What People Mean by “Struct Is Like a Class (Before)”?

It usually refers to the fact that both **structs** and **classes** in C#:

* Can have fields, properties, methods, and constructors.
* Can be instantiated using new.
* Are defined with similar syntax:

struct Person { ... } // Like a class

class Student { ... }

So at a glance, they look and act similarly — that's where the comparison comes from.

**Key Differences Behind the Scenes**

* **Structs are Value Types**, stored in the stack
* **Classes are Reference Types**, stored in the heap

**Other Important Differences:**

* Structs **don’t support inheritance**, but can implement interfaces.
* Structs are **copied** when passed around (no shared reference).
* Classes are **referenced**, meaning multiple variables can point to the same object.

### Why This Matters in Backend Logic

If you're modeling lightweight data (like coordinates, time ranges, or exam flags), structs are efficient. But if you're modeling a full entity with behavior and complexity (like Student or ExamSession), classes give more flexibility.