🔷 Partial Class

- Allows splitting a class definition across multiple files.
- Useful in large projects or when working with auto-generated code (e.g., WinForms Designer, EF models).

🔷 Private Constructor

- Used to prevent direct instantiation of a class.
- Core part of the Singleton Design Pattern → ensures only one instance exists.

🔷 Record (C# 9.0)

- New reference type for immutable data models.
- Provides value-based equality, concise syntax, and with expressions.
- Great for DTOs and read-only models.

🔷 Default Constructor of a Class

- If you don't define a constructor, the CLR automatically provides a parameterless constructor.
- Initializes fields with default values (0 for int, null for refs...).
- If you define any constructor yourself, the CLR does not add a default one.

🔷 Performance: Struct vs Class

- Struct (Value Type) → stored on the stack, faster in many cases, but causes overhead with boxing/unboxing.
- Class (Reference Type) → stored on the heap, requires garbage collection, more flexible (supports inheritance).
- Rule of thumb:
  - Use struct for small, lightweight data.
  - Use class for complex/large objects or when inheritance is required.

🔷 Save Memory Indirectly

- Structs can save memory when small.
- But large structs can hurt performance → because copying passes the entire struct by value.

🔷 Dependency Inversion Principle (SOLID - D)

- Depend on Abstractions, not on Concrete implementations.
- Example: Code against IRepository instead of SqlRepository.
- This makes the code more flexible, testable, and maintainable.

🔷 Couple Operators (Operator Overloading)

- In C#, you can overload operators like +, -, ==.
- Example: Override == for comparing object values instead of just references.

🔷 Source Browser (Reference Source)

- Tool/website for browsing .NET BCL (Base Class Library) source code.
- Helps in studying how .NET classes and overrides are implemented.

🔷 Override in BCL

- Most .NET base classes override methods like:
    - ToString() ➔ to return meaningful output.
    - Equals() ➔ for logical equality comparison.
    - GetHashCode() ➔ for dictionary/hash usage.