1.Remove duplicate:

Given an integer array nums sorted in **non-decreasing order**, remove the duplicates **in-place** such that each unique element appears only **once**. The **relative order** of the elements should be kept the **same**. Then return *the number of unique elements in* nums.

Consider the number of unique elements of nums to be k, to get accepted, you need to do the following things:

- Change the array nums such that the first k elements of nums contain the unique elements in the order they were present in nums initially. The remaining elements of nums are not important as well as the size of nums.
- Return k.

2.Remove Element

Given an integer array nums and an integer val, remove all occurrences of val in nums **in-place**. The order of the elements may be changed. Then return *the number of elements in* nums *which are not equal to* val.

Consider the number of elements in nums which are not equal to val be k, to get accepted, you need to do the following things:

- Change the array nums such that the first k elements of nums contain the elements which are not equal to val. The remaining elements of nums are not important as well as the size of nums.
- Return k.

3.Marge sorted array

You are given two integer arrays nums1 and nums2, sorted in **non-decreasing order**, and two integers m and n, representing the number of elements in nums1 and nums2 respectively.

Merge nums1 and nums2 into a single array sorted in **non-decreasing order**.

The final sorted array should not be returned by the function, but instead be *stored inside the array* nums1. To accommodate this, nums1 has a length of m + n, where the first m elements denote the elements that should be merged, and the last n elements are set to 0 and should be ignored. nums2 has a length of n.

4.Best Time to Buy and Sell Stock

You are given an array prices where prices[i] is the price of a given stock on the i^{th} 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.

5.Majority Element

Given an array nums of size n, return *the majority element*.

The majority element is the element that appears more than $\lfloor n / 2 \rfloor$ times. You may assume that the majority element always exists in the array.

6.Missing Number

Given an array nums containing n distinct numbers in the range [0, n], return *the only number in the range that is missing from the array.*

Example 1:

Input: nums = [3,0,1]

Output: 2

Explanation: n = 3 since there are 3 numbers, so all numbers are in the range [0,3]. 2 is the missing number in the range since it does not appear in nums

7.Move Zero

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]

8.Binary Search

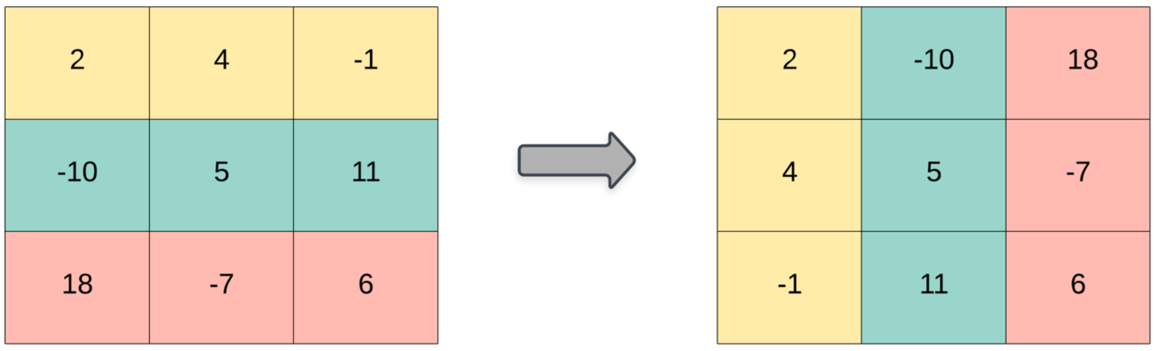
Given an array of integers nums which is sorted in ascending order, and an integer target, write a function to search target in nums. If target exists, then return its index. Otherwise, return -1.

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

9.Transpose Matrix

Given a 2D integer array matrix, return *the **transpose** of* matrix.

The **transpose** of a matrix is the matrix flipped over its main diagonal, switching the matrix's row and column indices.



10.Find All Roots Of A Quadratic Equation

Write a C program to find all roots of a quadratic equation using if else. How to find all roots of a quadratic equation using if else in C programming. Logic to find roots of quadratic equation in C programming.

Example Input

Input a: 8

Input b: -4

Input c: -2

11.Check Whether A Character Is Alphabet, Digit Or Special Character

Write a C program to input a character from user and check whether given character is alphabet, digit or special character using if else. How to check if a character is alphabet, digits or any other special character using if else in C programming. Logic to check alphabet, digit or special character in C programming.

Example:
Input

Input any character: 3

Output

3 is digit

12.C program to calculate electricity bill

Write a C program to input electricity unit charge and calculate the total electricity bill according to the given condition:
For first 50 units Rs. 0.50/unit
For next 100 units Rs. 0.75/unit
For next 100 units Rs. 1.20/unit
For unit above 250 Rs. 1.50/unit
An additional surcharge of 20% is added to the bill.

13.C program to create calculator using switch case and functions

Write a C program to create menu driven calculator that performs basic arithmetic operations (add, subtract, multiply and divide) using switch case and functions. The calculator should input two numbers and an operator from user. It should perform operation according to the operator entered and must take input in given format.

<number 1> <operator> <number 2>

Example

Input

5.2 – 3

14.Divisibility by 5 and 11

Statement: Given an integer N, determine whether it is divisible by both 5 and 11.

Input: A single integer N.

Output: Print "Yes" if divisible by both, otherwise "No".

15.Grade Calculator

Statement: Given a student's score S (out of 100), determine the grade based on the following ranges:

A: 90-100

B: 80-89

C: 70-79

D: 60-69

F: Below 60

Input: A single integer S.

Output: Print the grade.

16.Array in Reverse Order

Statement: Print the elements of an array of N integers in reverse order.

Input: First line contains an integer N. Second line contains N space-separated integers.

Output: Print the array in reverse order.

17.Character Case Check

Statement: Given a character C, determine whether it is uppercase or lowercase.

Input: A single character C.

Output: Print "Uppercase" or "Lowercase".

18.Element Search in Array

Statement: Search for an element X in an array of N integers. Return its position or -1 if not found.

Input: First line contains an integer N. Second line contains N space-separated integers. Third line contains X.

Output: Print the index of the element or -1.

19.Largest Element in Array

Statement: Find the largest element in a given array of N integers.

Input: First line contains an integer N. Second line contains N space-separated integers.

Output: Print the largest element.

20.Leap year

Statement: Given a year Y, determine whether it is a leap year.

Input: A single integer Y.

Output: Print "Leap Year" or "Not a Leap Year".

21.Maximum of Two Numbers

Statement: Given two integers A and B, find the larger number.

Input: Two space-separated integers A and B.

Output: Print the larger of the two.

22.Minimum of Three Numbers

Statement: Given three integers A, B, and C, find the smallest number.

Input: Three space-separated integers.

Output: Print the smallest number.

23.Even Odd

Statement: Given an integer N, determine if it is odd or even.

Input: A single integer N.

Output: Print "Odd" or "Even" based on the number.

24.Sign of a number

Statement: Given an integer N, determine whether it is positive, negative, or zero.

Input: A single integer N.

Output: Print "Positive", "Negative", or "Zero".

25.Sum of Array Elements

Statement: Given an array of N integers, find the sum of all elements.

Input: First line contains an integer N. Second line contains N space-separated integers.

Output: Print the sum of the array.

26.Triangle Validity

Statement: Given three sides of a triangle a, b, c, check if they can form a valid triangle.

Input: Three integers a, b, and c.

Output: Print "Valid" if they can form a triangle, otherwise "Invalid".

27.Vowel or Consonant?

Statement: Given a character C, check if it is a vowel or consonant.

Input: A single character C.

Output: Print "Vowel" or "Consonant".

28.Sum of Even Numbers:

Write a C program that reads an array of integers, then calculates and outputs the sum of all the even numbers in the array.

29.Check if a Year is a Century Year:

Write a C program that takes a year as input and checks if it is a century year (a year that is divisible by 100 but not by 400). Output "Century Year" or "Not a Century Year.

30.Check if a Number is a Power of Two:

Write a C program that takes an integer as input and checks if it is a power of two. Output "Power of Two" or "Not a Power of Two."

31.Categorize Age Group:

Write a C program that takes a person's age as input and categorizes them into one of the following groups:

- 0-12: Child
- 13-19: Teenager
- 20-64: Adult
- 65+: Senior

Output the corresponding age group.

32.Count Positive and Negative Numbers:

Write a C program that takes an array of integers as input and counts how many numbers are positive and how many are negative. Output both counts.

33.Find the Second Largest Number:

Write a C program that reads an array of integers and finds the second largest number in the array. Output the second largest number.

34.Check if Array is Sorted:

Write a C program that checks whether a given array of integers is sorted in ascending order. Output "Yes" if the array is sorted, otherwise output "No."

35.Count Odd and Even Numbers:

Write a C program that reads an array of integers and counts how many numbers are odd and how many are even. Output both counts.

36.Find All Prime Numbers in an Array:

Write a C program that takes an array of integers and identifies which elements are prime numbers. Output the prime numbers in the array.

37.Determine the Day of the Week:

Write a C program that takes a number (1-7) as input and outputs the corresponding day of the week (e.g., 1 for Monday, 2 for Tuesday, etc.). If the number is outside the range, output "Invalid Day.”

38.Check for Perfect Number:

Write a C program that takes an integer as input and checks if it is a perfect number (a number that is equal to the sum of its proper divisors, excluding itself). Output "Perfect Number" or "Not a Perfect Number."

39.Check for Armstrong Number:

Write a C program that takes a three-digit number as input and checks if it is an Armstrong number (a number equal to the sum of the cubes of its digits). Output "Armstrong Number" or "Not an Armstrong Number."

40.Find the Type of Quadrilateral:

Write a C program that takes the lengths of four sides and checks if they form a rectangle, square, or another quadrilateral. Output the type of quadrilateral.