Decorator Pattern Demo: Problem Statement

- Imagine you're developing a coffee ordering system for a cafe. Customers can order different types of coffee, such as Espresso, Mocha, or House Blend, and customize them with various toppings like whipped cream or soy milk. You need to support a way to handle these customizations dynamically and add these toppings in any order.

Why use the decorator pattern?

- The Decorator Pattern is suitable for this scenario because it allows you to add new features or customizations to objects dynamically at runtime. It enables the creation of flexible and reusable code by wrapping objects in layers of decorators. This pattern helps to keep the core components (base coffee types) separate from optional enhancements (toppings or extra flavors), promoting code maintainability and scalability.