

---

## Programming Exercises

---

### Control Structures & Loops

#### 1. FizzBuzz Variation

Print numbers from 1 to n, but replace multiples of 3 with "Fizz", 5 with "Buzz", and both with "FizzBuzz".

#### 2. Prime Number Checker

Check if a given number is prime. If not, list its factors.

#### 3. Sum of Digits Until Single Digit

Keep summing the digits of a number until a single-digit result is obtained.

#### 4. Reverse a Number and Check Palindrome

Reverse an integer and check if it's a palindrome.

#### 5. Find GCD and LCM of Two Numbers

Use loops and conditionals to compute both GCD and LCM.

---

### Arrays & Collections

#### 6. Second Largest Element in Array

Find the second largest number in an array without sorting.

#### 7. Remove Duplicates from Array

Remove duplicate elements from an integer array.

#### 8. Rotate Array by k Positions

Rotate an array to the right by k positions.

#### 9. Find Missing Number in Sequence

Given an array of n-1 numbers from 1 to n, find the missing number.

#### 10. Check if Array is Sorted

Determine if an array is sorted in ascending, descending, or unsorted.

---

### Strings & Character Logic

#### 11. Anagram Checker

Check if two strings are anagrams of each other.

## 12. Count Vowels and Consonants

Count the number of vowels and consonants in a string.

## 13. Longest Word in a Sentence

Find the longest word in a given sentence.

## 14. Toggle Case of Each Character

Convert uppercase letters to lowercase and vice versa.

## 15. Check Pangram

Check if a sentence contains every letter of the alphabet at least once.

---

## Number Logic & Math

### 16. Armstrong Number Checker

Check if a number is an Armstrong number (e.g.,  $153 = 1^3 + 5^3 + 3^3$ ).

### 17. Generate Fibonacci Series

Generate the first n numbers in the Fibonacci sequence.

### 18. Find All Perfect Numbers up to N

A perfect number is equal to the sum of its proper divisors.

### 19. Convert Decimal to Binary and Vice Versa

Implement manual conversion without using built-in methods.

### 20. Sum of Prime Numbers in a Range

Sum all prime numbers between two given numbers.

---

## Logical Thinking & Patterns

### 21. Pascal's Triangle Generator

Print the first n rows of Pascal's Triangle.

### 22. Find Leader Elements in Array

An element is a leader if it is greater than all elements to its right.

### 23. Matrix Transpose

Transpose a given 2D matrix.

### 24. Count Frequency of Each Word in a Sentence

Use loops and maps to count word frequency.

### 25. Check Sudoku Row Validity

Given a row from a Sudoku puzzle, check if it contains all digits from 1 to 9 without repetition.

## 26. Spiral Matrix Generator

### Problem:

Given an integer  $n$ , generate an  $n \times n$  matrix filled with numbers from 1 to  $n^2$  in a spiral order (clockwise).

## 27. Password Strength Validator

### Problem:

Write a program that takes a string input and checks if it qualifies as a strong password. A strong password must:

- Be at least 8 characters long
- Contain at least one uppercase letter
- Contain at least one lowercase letter
- Contain at least one digit
- Contain at least one special character (e.g., !@#\$%^&\*())

## 28. Number Pyramid Pattern

### Problem:

Generate a pyramid pattern of numbers for a given input  $n$ . For example, if  $n = 4$ , output:

```
1      1
2     2 3
3    4 5 6
4   7 8 9 10
```

## 29. Digit Frequency Counter

### Problem:

Take an integer input and count the frequency of each digit (0–9) in it. For example, input 1223334444 should output:

```
1 Digit 1: 1 time
2 Digit 2: 2 times
3 Digit 3: 3 times
4 Digit 4: 4 times
```

## 30. Magic Square Validator

### Problem:

Write a program that checks whether a given  $n \times n$  matrix is a magic square. A magic square is a grid where the sum of every row, column, and both diagonals is the same.

# SQL exercises

---

 Customers Table

CustomerID	CustomerName	City
1	Customer_1	New York
2	Customer_2	New York
3	Customer_3	New York
4	Customer_4	Chicago
5	Customer_5	Chicago
6	Customer_6	New York
7	Customer_7	New York
8	Customer_8	New York
9	Customer_9	New York
10	Customer_10	Chicago
11	Customer_11	Houston
12	Customer_12	Phoenix
13	Customer_13	Houston
14	Customer_14	Chicago
15	Customer_15	Chicago
16	Customer_16	Phoenix
17	Customer_17	Houston
18	Customer_18	Chicago
19	Customer_19	Los Angeles
20	Customer_20	Phoenix
21	Customer_21	Chicago
22	Customer_22	Los Angeles
23	Customer_23	Phoenix
24	Customer_24	New York
25	Customer_25	Houston

 Products Table

ProductID	ProductName	Price
1	Product_1	29.75
2	Product_2	12.86
3	Product_3	19.00
4	Product_4	68.60
5	Product_5	77.10
6	Product_6	29.08
7	Product_7	25.74
8	Product_8	37.62
9	Product_9	59.06
10	Product_10	98.14
11	Product_11	38.13
12	Product_12	90.97
13	Product_13	17.88
14	Product_14	70.26

15	Product_15	24.75
16	Product_16	85.02
17	Product_17	28.75
18	Product_18	74.40
19	Product_19	71.02
20	Product_20	23.62
21	Product_21	29.79
22	Product_22	31.25
23	Product_23	96.69
24	Product_24	77.87
25	Product_25	34.99

## Orders Table

OrderID	CustomerID	ProductID	OrderDate	Quantity
1	12	1	2025-09-30	5
2	7	2	2025-03-29	4
3	6	13	2025-01-24	5
4	3	5	2025-05-10	3
5	17	9	2025-08-15	1
6	19	17	2024-11-13	4
7	16	14	2025-09-19	4
8	21	16	2024-11-10	4
9	18	5	2025-02-23	2
10	25	20	2025-02-20	5
11	18	20	2025-01-02	4
12	25	1	2025-06-17	5
13	18	8	2025-05-12	2
14	12	16	2025-10-31	3
15	20	4	2024-11-16	3
16	9	16	2025-02-04	1
17	6	18	2024-11-11	1
18	18	18	2025-03-24	3
19	14	13	2025-08-28	1
20	21	14	2024-12-28	4
21	1	9	2025-08-12	4
22	2	12	2024-12-24	5
23	17	14	2025-05-22	2
24	25	25	2024-12-28	2
25	5	17	2025-10-03	1

## 20 SQL Join Practice Problems

1. List all orders with customer names and product names.
2. Find all customers who placed an order for "Product\_14".
3. Show total quantity ordered per customer.
4. List customers who have never placed an order.
5. Find products that have never been ordered.
6. Show total revenue per product (Price × Quantity).

7. List orders placed in the last 90 days with customer and product details.
  8. Find the most frequently ordered product.
  9. Show customers who ordered more than 3 different products.
  10. List cities with total number of orders placed.
  11. Find customers who ordered products priced above 50.
  12. Show products ordered by customers from "Chicago".
  13. List customers who ordered the same product more than once.
  14. Show the latest order date for each customer.
  15. Find the top 3 customers by total spending.
  16. List products ordered by more than 5 different customers.
  17. Show average quantity ordered per product.
  18. Find customers who ordered both "Product\_1" and "Product\_25".
  19. List all orders with product price and total cost.
  20. Show customers who ordered products from multiple cities (if product location is added later).
- 

## 5 SQL Subquery Practice Problems

1. Find customers who placed the highest quantity order.
  2. List products with price above the average product price.
  3. Show customers whose total spending is above the average spending.
  4. Find the most expensive product ordered.
  5. List customers who ordered the cheapest product.
-