C# Theoretical Questions - Answers

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

Structs are value types and implicitly inherit from System.ValueType.
C# does not support inheritance between structs or from classes to maintain value-type behavior
and memory efficiency.

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

Access modifiers (private, internal, public, protected) control where a member can be accessed
from.
They enforce encapsulation and protect data from unauthorized access.

3) Why is encapsulation critical in software design?

Encapsulation protects internal object data, ensures controlled access, improves maintainability,
and prevents unintended modification of data.

4) What are constructors in structs?

Constructors in structs are special methods used to initialize struct fields when creating a new
instance.

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

Overriding ToString() provides meaningful and formatted output instead of the default type name,
making debugging and logging clearer.

6) How does memory allocation differ for structs and classes in C#?

Structs (value types) are stored in the stack or inline within heap objects.
Classes (reference types) are stored in the heap, and variables hold references to their memory
location.

7) What is a Copy Constructor?

A copy constructor creates a new object by copying the values of another object of the same type.

8) What is an Indexer and when is it used?

An indexer allows an object to be accessed like an array using [] notation.
It is used in custom collections, data containers, caching systems, and business entities that require
indexed access