**1) Question: Why can't a struct inherit from another struct or class in C#?**

C# structs cannot inherit from another struct or class because they are value types, designed for lightweight data containers without the overhead of inheritance. Structs can only implement interfaces to support polymorphism without inheritance.

**2) Question: How do access modifiers impact the scope and visibility of a class member?**

Access modifiers define a class member's visibility. For example, public makes it accessible everywhere, private restricts it to the containing class, and protected allows access in derived classes.

**3) Why is encapsulation critical in software design?**

Encapsulation restricts direct access to an object's data, ensuring control over modifications. This improves code maintainability, security, and reduces unintended interference.

**4) Question: what is constructors in structs?**

Structs in C# can have parameterized constructors to initialize fields at creation but cannot have parameterless constructors (except with readonly structs). The compiler ensures all fields are initialized before use.

**5) Question: How does overriding methods like ToString() improve code readability?**

Overriding ToString() provides meaningful string representations of objects. This enhances debugging and logging by clearly conveying object states.

**6) What is copy constructor?**

A copy constructor is a special constructor that creates a new object by copying the values of an existing object. C# does not have built-in support for copy constructors, but you can define one manually by taking an object of the same type as a parameter.

**7) What is Indexer, when used, as business mention cases u have to utilize it?**

An indexer in C# allows an object to be accessed like an array using the [] syntax. It is useful when a class logically represents a collection of data.

**Usage**

An indexer is defined using the this keyword and provides a way to manage data without exposing the underlying structure directly.

**Business Case Examples**

1. **Data Lookup in Collections**  
   A CustomerCollection class that stores multiple customers can use an indexer to retrieve a customer by index or key.
2. **Mapping Keys to Values**  
   A configuration class might use an indexer to retrieve or update settings using keys.
3. **Custom Data Structures**  
   Classes like a matrix, table, or grid can use indexers to allow intuitive access to elements by row and column.