Part01

- **Question: What is the difference between int.Parse and Convert.ToInt32 when handling null inputs?**
  With int.Parse, if the input is null, an exception will be thrown.
  With Convert.ToInt32, if the input is null, it will return 0.

- **Question: Why is TryParse recommended over Parse in user-facing applications?**
  When we use Parse and the user enters invalid input, it throws an exception and stops the program.
  But when we use TryParse, it does not throw an exception, it handles the input safely, and allows you to check whether the input is valid before using it.

- **Question: Explain the real purpose of the GetHashCode() method.**
  Its purpose is to return a number (hash code) that is used to speed up searching and comparison in hash-based collections.

- **Question: What is the significance of reference equality in .NET?**
  It is important to know whether changing one reference will affect the other, especially with reference types like classes and arrays.
  It also helps determine whether two variables refer to the same object in memory or just have the same value.

- **Question: Why string is immutable in C# ?**
  In C#, a string is immutable, which means that any modification to the text creates a new object instead of changing the original object.
  Although string is a reference type, it behaves like a value type in terms of behavior, because any change does not affect the original object.

- **Question: How does StringBuilder address the inefficiencies of string concatenation?**
  Each time we modify a string, a new object is created, which wastes memory.
  In contrast, StringBuilder modifies the text within the same object without creating a new one, which preserves memory and improves performance.

- **Question: Why is StringBuilder faster for large-scale string modifications?**
  In C#, a string is immutable : each modification creates a new object → consuming time and memory.
  StringBuilder is mutable : it can modify the text within the same object without creating a new one.
  This means it is much faster and more memory-efficient when you have many modifications or large texts.

- **Question: Which string formatting method is most used and why?**
  String interpolation ($) is the most used because it is easier to read and write, and it automatically arranges variables, making it safe from errors.

- **Question: Explain how StringBuilder is designed to handle frequent modifications compared to strings.**
  StringBuilder is designed for frequent modifications because it is mutable and uses an internal buffer, allowing changes within the same object without creating new string objects, making it faster and more memory-efficient than strings.

---

part02

- **What's Enum data type, when is it used? And name three common built_in enums used frequently?**
  Enum in C# is a data type that defines a set of named constant values. It is used when you want to represent a fixed set of related options in a readable and maintainable way.
  Three commonly used built-in enums:
  DayOfWeek – represents the days of the week.
  ConsoleColor – represents console text and background colors.
  FileMode – represents different modes of opening a file

- **what are scenarios to use string Vs StringBuilder?**
  Use string when:
  The text is small or rarely changed.
  You want clean and simple code without performance concerns.

  Use StringBuilder when:
  The text is large or frequently modified.
  You want better memory usage and faster performance.

- **LinkedIn article about string immutability:**

Mohamed Metwally · You
Software Engineer | Full Stack Web Developer
1m ·

Why Strings Are Immutable in C# and What That Means for Your Code

Many developers assume that string behaves like value types such as int or double and is stored on the stack. This is a misconception. In C#, string is a reference type. The actual characters are stored in the heap, while the variable holds a reference.

Interestingly, even though string is a reference type, it acts like a value type because it is immutable. Any modification creates a new object. For example, if you write:

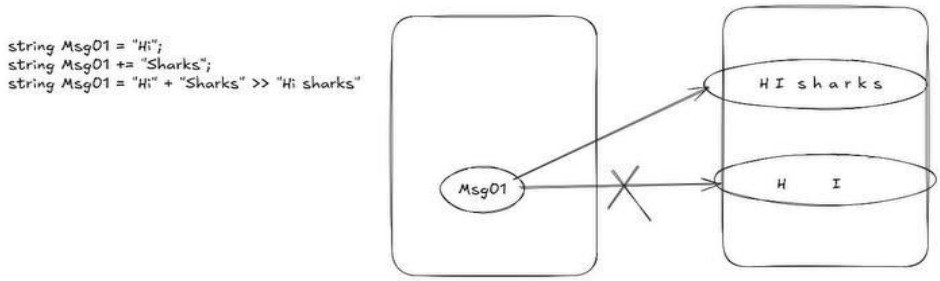
```
string x = "Hi";
x += ", Mohamed"; // A new object is created
```

Here, the original "Hi" remains unchanged, and x now points to "Hi, Mohamed". Frequent modifications like this can waste memory and reduce performance.

Enter StringBuilder

StringBuilder is designed for frequent text modifications. It modifies the text in the same object, avoiding the creation of new objects. This makes it faster and more memory-efficient. For example:

```
StringBuilder sb = new StringBuilder("Hi");
sb.Append(", Mohamed"); // Modifies the same object
```



Part03

- **what meant by user defined constructor and its role in initialization?**
  A user-defined constructor is a constructor created by the programmer to initialize an object with specific values, ensuring it starts in a valid state.

- **compare between Array and Linked List**
  Array:
  Fixed size
  Very fast access
  Insertion and deletion are slow

  LinkedList:
  Dynamic size
  Slower access
  Insertion and deletion are fast
