# Abdallah Hany

# Part01

* Problem: Write a program that:
  + Initializes a one-dimensional array in three different ways (new int[size], initializer list, and Array syntax sugar).
  + Assigns values to each element in the array and prints them.
  + Demonstrates an IndexOutOfRangeException.
* Question: What is the default value assigned to array elements in C#?

Default value is 0.

* Problem: Write a program to:
  + Create two arrays (arr1 and arr2).
* Perform a shallow copy and demonstrate how modifying one affects the other.

o Perform a deep copy using the Clone method and show that modifications do not affect the copied array.

* Question: What is the difference between Array.Clone() and Array.Copy()?

Array Copy: copy only address.  
Array Clone copy object data in a new created object but, in different address.

* Problem: Write a program to:
  + Create a 2D array with student grades (3 students, 3 subjects each).
  + Take input from the user to fill the array.
  + Print the grades for each student using nested loops.
* Question: What is the difference between GetLength() and Length for multidimensional arrays?

Length: prints the sum of the tow indexes of 2D array  
Get Length :accepts 0 to print rows and 1 to print columns

* Problem: Write a program that:
  + Demonstrates at least 5 array methods (Sort, Reverse, IndexOf, Copy, Clear).
  + Explains the changes before and after applying each method.
* Question: What is the difference between Array.Copy() and Array.ConstrainedCopy()?

Array constrained copy: Handles any exception (safe copy), copy all or no.

Array copy: Never handles exceptions, it may copy then throw exception.

* Problem: Create a program that:
* Uses a for loop to print all elements of a 1D array.
* Uses a foreach loop to print all elements of the same array.
* Uses a while loop to print all elements in reverse order.
* Question: Why is foreach preferred for read-only operations on arrays?

It’s automatically iterating over all elements without requiring index management.

* Problem: Write a program that:
  + - Repeatedly asks the user for a positive odd number.
    - Uses defensive coding to validate input using int.TryParse and a do-while loop.
* Question: Why is input validation important when working with user inputs?

It prevents errors and crashes, protects against malicious input.

* Problem: Write a program to:
  + - Create a 2D array with fixed values.
    - Print the array elements in a matrix format (rows and columns).
* Question:How can you format the output of a 2D array for better readability?

By spacing and displaying in rows.

* Problem: Write a program that:
  + - Asks the user to enter a month number.
    - Uses an if-else statement to determine the month name.
    - Uses a switch statement to perform the same task.
* Question: When should you prefer a switch statement over if-else?

When checking a single variable against multiple constant values.

* Problem: Write a program to:
  + - Sort an array of integers using Array.Sort().
    - Search for a specific value using Array.IndexOf() and Array.LastIndexOf(). Question: What is the time complexity of Array.Sort()?

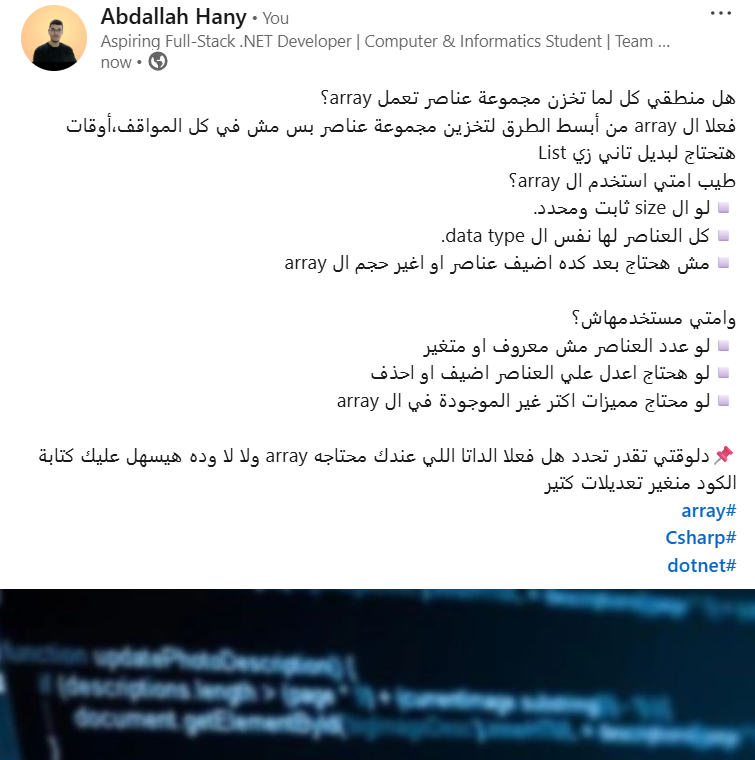
T(n)=nlog(n)

* Problem: Write a program that:
  + - Creates an array of integers.
    - Uses a for loop to calculate and print the sum of all elements.
    - Uses a foreach loop to calculate the same sum.
* Question: Which loop (for or foreach) is more efficient for calculating the sum of an array, and why?

For, because it’s never using an enumerator as for each.

# Part02

1. LinkedIn article about loops statements in Csharp



1. Define an enum called DayOfWeek with values: Monday, Tuesday, Wednesday, Thursday, Friday, Saturday, Sunday.

Write a program that takes an integer input from the user (1-7) and prints the corresponding day using the enum.

Use Enum.Parse to convert an integer to an enum value.

1. What happens if the user enters a value outside the range of 1 to 7?

It prints the value as it is.

# Part03 Bonus

1. self study report
2. what’s the default size of stack and heap and what are the consideration

Stack:

64-bit process🡪1MB

* Stack memory is limited. Deep recursion or very large local variables (like big arrays) can cause a stack over flow exception.
* The stack is fast because it’s managed automatically (LIFO order).
* Each thread has its own stack.

Heap:

Not fixed but nearly 3GB.

* Heap memory is slower to allocate compared to the stack because it requires garbage collection.
* Managed heap in .NET is divided into generations (Gen 0, Gen 1, Gen 2, Large Object Heap).
* Large objects (> 85 KB) go to the Large Object Heap (LOH), which is collected less frequently.

1. what is time complexity

Time complexity is a measure of how many steps an algorithm takes to finish.