2- What we mean by Generalization concept using Generics?
Generalization with Generics means creating classes, methods, or interfaces that can work with different data types without rewriting the same logic for each type. Instead of designing separate solutions for every data type, we use a **generic placeholder** (like `<T>`) to represent a type that will be specified when the class or method is used.

This provides:

- **Reusability**: Write once, use with any data type.
- **Type Safety**: Catch errors at compile time instead of runtime.
- **Flexibility**: The same structure can handle multiple scenarios.

For example, instead of writing separate methods for integers, strings, or doubles, a generic method can handle all of them while maintaining type safety.

3- What we mean by hierarchy design in real business?
Hierarchy design in real business refers to structuring classes in a way that reflects the real-world organization or workflow of a company. It is about designing a **class hierarchy** where higher-level (parent) classes represent general concepts, while lower-level (child) classes represent more specific roles or entities.

For example, in a business context:

- At the top, you might have a general `Employee` class.
- Then, subclasses like `Manager`, `Developer`, and `Intern` inherit from it.
- Each subclass has additional behaviors or attributes specific to its role.

This mirrors how companies are structured — starting with a broad category (employees) and breaking it down into more specialized roles. Hierarchy design helps create software that models real-world business logic, making it easier to maintain, extend, and understand.