) is equal to the number.**

**Example:**

**Input: 6**

**Output: True (1 + 2 + 3 = 6)**

1. **First n Armstrong numbers:   25**

**Description: Armstrong numbers are numbers that are equal to the sum of their own digits raised to the power of the number of digits.**

**Example:**

**Input: 3**

**Output: 153, 370, 371**

1. **Magic number check:**

**Description: A magic number is a number where the sum of digits repeated until a single digit equals .**

**Example:**

**Input: 19**

**Output: True (1 + 9 = 10, 1 + 0 = 1)**

1. **Find the prime factors of a number**

**Description: Find all prime factors of a given number.**

**Example:**

**Input: 18**

**Output: 2, 3**

1. **https://leetcode.com/problems/search-a-2d-matrix/description/**
2. **Multiplication table:**

**Description: Print the multiplication table of a given number.**

**Example:**

**Input: 5**

**Output: 5, 10, 15, 20, 25, 30, 35, 40, 45, 50**

1. **Check if a number is a prime factor of another number   25**

**Description: Check if one number is a prime factor of another number.**

**Example:**

**Input: 2, 20**

**Output: True (2 is a prime factor of 20)**

1. **Given an integer array nums, move all 0's to the end of it while maintaining the relative order of the non-zero elements.**

**Note that you must do this in-place without making a copy of the array.**

**Example 1:**

**Input: nums = [0,1,0,3,12]**

**Output: [1,3,12,0,0]**

**Example 2:**

**Input: nums = [0]**

**Output: [0]**

1. **Calculate the Perimeter of a Rectangle:**

**Description: Find the perimeter given length and breadth.**

**Input: Length = 5, Breadth = 3**

**Output: Perimeter = 16**

1. **Find the Mode of an Array:**

**Description: Identify the number that appears most frequently in an array.**

**Input: [1, 2, 2, 3, 4]**

**Output: 2**

1. **Given an array arr[ ] of integers, the task is to find the next greater element for each element of the array in order of their appearance in the array. Next greater element of an element in the array is the nearest element on the right which is greater than the current element.  
   If there does not exist next greater of current element, then next greater element for current element is -1. For example, next greater of the last element is always -1.**

**Examples**

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

**Output: [3, 4, 4, -1]**

**Explanation: The next larger element to 1 is 3, 3 is 4, 2 is 4 and for 4, since it doesn't exist, it is -**

**Input: arr[] = [6, 8, 0, 1, 3]**

**Output: [8, -1, 1, 3, -1]**

**Explanation: The next larger element to 6 is 8, for 8 there is no larger elements hence it is -1, for 0 it is 1 , for 1 it is 3 and then for 3 there is no larger element on right and hence -1.**

**Input: arr[] = [10, 20, 30, 50]**

**Output: [20, 30, 50, -1]**

**Explanation: For a sorted array, the next element is next greater element also exxept for the last element.**

**Input: arr[] = [50, 40, 30, 10]**

**Output: [-1, -1, -1, -1]**

**Explanation: There is no greater element for any of the elements in the array, so all are -1.**

1. **HCF of two numbers**:

**Description**: The highest common factor (HCF) of 2 numbers.

**Example**:

Input: 12, 18

Output: 6

1. **Second smallest element   25**

**Description: Find the second smallest element in an array.**

**Example:**

**Input: [5, 2, 9, 1]**

**Output: Second smallest = 2**

**40**

1. **You are given an array of integers stones where stones[i] is the weight of the ith stone.**

**We are playing a game with the stones. On each turn, we choose the heaviest two stones and smash them together. Suppose the heaviest two stones have weights x and y with x <= y. The result of this smash is:**

* **If x == y, both stones are destroyed, and**
* **If x != y, the stone of weight x is destroyed, and the stone of weight y has new weight y - x.**

**At the end of the game, there is at most one stone left.**

**Return *the weight of the last remaining stone*. If there are no stones left, return 0.**

**Example 1:**

**Input: stones = [2,7,4,1,8,1]**

**Output: 1**

**Explanation:**

**We combine 7 and 8 to get 1 so the array converts to [2,4,1,1,1] then,**

**we combine 2 and 4 to get 2 so the array converts to [2,1,1,1] then,**

**we combine 2 and 1 to get 1 so the array converts to [1,1,1] then,**

**we combine 1 and 1 to get 0 so the array converts to [1] then that's the value of the last stone.**

**Example 2:**

**Input: stones = [1]**

**Output: 1**

1. **Print the pattern of stars (pyramid)**

**Description: Print a pyramid-shaped pattern using stars (\*).**

**Example:**

**Input: 5**

**Output:**

**\*     
\*\*\*     
\*\*\*\*\*     
\*\*\*\*\*\*\*     
\*\*\*\*\*\*\*\*\***

1. **Print the Diagonal Elements of a Matrix:       25**

**Description: Extract the diagonal elements of a square matrix.**

**Input: [[1, 2, 3], [4, 5, 6], [7, 8, 9]]**

**Output: [1, 5, 9]**

1. **Given an element x and a sorted array, arr[], find the indices of first and last occurrences of the x's in the array.**

**Note: If the number x is not found in the array, return an array containing -1 only.**

**Examples:**

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

**Output: [1, 2]**

**Explanation: The first occurrence of X = 3 is at index = 1 and the last at index = 2.**

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

**Output: [-1]**

**Explanation: 5 is not present, so the answer is -1.**

1. **Print the pattern of stars (reverse pyramid)**

**Description: Print an inverted pyramid pattern using stars (\*).**

**Example:**

**Input: 5**

**Output:**

**\*\*\*\*\*\*\*\*\*     
\*\*\*\*\*\*\*     
\*\*\*\*\*     
\*\*\*     
\***

1. **Find the Length of the Longest Word in a Sentence:**

**Description: Determine the length of the longest word in a sentence.**

**Input: "I love programming"**

**Output: 11 (programming)**

1. **Given an unsorted array arr[] of integers and an integer x, find the floor and ceiling of x in arr[].**

**Floor of x is the largest element which is smaller than or equal to x. Floor of x doesn’t exist if x is smaller than smallest element of arr[].  
Ceil of x is the smallest element which is greater than or equal to x. Ceil of x doesn’t exist if x is greater than greatest element of arr[].**

**Return an array of integers denoting the [floor, ceil]. Return -1 for floor or ceiling if the floor or ceiling is not present.**

**Examples:**

**Input: x = 7 , arr[] = [5, 6, 8, 9, 6, 5, 5, 6]**

**Output: 6, 8**

**Explanation: Floor of 7 is 6 and ceil of 7 is 8.**

**Input: x = 10 , arr[] = [5, 6, 8, 8, 6, 5, 5, 6]**

**Output: 8, -1**

**Explanation: Floor of 10 is 8 but ceil of 10 is not possible**

1. **Sum of even digits in a number   25**

**Description: Find the sum of all even digits in a given number.**

**Example:**

**Input: 123456**

**Output: 12 (2 + 4 + 6)**

1. [**https://leetcode.com/problems/merge-two-sorted-lists/description/**](https://leetcode.com/problems/merge-two-sorted-lists/description/)
2. **Perfect number check:   15**

**Description: A perfect number is a number whose sum of proper divisors (excluding the number itself) is equal to the number.**

**Example:**

**Input: 6**

**Output: True (1 + 2 + 3 = 6)**

1. **Convert Celsius to Fahrenheit:   15**

**Description: Convert temperature from Celsius to Fahrenheit.**

**Input: Celsius = 25**

**Output: Fahrenheit = 77 (F = (C × 9/5) + 32)**

1. **Group All Zeroes to the End of an Array:   25**

**Description: Move all zero elements in an array to the end while maintaining the order of non-zero elements.**

**Input: [0, 1, 2, 0, 3]**

**Output: [1, 2, 3, 0, 0]**

1. **Given an array of *n* names arr of candidates in an election, where each name is a string of lowercase characters. A candidate name in the array represents a vote casted to the candidate. Print the name of the candidate that received the maximum count of votes. If there is a draw between two candidates, then print *lexicographically* smaller name.**

**Examples :**

**Input: n = 13**

**arr[] = {john,johnny,jackie,johnny,john,jackie,jamie,jamie,john,johnny,jamie,johnny,john}**

**Output: john 4**

**Explanation: john has 4 votes casted for him, but so does johnny. john is lexicographically smaller, so we print john and the votes he received.**

**Input: n = 3**

**arr[] = {andy,blake,clark}**

**Output: Andy 1**

**Explanation: All the candidates get 1 votes each. We print andy as it is lexicographically smaller.**

1. **Find the Minimum of Three Numbers:**

**Description: Determine the smallest of three numbers.**

**Input: 5, 2, 8**

**Output: 2**

1. **Sum of all prime numbers in a range:**

**Description: The sum of all prime numbers between two numbers is calculated by first identifying all primes in that range and then adding them.**

**Example:**

**Input: 10 to 20**

**Output: 60 (11 + 13 + 17 + 19)**

1. **Given a string s and an integer k, return *the maximum number of vowel letters in any substring of* s *with length* k.**

**Vowel letters in English are 'a', 'e', 'i', 'o', and 'u'.**

**Example 1:**

**Input: s = "abciiidef", k = 3**

**Output: 3**

**Explanation: The substring "iii" contains 3 vowel letters.**

**Example 2:**

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

**Output: 2**

**Explanation: Any substring of length 2 contains 2 vowels.**

**Example 3:**

**Input: s = "leetcode", k = 3**

**Output: 2**

**Explanation: "lee", "eet" and "ode" contain 2 vowels.**

1. **Find the smallest digit in a number**

**Description: Write a program to find the smallest digit in a given number.**

**Example:**

**Input: 98321**

**Output: 1**

1. **Find the Sum of Digits in a Number Until a Single Digit is Obtained:**

**Description: Keep summing the digits of a number until a single digit is reached.**

**Input: 987**

**Output: 6 (9 + 8 + 7 = 24; 2 + 4 = 6)**

1. **Sum of digits:**

**Description: The sum of digits of a number is calculated by adding each individual digit.**

**Example:**

**Input: 1234**

**Output: 10 (1 + 2 + 3 + 4 = 10)**

1. **Sum of odd digits in a number**

**Description: Find the sum of all odd digits in a given number.**

**Example:**

**Input: 123456**

**Output: 9 (1 + 3 + 5)**

1. **Check if two strings are anagrams**

**Description: Check if two strings contain the same characters in any order.**

**Example:**

**Input: "listen", "silent"**

**Output: True**

1. **Simple and compound interest:**

**Description: Calculate simple interest (SI) and compound interest (CI) based on principal, rate, and time.**

**Example:**

**Input: P = 1000, R = 5%, T = 2**

**Output: SI = 100, CI = 102.5**

1. **First n Armstrong numbers:**

**Description: Armstrong numbers are numbers that are equal to the sum of their own digits raised to the power of the number of digits.**

**Example:**

**Input: 3**

**Output: 153, 370, 371**

1. **Check if two strings are anagrams**

**Description: Check if two strings contain the same characters in any order.**

**Example:**

**Input: "listen", "silent"**

**Output: True**

1. **Linear search:**

**Description: A search algorithm that checks each element in the list sequentially to find the target element.**

**Example:**

**Input: [5, 2, 9, 1], target = 9**

**Output: Index 2**

1. **Second smallest element**

**Description: Find the second smallest element in an array.**

**Example:**

**Input: [5, 2, 9, 1]**

**Output: Second smallest = 2**

1. **Given an integer array nums, return *an array* answer *such that* answer[i] *is equal to the product of all the elements of* nums *except* nums[i].**

**The product of any prefix or suffix of nums is guaranteed to fit in a 32-bit integer.**

**You must write an algorithm that runs in O(n) time and without using the division operation.**

**Example 1:**

**Input: nums = [1,2,3,4]**

**Output: [24,12,8,6]**

**Example 2:**

**Input: nums = [-1,1,0,-3,3]**

**Output: [0,0,9,0,0]**

1. **Frequency of digits:**

**Description: Count the frequency of each digit in a number.**

**Example:**

**Input: 112233**

**Output: 1: 2, 2: 2, 3: 2**

1. **Sum of all prime numbers in a range:**

**Description: The sum of all prime numbers between two numbers is calculated by first identifying all primes in that range and then adding them.**

**Example:**

**Input: 10 to 20**

**Output: 60 (11 + 13 + 17 + 19)**

1. **Given an array of *n* names arr of candidates in an election, where each name is a string of lowercase characters. A candidate name in the array represents a vote casted to the candidate. Print the name of the candidate that received the maximum count of votes. If there is a draw between two candidates, then print *lexicographically* smaller name.**

**Examples :**

**Input: n = 13**

**arr[] = {john,johnny,jackie,johnny,john,jackie,jamie,jamie,john,johnny,jamie,johnny,john}**

**Output: john 4**

**Explanation: john has 4 votes casted for him, but so does johnny. john is lexicographically smaller, so we print john and the votes he received.**

**Input: n = 3**

**arr[] = {andy,blake,clark}**

**Output: Andy 1**

**Explanation: All the candidates get 1 votes each. We print andy as it is lexicographically smaller.**

1. **Sort an array:**

**Description: A sorting algorithm that repeatedly steps through the list, compares adjacent items, and swaps them if necessary.**

**Example:**

**Input: [5, 2, 9, 1]**

**Output: [1, 2, 5, 9]**

1. **First n Armstrong numbers:**

**Description: Armstrong numbers are numbers that are equal to the sum of their own digits raised to the power of the number of digits.**

**Example:**

**Input: 3**

**Output: 153, 370, 371**

1. **Given a string s and an integer k, return *the maximum number of vowel letters in any substring of* s *with length* k.**

**Vowel letters in English are 'a', 'e', 'i', 'o', and 'u'.**

**Example 1:**

**Input: s = "abciiidef", k = 3**

**Output: 3**

**Explanation: The substring "iii" contains 3 vowel letters.**

**Example 2:**

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

**Output: 2**

**Explanation: Any substring of length 2 contains 2 vowels.**

**Example 3:**

**Input: s = "leetcode", k = 3**

**Output: 2**

**Explanation: "lee", "eet" and "ode" contain 2 vowels.**

1. **Find the Missing Number in an Array**

**Description: Identify the missing number from a sequence of numbers.**

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

1. **Remove duplicates from an array**

**Description: Remove all duplicate elements from an array.**

**Example:**

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

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

1. **Given a binary array nums, return *the maximum number of consecutive* 1*'s in the array*.**

**Example 1:**

**Input: nums = [1,1,0,1,1,1]**

**Output: 3**

**Explanation: The first two digits or the last three digits are consecutive 1s. The maximum number of consecutive 1s is 3.**

**Example 2:**

**Input: nums = [1,0,1,1,0,1]**

**Output: 2**

**Constraints:**

* **1 <= nums.length <= 105**
* **nums[i] is either 0 or 1.**

1. **Factors of a number:**

**Description: A factor of a number is any number that divides the original number without leaving a remainder. Print all the factors in comma seperated manner.**

**Example:**

* **Input: 12**
* **Output: 1, 2, 3, 4, 6, 12**

1. **Find the Length of the Longest Word in a Sentence:**

**Description: Determine the length of the longest word in a sentence.**

* **Input: "I love programming"**
* **Output: 11 (programming)**

1. **Check if a Matrix is Identity Matrix:**

**Description: Verify if a matrix is an identity matrix.**

* **Input: [[1, 0, 0], [0, 1, 0], [0, 0, 1]]**
* **Output: True**

1. **Convert a string to uppercase without using internal libraries**

**Description: Write a program to convert all letters in a string to uppercase.**

**Example:**

**Input: "hello"**

**Output: "HELLO"**

1. **Find the common divisors of two numbers**

**Description: Find the divisors common to two numbers.**

**Example:**

**Input: 12, 18**

**Output: 1, 2, 3, 6**

1. **Check if a string contains only digits**

**Description: Check if a given string consists solely of digits.**

**Example:**

**Input: "12345"**

**Output: True5**

1. **Check if a String is a Pangram:**

**Description: Determine if a string contains every letter of the English alphabet at least once.**

* **Input: "The quick brown fox jumps over the lazy dog"**
* **Output: True**

1. **Check if two strings are anagrams**

**Description: Check if two strings contain the same characters in any order.**

* **Example:**

**Input: "listen", "silent"**

**Output: True**

1. **Find the number of digits in a number**

**Description: Find how many digits are in a given number.**

* **Example:**

**Input: 12345**

**Output: 5**

**Output: True (45^2 = 2025 and 20 + 25 = 45)**

1. **Reverse the Words in a Sentence:**

**Description: Reverse the order of words in a given sentence.**

* **Input: "I love coding"**
* **Output: "coding love I"**

1. **Rotate a matrix by 90 degrees:**

**Description: Rotate a matrix 90 degrees clockwise.**

* **Example:**
* **Input: [[1, 2], [3, 4]]**
* **Output: [[3, 1], [4, 2]]**

1. **Calculate the Volume of a Cube:**

**Description: Compute the volume of a cube given its side length.**

* **Input: Side = 3**
* **Output: Volume = 27**

1. **Prime palindrome check:**

**Description: A prime palindrome is a number that is both prime and a palindrome.**

* **Example:**
* **Input: 131**

**Output: True (131 is prime and palindrome)**

1. **You are given a 2D integer array logs where each logs[i] = [birthi, deathi] indicates the birth and death years of the ith person.**

**The population of some year x is the number of people alive during that year. The ith person is counted in year x's population if x is in the inclusive range [birthi, deathi - 1]. Note that the person is not counted in the year that they die.**

**Return *the earliest year with the maximum population*.**

**Example 1:**

**Input: logs = [[1993,1999],[2000,2010]]**

**Output: 1993**

**Explanation: The maximum population is 1, and 1993 is the earliest year with this population.**

**Example 2:**

**Input: logs = [[1950,1961],[1960,1971],[1970,1981]]**

**Output: 1960**

**Explanation:**

**The maximum population is 2, and it had happened in years 1960 and 1970.**

**The earlier year between them is 1960.**

1. **Sum of odd numbers in a range:**

**Description: Calculate the sum of all odd numbers between two given numbers.**

**Example:**

* **Input: 10 to 20**
* **Output: 75 (11 + 13 + 15 + 17 + 19)**

1. **Group All Zeroes to the End of an Array:**

**Description: Move all zero elements in an array to the end while maintaining the order of non-zero elements.**

* **Input: [0, 1, 2, 0, 3]**
* **Output: [1, 2, 3, 0, 0]**

1. **Given an array arr[ ] of integers, the task is to find the next greater element   for each element of the array in order of their appearance in the array. Next greater element of an element in the array is the nearest element on the right which is greater than the current element.  
   If there does not exist next greater of current element, then next greater element for current element is -1. For example, next greater of the last element is always -1.**

**Examples**

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

**Output: [3, 4, 4, -1]**

**Explanation: The next larger element to 1 is 3, 3 is 4, 2 is 4 and for 4, since it doesn't exist, it is -1.**

1. Swap two numbers using XOR: 15

Description: Swap two numbers using the XOR bitwise operation without using a temporary variable.

Example:

Input: a = 5, b = 3

Output: a = 3, b = 5

1. Prime palindrome check: 25

Description: A prime palindrome is a number that is both prime and a palindrome.

Example:

Input: 131

Output: True (131 is prime and palindrome)

1. Given an array of strings, return all groups of strings that are anagrams. The strings in each group must be arranged in the order of their appearance in the original array. Refer to the sample case for clarification.

Examples:

Input: arr[] = ["act", "god", "cat", "dog", "tac"]

Output: [["act", "cat", "tac"], ["god", "dog"]]

Explanation: There are 2 groups of anagrams "god", "dog" make group 1. "act", "cat", "tac" make group 2.