1. Find the smallest number in an array
2. Find the largest number in an array
3. Second Smallest and Second Largest element in an array
4. Reverse a given array
5. Count frequency of each element in an array
6. Rearrange array in increasing-decreasing order
7. Calculate sum of the elements of the array
8. Rotate array by K elements – Block Swap Algorithm
9. Average of all elements in an array
10. Find the median of the given array
11. Remove duplicates from a sorted array
12. Remove duplicates from unsorted array
13. Adding Element in an array
14. Find all repeating elements in an array
15. Find all non-repeating elements in an array
16. Find all symmetric pairs in array
17. Maximum product subarray in an array (ex- [7, 2, 3, 5] output: 15 ->(3 \* 5) continuous)
18. Replace each element of the array by its rank in the array
19. Sorting elements of an array by frequency
20. Rotation of elements of array- left and right
21. Finding equilibrium index of an array
22. Finding Circular rotation of an array by K positions
23. Sort an array according to the order defined by another array
24. Search an element in an array
25. Check if Array is a subset of another array or not

Problems on Numbers

Number-based problems are fundamental in competitive programming. These will help you improve your mathematical reasoning and logic:

1. Prime numbers in a given range
2. Check if a number is palindrome or not
3. Find all **Palindrome numbers** in a given range
4. Check if a number is **prime** or not
5. Check if a number is **armstrong number** of not
6. Check if a number is **perfect number** :

**# A perfect number is a positive intger that is equal to the sum of its proper positive divisors(excluding number itself)**

1. Check if a number is a **strong number** or not :

A Strong number is a number whose sum of the factorials of its digits is equal to the number itself.

Example:

145 is a Strong number because:

1. 1! + 4! + 5! = 1 + 24 + 120 = 145
2. Check if a Number is **Automorphic**

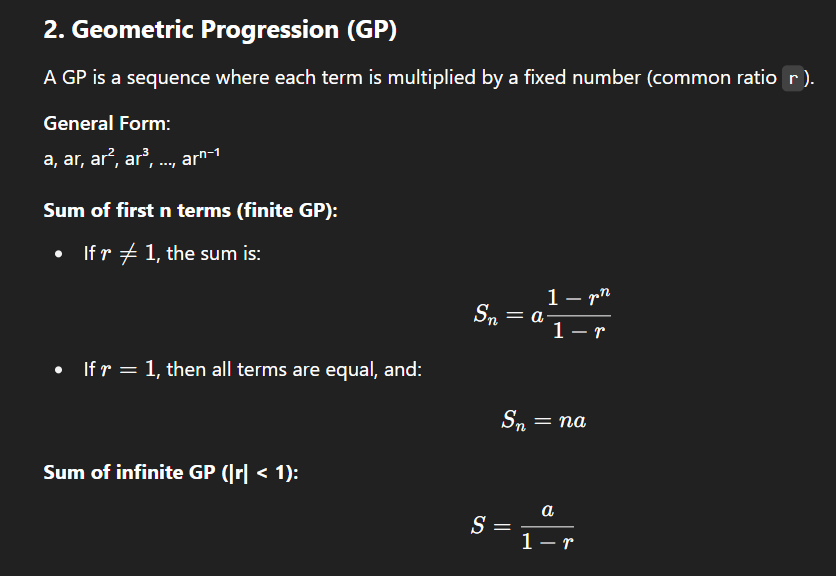
An Automorphic number (also known as a circular number) is a number whose square ends with the number itself.

Examples:

5² = 25 → ends with 5 → Automorphic

76² = 5776 → ends with 76 → Automorphic

6² = 36 → does not end with 6 → Not Automorphic

1. Check if a number is **Harshad number/ Niven number**
2. Check if the number is **abundant number** or not
3. Even or Odd
4. Check weather a given number is positive or negative
5. Sum of first N natural numbers ( n \* (n + 1)) /2
6. Find Sum of AP Series Sₙ = n/2 [2a + (n-1)d]
7. Program to find sum of GP Series :
8. Greatest of two numbers
9. Greatest of three numbers
10. Leap Year or not
11. Reverse digits of a number
12. Maximum and Minimum digit in a number
13. Print Fibonacci upto Nth Term
14. Factorial of a number
15. Power of a number
16. Factors of a given number
17. Print all prime factors of the given number
18. GCD of two numbers
19. LCM of two numbers
20. Sum of digits of a number
21. Sum of numbers in the given range
22. Permutations in which N people can occupy R seats in a classroom = N! /(N-R)!
23. Program to add two fractions
24. Replace all 0s with 1s in a given integer
25. is a number a sum of two prime numbers
26. Calculate the area of circle
27. Program to find roots of a Quadratic Equation Problems on Number System

These problems will test your understanding of number systems, a crucial concept for many coding interviews and exams:

1. Convert Binary to Decimal
2. Convert binary to octal
3. Decimal to Binary conversion
4. Convert decimal to octal
5. Convert octal to binary
6. Convert octal to decimal
7. Convert digits/numbers to wordsProblems on Sorting

Sorting algorithms are essential, and understanding the different techniques will give you an edge in problem-solving:

1. Bubble Sort Algorithm
2. Selection Sort Algorithm
3. Insertion Sort Algorithm
4. Quick Sort Algorithm
5. Merge sort algorithm

**Problems on Strings**  
String manipulation is an important skill, and there are many practical problems that require efficient string handling:

1. Check if a given string is palindrome or not
2. Count number of vowels, consonants, spaces in String
3. Find the ASCII value of a character
4. Remove all vowels from the string
5. Remove spaces from a string
6. Remove characters from a string except alphabets
7. Reverse a String
8. Remove brackets from an algebraic expression
9. Sum of the numbers in a String
10. Capitalize first and last character of each word
11. Calculate frequency of characters in a string
12. Find Non-repeating characters of a String
13. Check if two strings are anagram of each other
14. Count common sub-sequence in two strings
15. Check if two strings match where one string contains wildcard characters
16. Return maximum occurring character in the input string
17. Remove all duplicates from the input string.
18. Print all the duplicates in the input string.
19. Remove characters from first string present in the second string
20. Change every letter with the next lexicographic alphabet in the given string
21. Write a program to find the largest word in a given string.
22. Write a program to sort characters in a string
23. Count number of words in a given string
24. Write a program to find a word in a given string which has the highest number of repeated letters
25. Change case of each character in a string
26. Concatenate one string to another
27. Write a program to find a substring within a string. If found display its starting position
28. Reverse words in a string

2. Data Structures:

* **Arrays:** Basic array operations, searching, sorting, and manipulation
* **Linked Lists:** Understanding linked list nodes, traversal, and manipulation
* **Stacks and Queues:** FIFO and LIFO data structures, their operations, and applications
* **Trees:** Binary trees, traversal techniques, and related algorithms
* **Graphs:** Understanding graph representations, traversal, and shortest path algorithms

3. Algorithms:

* **Sorting algorithms:** Bubble sort, insertion sort, selection sort, merge sort, quicksort, etc.
* **Searching algorithms:** Linear search, binary search
* **Dynamic programming:** Understanding the concept and applying it to optimize solutions