



Camp1



بِسْمِ اللَّهِ الرَّحْمَنِ الرَّحِيمِ

وَأَنْ لَّيْسَ لِلْإِنْسَانِ إِلَّا مَا سَعَى (39) وَأَنْ سَعْيُهُ سَوْفَ يَرَى (40) ثُمَّ يُجْزَاهُ الْجَزَاءَ الْأَوْفَى (41) وَأَنْ إِلَى رَبِّكَ الْمُنْتَهَى (42)

Algorithms & Data Structure & SQL Queries

Algorithms

Definition:

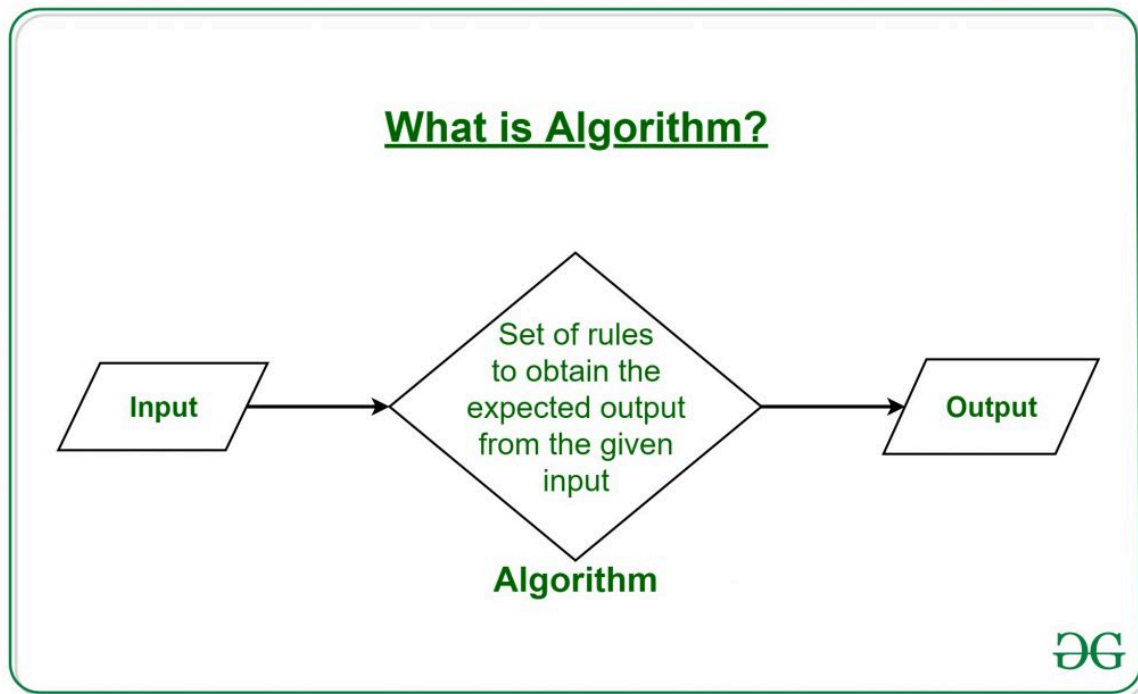
خطوات مرتبة وواضحة بتتحل بيها مشكلة معينة

زي وصفة الطبخ

1. تأخذ مدخلات (input)
2. تعمل شوية خطوات
3. تطلع نتيجة (output)

مثال بسيط:

Algorithm. لو عايز تدور على رقم في قائمة، الخطوات اللي هتعملها علشان تلاقيه هي الـ



Analysis of Algorithms :

تحلله من ناحيتين:

1. الزمن (Time Complexity)

- تقيس عدد الخطوات اللي بياخذها مع حجم البيانات
- Big O: تكتبها بـ
 - $O(1)$ → سريع جدًا
 - $O(n)$ → كل ما الداتا تكبر، الوقت يكبر
 - $O(n^2)$ → بطيء مع البيانات الكبيرة

2. المساحة (Space Complexity)

- بتقيس حجم الذاكرة اللي بيحتاجها علشان يشتغل
- هل بيحتاج مكان إضافي؟ ولا بيشتغل على نفس الداتا؟

...

إليه العوامل اللي تخلي Algorithm احترافي

العامل	معناه
✅ الصحة (Correctness)	هل بيطلع النتيجة الصح لكل الحالات؟
⚡ الكفاءة (Efficiency)	(Time + Space) هل بيشتغل بسرعة مع بيانات كبيرة؟
🔄 القابلية للتكرار	ينفع تستخدمه في سيناريوهات مختلفة؟
🧠 الوضوح (Clarity)	سهل يتفهم ويتشرح؟ الكود مش معقد؟
🔧 القابلية للتنفيذ	ينفع يتحول لكود؟ مفهوش منطق خيالي؟
📈 قابلية التوسع (Scalable)	لما البيانات تكبر 1000 مرة، هيشغل ولا ينهار؟

The Most Famous Problem Solving in Interviews:



1. Write a Program For Armstrong Number

هو عبارة عن رقم مكون من عدد من الارقام لو خدت كل رقم ورفعتو لاس عدد الارقام ثم جمعتهم هيديك نفس الرقم

مثال:

$$153 \Rightarrow n = 3$$

$$1^3 + 5^3 + 3^3 = 153$$

نفكر كذا شوية اي الي هحتاجو عشان احول الكلام دا لكود؟

1. محتاج احافظ علي الرقم الاصلي عشان في النهاية اقارنو بالمجموع
2. هحتاج تحيب رقم الاحاد في كل مره من الرقم الاصلي
3. هحتاج تحذف الرقم الي خلصتو وتعيد ثاني علي باقي الارقام ثم تجمعهم وتقارنهم بالاصلي

Code:

```
namespace Camp1
{
    internal class Program
    {
        static void Main(string[] args)
        {
            int num,temp, rem,sum=0;
            Console.WriteLine("Enter a number: ");
            num = Convert.ToInt32(Console.ReadLine());
            temp = num; // Store the original number for later comparison
            int digits = temp.ToString().Length; // Count the number of digits in the number
            while (temp > 0)
            {
                rem = temp % 10; // Get the last digit
                sum += (int)Math.Pow(rem,digits);
                // Raise the digit to the power of the number of digits and add to sum
                temp /= 10; // Remove the last digit from the number
            }
            if (sum == num)
            {
                Console.WriteLine(num + " is an Armstrong number.");
            }
            else
            {
                Console.WriteLine(num + " is not an Armstrong number.");
            }
        }
    }
}
```



2. Write a Program For Palindrome Number

هو عبارة عن رقم يمينو زي شمالو زي مثلا 121 , 1313

نفكر كدا شوية اي الي هحتاجو عشان احول الكلام دا لكود؟

1. محتاج احافظ علي الرقم الاصلي عشان في النهاية اقارنو بالمجموع
2. هحتاج تجيب رقم الاحاد في كل مره من الرقم الاصلي
3. نعكس الرقم عن طريق نزود خانة في المجموع بالضرب في 10
4. نحذف الرقم الي خلص

Code:

```
namespace Camp1
{
    internal class Program
    {
        static void Main(string[] args)
        {
            int num,temp, rem,sum=0;
            Console.WriteLine("Enter your Palindrome number: ");
            num = Convert.ToInt32(Console.ReadLine());
            temp = num;// Store the original number for later comparison
            while (num > 0)
            {
                rem = num % 10; // Get the last digit
                sum = (sum * 10) + rem; // Build the reversed number
                num = num / 10; // Remove the last digit
            }
            if (temp == sum) // Compare the original number with the reversed number
            {
                Console.WriteLine("The number is a Palindrome.");
            }
            else
            {
                Console.WriteLine("The number is not a Palindrome.");
            }
            Console.ReadKey(); // Wait for user input before closing the console
        }
    }
}
```



3. Swap Two numbers without using third variable

عاوزين نبدل رقمين من غير منعزل مخزن تالت

ببساطة نجمع الرقمين في رقم منهم ونرجع نشيل قيمه الرقم الاصيلي من المجموع

Code:

```
namespace Camp1
{
    internal class Program
    {
        static void Main(string[] args)
        {
            int num1 = 30; int num2 = 70;
            Console.WriteLine("Before Swapping : ");
            Console.WriteLine("num1 = " + num1);
            Console.WriteLine("num2 = " + num2);
            // Swapping without temporary variable
            num1 = num1 + num2; // num1 now becomes 100
            num2 = num1 - num2; // num2 becomes 30 (100 - 70)
            num1 = num1 - num2; // num1 becomes 70 (100 - 30)
            Console.WriteLine("After Swapping : ");
            Console.WriteLine("num1 = " + num1);
            Console.WriteLine("num2 = " + num2);
        }
    }
}
```



4. Write a program to reverse string:

يبقي معاك كلمة وتعكسها

ببساطة الفكرة هنعمل عليها لوب بالعكس

Code:

```
namespace Camp1
{
    internal class Program
    {
        static void Main(string[] args)
        {
            Console.WriteLine("Enter Your String : ");
            string input = Console.ReadLine().ToLower();
            if (string.IsNullOrEmpty(input))
            {
                Console.WriteLine("You entered an empty string.");
                return;
            }
            Console.WriteLine("Reversed String ");
            for (int i = input.Length - 1; i >= 0; i--)
            {
                Console.Write(input[i]);
            }
        }
    }
}
```



5. Count How Many Words in Your String

عوزين نعد عدد الكلمات في جملة

الي هعتمد عليه هنا ان الكلمة بيتفصل بنها وبين الاخري بمسافة او علامة ترقيم

Code:

```
namespace Camp1
{
    internal class Program
    {
        static void Main(string[] args)
        {
            Console.WriteLine("Enter Your String : ");
            string input = Console.ReadLine().ToLower();
            if (string.IsNullOrEmpty(input))
            {
                Console.WriteLine("You entered an empty string.");
                return;
            }

            int count = 0;
            bool isword = false;
            foreach (char c in input)
            {
                if (char.IsWhiteSpace(c) || char.IsPunctuation(c))
                {
                    if (isword)
                    {
                        count++;
                        isword = false; // Reset for the next word
                    }
                }
                else
                {
                    isword = true;
                }
            }
            Console.WriteLine($"The number of words in the string is: {count}");
        }
    }
}
```




6. Write a Program to find a factorial of number

1...3 * (n-3) * (n-2) * (n-1) * n حساب مضروب عدد بيبقي عبارة عن

نستخدم ال recursion

هي تقنية في البرمجة، بتخلي الدالة تستدعي نفسها لحل مشكلة عن Recursion توقف (Base Case) طريق تقسيمها لنسخة أبسط منها، لحد ما توصل لحالة نهائية فيها التكرار.

هنا شرط التوقف ان مضروب ال 0 و 1 يساوي 1 فلما نوصل ليه نقف

Code:

```
namespace Camp1
{
    internal class Program
    {
        static void Main(string[] args)
        {
            Console.WriteLine("Enter Nuber For Factorial : ");
            int n = Convert.ToInt32(Console.ReadLine());

            var result=Factorial(n);

            Console.WriteLine($"Factorial of {n} is {result}");
        }

        static int Factorial(int n)
        {
            if (n == 0 || n == 1)
                return 1;
            else
                return n * Factorial(n-1);
        }
    }
}
```

Data Structure

Definition:

هي طريقة منظمة لتخزين البيانات وترتيبها داخل الذاكرة، علشان Data Structure الـ تسهّل عمليات الوصول، المعالجة، والإدارة بكفاءة عالية حسب نوع المشكلة اللي بتحلّها.

طب ليه نستخدمها ؟

لان الطريقة الي بنخزن بيها البيانات بتأثر علي

- سرعة البحث
- كفاءة التعديل
- ترتيب البيانات
- استهلاك الذاكرة

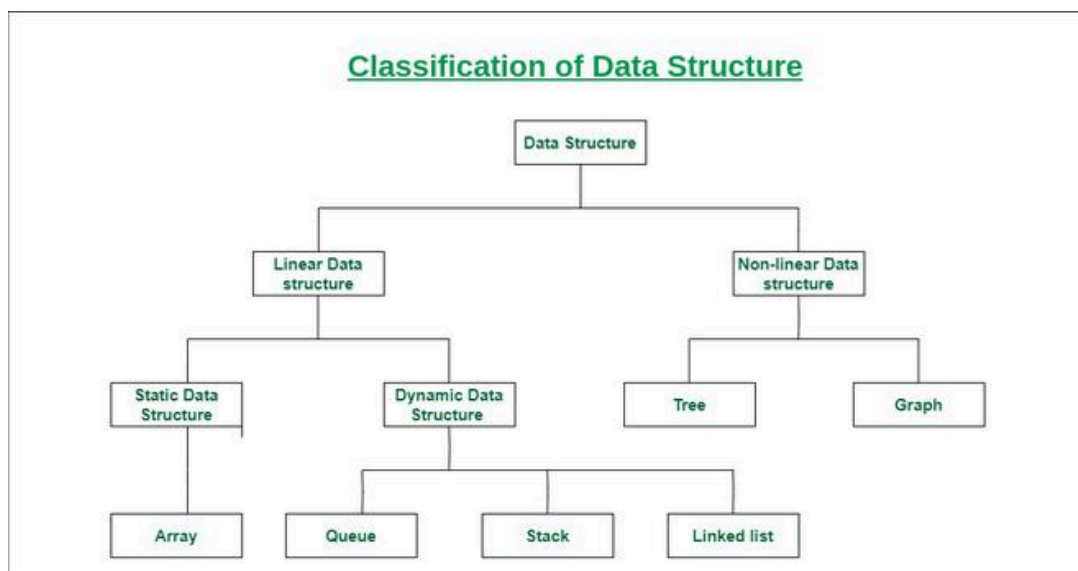
مثال بسيط:

لو معاك أوراق كتير

- لو رميتهم في كيس: صعب تدور على ورقة معينة
- لو رتبتهم حسب التاريخ في دوسيه: تلاقي أي ورقة بسرعة

Data Structure = الدوسيه

Classification of Data Structure:



Static Data Structure : ⇒ Array

هو هيكل بيانات يتم تحديد حجمه وشكله بالكامل قبل تشغيل البرنامج، ولا يمكن تغيير حجمه أثناء التنفيذ، ويُستخدم عندما يكون حجم البيانات معروفًا وثابتًا مسبقًا.

Types of Array:

1. One-Dimensional Array
2. Two-Dimensional Array
3. Multi-Dimensional Array
4. Jagged Array

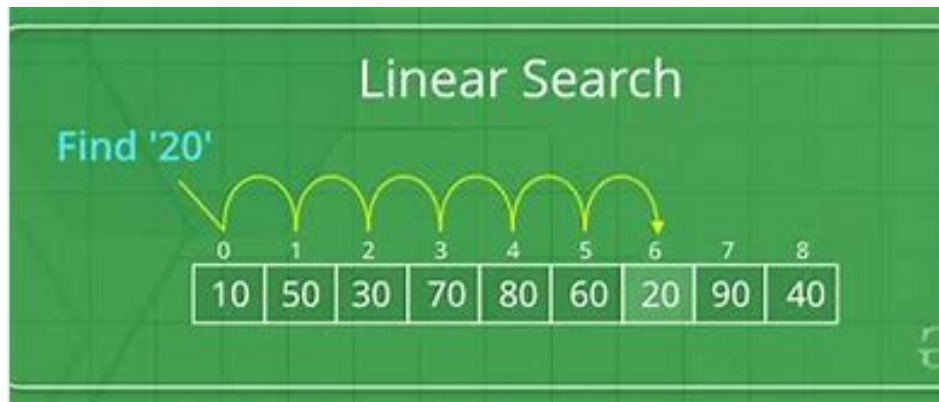
هنتعرف علي اي في ال Array

1. تعريف وإنشاء المصفوفة
2. الوصول للعناصر
3. التعديل على العناصر
4. التكرار عليها (loop)
5. البحث داخلها (Linear / Binary Search)
6. الفرز (Sorting)
7. إيجاد القيم (أكبر، أصغر، مجموع، متوسط...)
8. نسخ المصفوفة
9. التعامل مع المصفوفات متعددة الأبعاد
10. Jagged Arrays استخدام
11. تمريرها للدوال واستقبالها منها
12. IndexOutOfRangeException (مثل) التعامل مع الاستثناءات

Types of Search in Array

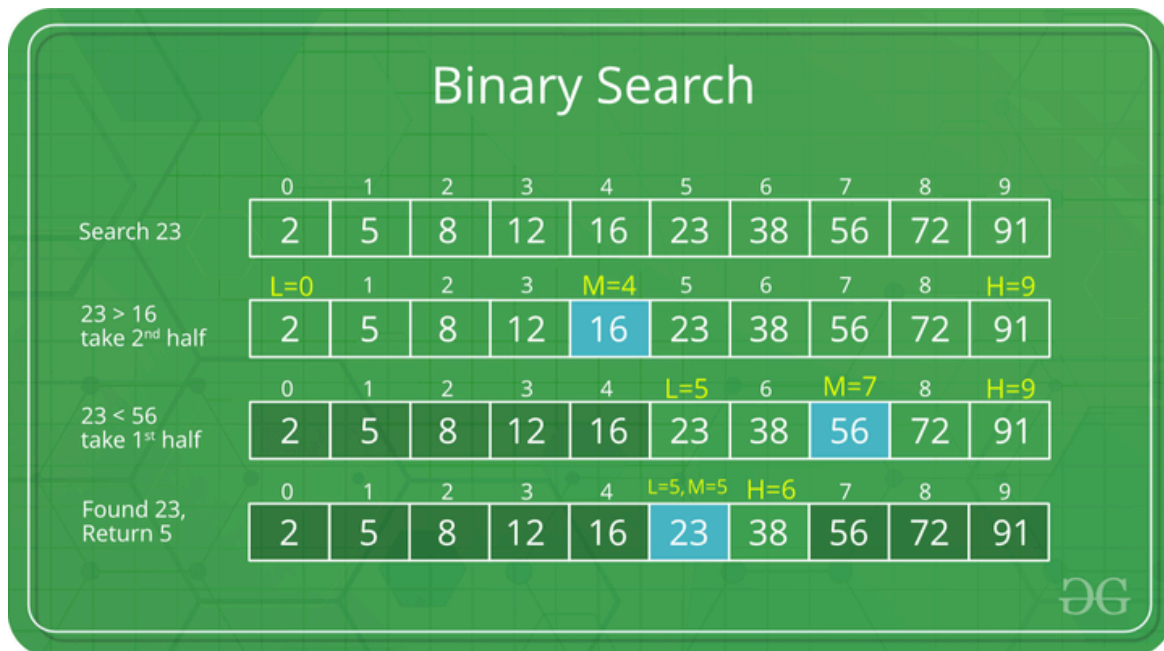
1. Linear Search

هو خوارزمية بحث بسيطة تقوم بفحص كل عنصر في القائمة أو المصفوفة بالتتابع، من البداية للنهاية، حتى تجد العنصر المطلوب أو تنتهي العناصر دون إيجاده.



2. Binary Search

هي خوارزمية بحث فعّالة تعمل على مصفوفة مرتبة، تقوم بتقسيم المصفوفة إلى نصفين في كل خطوة، وتختار النصف الذي قد يحتوي على العنصر المطلوب، وتكرر العملية حتى تعثر على العنصر أو تنفذ النصف بدون إيجاده.



the common types of sorting algorithms:

1. **Bubble Sort**
2. **Selection Sort**
3. **Insertion Sort**
4. **Merge Sort**
5. **Quick Sort**
6. **Heap Sort**
7. **Counting Sort**
8. **Radix Sort**
9. **Bucket Sort**
10. **Shell Sort**

هنشرحم بالتفصيل في Camp2

Code :

```

namespace Camp1
{
    class Program
    {
        static void Main()
        {
            // 1. Define and initialize an array
            int[] arr = { 5, 3, 0, 6, 2 };

            // 2. Access elements
            Console.WriteLine("First element: " + arr[0]);

            // 3. Modify elements
            arr[1] = 10;
            Console.WriteLine("After modification, second element: " + arr[1]);

            // 4. Loop through the array
            Console.WriteLine("All elements:");
            for (int i = 0; i < arr.Length; i++)
            {
                Console.WriteLine(arr[i] + " ");
            }
            Console.WriteLine();

            // 5. Sorting
            Array.Sort(arr);
            Console.WriteLine("After sorting:");
            foreach (int item in arr)
            {
                Console.WriteLine(item + " ");
            }
            Console.WriteLine();

            // 6.1 Search inside the array (Linear Search)
            int target = 0;
            int index = LinearSearch(arr, target);
            Console.WriteLine(index >= 0 ? $"Found {target} at index {index}" : $"Target not found");

            // 6.2 Binary search
            int binaryIndex = BinarySearch(arr, target);
            if (binaryIndex >= 0)
            {
                Console.WriteLine($"Binary search found {target} at index {binaryIndex}");
            }
            else
            {
                Console.WriteLine($"Target not found using binary search");
            }

            // 7. Find max, min, sum, average
            int max = arr[arr.Length - 1]; // After sorting, max is last element
            int min = arr[0]; // Min is first element
            int sum = 0;
            foreach (int item in arr)
            {
                sum += item;
            }
            double avg = (double)sum / arr.Length;
            Console.WriteLine($"Max: {max}, Min: {min}, Sum: {sum}, Avg: {avg:F2}");

            // 8. Copy array
            int[] copy = new int[arr.Length];
            Array.Copy(arr, copy, arr.Length);
            Console.WriteLine("Copy of the array:");
            foreach (int item in copy)
            {
                Console.WriteLine(item + " ");
            }
            Console.WriteLine();

            // 9. Multi-dimensional array (2D)
            int[,] matrix = {
                { 1, 2, 3 },
                { 4, 5, 6 }
            };
            Console.WriteLine("2D array:");
            for (int i = 0; i < matrix.GetLength(0); i++)
            {
                for (int j = 0; j < matrix.GetLength(1); j++)
                {
                    Console.WriteLine(matrix[i, j] + " ");
                }
                Console.WriteLine();
            }

            // 10. Jagged arrays
            int[][] jagged = new int[2][];
            jagged[0] = new int[] { 1, 2 };
            jagged[1] = new int[] { 3, 4, 5 };
            Console.WriteLine("Jagged array:");
            for (int i = 0; i < jagged.Length; i++)
            {
                for (int j = 0; j < jagged[i].Length; j++)
                {
                    Console.WriteLine(jagged[i][j] + " ");
                }
                Console.WriteLine();
            }

            // 11. Passing arrays to methods and returning values
            int total = SumArray(arr);
            Console.WriteLine("Sum of array elements (from method): " + total);

            // 12. Handling exceptions (Like IndexOutOfRangeException)
            try
            {
                Console.WriteLine(arr[100]); // Trying to access an invalid index
            }
            catch (IndexOutOfRangeException)
            {
                Console.WriteLine("Error: Tried to access an invalid index!");
            }
        }

        static int LinearSearch(int[] array, int target)
        {
            for (int i = 0; i < array.Length; i++)
            {
                if (array[i] == target)
                {
                    return i;
                }
            }
            return -1;
        }

        static int BinarySearch(int[] arr, int target)
        {
            int left = 0;
            int right = arr.Length - 1;

            while (left <= right)
            {
                int mid = left + (right - left) / 2;

                if (arr[mid] == target)
                {
                    return mid; // Found target, return index
                }

                if (arr[mid] < target)
                {
                    left = mid + 1; // Search right half
                }
                else
                {
                    right = mid - 1; // Search left half
                }
            }

            return -1; // Target not found
        }

        static int SumArray(int[] array)
        {
            int sum = 0;
            foreach (int item in array)
            {
                sum += item;
            }
            return sum;
        }
    }
}

```



لو مهتم تستخدم ال Array in high performance

see this post ⇒ https://www.linkedin.com/posts/ahmed-hany-899a9a321_%D8%A7%D9%84%D8%B3%D9%84%D8%A7%D9%85-%D8%B9%D9%84%D9%8A%D9%83%D9%85-%D8%A7%D8%B2%D8%A7%D9%8A-%D8%AA%D8%B3%D8%AA%D8%AE%D8%AF%D9%85-%D8%A7%D9%84array-%D8%A8performance-activity-7294802439600226304-iwEX?utm_source=share&utm_medium=member_android&rcm=ACoAAFF_KXUB_4sXiwxIEmO-rsGzGT4OI4j2cLU

SQL Queries



Query1

The question from the image is as follows:

Problem Statement: Combine Two Tables (LeetCode Problem 175)

Tables:

1. Table: Person

+-----+-----+	
Column Name	Type
+-----+-----+	
PersonId	int
FirstName	varchar
LastName	varchar
+-----+-----+	

PersonId is the primary key column for this table.

2. Table: Address

+-----+-----+	
Column Name	Type
+-----+-----+	
AddressId	int
PersonId	int
City	varchar
State	varchar
+-----+-----+	

AddressId is the primary key column for this table.

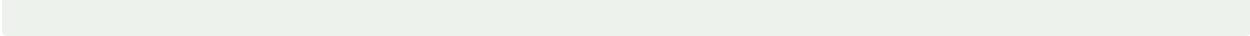
Task:

Write an SQL query to provide the following information for each person in the **Person** table, regardless of whether there is an address for each of those people.

بيقلك عاوز يجيب بيانات الاشخاص سواء ليهم عناوين او لا

Query:

```
Select P.PersonId, P.FirstName,P.LastName A.City, A.State
from Person P
left Join Address A
on P.PersonId = A.PersonId
```



Query2

Problem Statement: Second Highest Salary (LeetCode Problem 176)

Task:

Write an SQL query to get the second highest salary from the `Employee` table.

Example:

Given the following `Employee` table:

```

+----+-----+
| Id | Salary |
+----+-----+
| 1  | 100    |
| 2  | 200    |
| 3  | 300    |
+----+-----+

```

The query should return `200` as the second highest salary. If there is no second highest salary, the query should return `NULL`.

Expected Output:

```

+-----+
| SecondHighestSalary |
+-----+
| 200                  |
+-----+

```

ازاي نجيب ثاني اكبر مرتب؟

عن طريق نحسب نعمل sub query ونقارنو باكبر مرتب من الجدول بدون

query:

```

Select max(Salary) as SecondHighestSalary
from Employee Where Salary < (Select max(Salary) from Employee)

```

او ان ندي كل صف رتبه بدون تكرار ومرتبين تنازلي وناخذ ثاني رتبه يبقي هو ثاني اكبر مرتب

ودي افضل لان لو عاوز ثالث اكبر هتجيب الرتبه 3 وهكذا

query:

```
WITH RankedSalaries AS (  
  SELECT  
    Salary,  
    DENSE_RANK() OVER (ORDER BY Salary DESC) AS SalaryRank  
  FROM Employees  
)  
SELECT Salary as SecondHighestSalary  
FROM RankedSalaries  
WHERE SalaryRank = 2;
```



Query3

Problem Statement: Nth Highest Salary (LeetCode Problem 177)

Task:

Write an SQL query to get the **nth highest salary** from the `Employee` table.

Example:

Given the following `Employee` table:

```

+----+-----+
| Id | Salary |
+----+-----+
| 1  | 100    |
| 2  | 200    |
| 3  | 300    |
+----+-----+

```

If `n = 2`, the query should return `200` as the second highest salary. If there is no nth highest salary, the query should return `NULL`.

Expected Output:

For `n = 2`:

```

+-----+
| getNthHighestSalary |
+-----+
| 200                  |
+-----+

```

نفس فكرة ال query السابقه بس تكون dynamic اكثر بمعنى هنعمل دالة اديها الرتبه الي عوزها
ويحسبلي المطلوب

Query:

```

CREATE FUNCTION GetNthHighestSalary (@N INT)
RETURNS INT
AS
BEGIN
    DECLARE @Result INT;

    SELECT @Result = Salary

```

```
FROM (  
    SELECT  
        Salary,  
        DENSE_RANK() OVER (ORDER BY Salary DESC) AS SalaryRank  
    FROM Employee  
    ) AS Ranked  
WHERE SalaryRank = @N;  
  
RETURN @Result;  
END;
```



Query4

Problem Statement: Rank Scores

Task:

Write an SQL query to rank scores. If there is a tie between two scores, both should have the same ranking. Note that after a tie, the next ranking number should be the next consecutive integer value. In other words, there should be no "holes" between ranks.

Example:

Given the following `Scores` table:

Id	Score
1	3.50
2	3.65
3	4.00
4	3.85
5	4.00
6	3.65

Your query should generate the following report (ordered by highest score):

Score	Rank
4.00	1
4.00	1
3.85	2
3.65	3
3.65	3
3.50	4

من الأعلى للأقل، وتدي لكل درجة ترتيب (scores) عاوزك ترتب الدرجات

- لو درجتين متساويتين → ياخدوا نفس الترتيب
- الترتيب اللي بعدهم يبقى على طول بالترتيب (يعني من غير ما نسيب أرقام)

Query :

```
SELECT  
  Score,  
  DENSE_RANK() OVER (ORDER BY Score DESC) AS Rank  
FROM Scores;
```



Query 5

Problem Statement: Consecutive Numbers

Task:

Write an SQL query to find all numbers that appear at least three times consecutively.

- Return the result table in any order.
- The query result format is provided in the example below.

Example:

Given the following `Logs` table:

```
+----+-----+
| Id | Num |
+----+-----+
| 1 | 1 |
| 2 | 1 |
| 3 | 1 |
| 4 | 2 |
| 5 | 1 |
| 6 | 2 |
| 7 | 2 |
+----+-----+
```

The expected output is:

```
+-----+
| ConsecutiveNums |
+-----+
| 1 |
+-----+
```

Explanation:

- The number `1` appears consecutively three times (rows with `Id = 1, 2, 3`), so it should be included in the result.
- The number `2` does not appear consecutively three times, so it should not be included.

عوزين نجيب الرقم المكرر 3 مرات وري بعض

عندنا دالتين جمال اسمهم lag , lead

بتجيب العنصر السابق : lag

بتجيب العنصر التالي: lead

Query

--with lag

```
SELECT Num AS ConsecutiveNums
FROM (
  SELECT
    Num,
    LAG(Num, 1) OVER (ORDER BY Id) AS prev_num1,
    LAG(Num, 2) OVER (ORDER BY Id) AS prev_num2
  FROM Logs
) sub
WHERE Num = prev_num1 AND Num = prev_num2;
```

--with lead




```
SELECT Num AS ConsecutiveNums
FROM (
  SELECT
    Num,
    lead(Num, 1) OVER (ORDER BY Id) AS prev_num1,
    lead(Num, 2) OVER (ORDER BY Id) AS prev_num2

  FROM Logs
) sub
WHERE Num = prev_num1 AND Num = prev_num2;
```



لو استفدت لو بمعلومة متنساش دعوة حلوة في الاسام المفترجة دي

 **My personal accounts links**

 LinkedIn	https://www.linkedin.com/in/ahmed-hany-899a9a321? utm_source=share&utm_campaign=share_via&utm_content=profile&utm_medium=android_app
 WhatsApp	https://wa.me/qr/7KNUQ7ZI3KO2N1
 Facebook	https://www.facebook.com/share/1NFM1PfSjc/