Day 9

Part 01

1. **Why is it recommended to explicitly assign values to enum members?**
   It ensures stable numeric mapping, maintains compatibility with stored data, and controls value changes safely.

2. **What happens if an enum value exceeds the underlying type's range?**
   Overflow or compilation/runtime errors may occur depending on the context and type handling.

3. **What is the purpose of the virtual keyword when used with properties?**
   It allows derived classes to override the property and supports polymorphism.

4. **Why can't you override a sealed property or method?**
   Because sealed members are restricted to prevent modification and preserve implementation integrity.

5. **What is the key difference between static and object members?**
   Static members belong to the class and are shared, while object members belong to specific instances.

6. **Can all operators be overloaded in C#? Explain.**
   No. Only specific operators can be overloaded because some core language operators cannot be customized.

7. **When should you consider changing the underlying type of an enum?**
   When optimizing memory usage or when integrating with external systems that require specific data sizes.

8. **Why can't a static class have instance constructors?**
   Because a static class cannot be instantiated and only contains static members.

9. **What are the advantages of using Enum.TryParse over direct parsing?**
   It handles invalid input safely without throwing exceptions.

10. **What is the difference between overriding Equals and ==?**
    Equals compares logical values, while == compares references by default unless overloaded.

11. **Why is overriding ToString beneficial?**
    It provides a meaningful string representation of custom objects and improves readability.

12. **Can generics be constrained to specific types?**
    Yes. Generic constraints limit allowed types to ensure type safety and required behavior.

13. **What are the differences between generic methods and generic classes?**
    Generic classes define type flexibility at the class level, while generic methods define it at the method level.

14. **Why is a generic swap method preferable?**
    It provides reusability and type safety without needing separate implementations for each type.

15. **How does overriding Equals in related classes improve search accuracy?**
    It enables logical comparison instead of reference comparison, improving matching results.

16. **Why is == not implemented by default for structs?**
    Because structs are value types and equality behavior must be explicitly defined based on application needs.

Part 02

1. **What is generalization using generics?**
   It is writing reusable and type-safe code that works with multiple data types while avoiding duplication.

2. **What is hierarchy design in real business?**
   It is organizing system components using parent-child relationships to model organizational structure and responsibilities.