**Título: Aplicação do Padrão Builder para Criação de Pizzas**

---

## Objetivo:

O objetivo deste laboratório é implementar o padrão de projeto **Builder** para criar objetos do tipo `Pizza`. Você deverá permitir que diferentes tipos de pizzas sejam construídas passo a passo, com características configuráveis como tamanho, tipo de massa e ingredientes. Além disso, um **Diretor** será responsável por definir receitas predefinidas.

---

## Descrição do Problema:

Uma pizzaria precisa de um sistema que permita configurar diferentes tipos de pizzas com características específicas. Algumas pizzas seguem receitas padrão, enquanto outras podem ser personalizadas pelo cliente. Você deve implementar as classes necessárias para atender aos seguintes requisitos:

1. Cada pizza pode ter:
    - Tamanho (`pequena`, `média`, `grande`);
    - Tipo de massa (`tradicional`, `fina`, `recheada`);
    - Lista de ingredientes opcionais.
2. Crie dois tipos de pizzas predefinidas:
    - **Pizza Margherita**:
        - Tamanho: `grande`;
        - Massa: `fina`;
        - Ingredientes: `Queijo`, `Tomate`, `Manjericão`.
    - **Pizza Pepperoni**:
        - Tamanho: `média`;
        - Massa: `tradicional`;
        - Ingredientes: `Queijo`, `Pepperoni`.
3. O cliente deve ter a possibilidade de criar uma pizza personalizada, escolhendo:
    - O tamanho;
    - O tipo de massa;
    - Ingredientes adicionais de sua preferência.
4. Utilize o padrão **Builder** para separar a construção da pizza em passos.

---

## Tarefas:

1. **Criar a classe `Pizza`:**
    - A classe deve representar o produto final e conter os atributos:
        - `size` (tamanho);
        - `dough` (massa);

        - `toppings` (ingredientes).
    - Implemente um método `display` que exibe os detalhes da pizza.
2. **Criar a interface `PizzaBuilder`:**
    - Defina os métodos para construir as pizzas:
        - `reset()`: Reseta o estado da construção.
        - `setSize(size: string)`: Define o tamanho da pizza.
        - `setDough(dough: string)`: Define o tipo de massa.
        - `addTopping(topping: string)`: Adiciona um ingrediente.
        - `getResult()`: Retorna a pizza construída.
3. **Criar dois Builders concretos:**
    - `MargheritaPizzaBuilder`: Implementa a receita de uma pizza Margherita.
    - `PepperoniPizzaBuilder`: Implementa a receita de uma pizza Pepperoni.
4. **Criar a classe `PizzaDirector`:**
    - O diretor deve definir a sequência de passos para criar as pizzas predefinidas (Margherita e Pepperoni).
    - Permita que o diretor troque o Builder durante a execução.
5. **Criar o código cliente:**
    - Use o `PizzaDirector` para criar pizzas Margherita e Pepperoni.
    - Crie uma pizza personalizada diretamente com um Builder.

---

## Requisitos de Implementação:

- Utilize **TypeScript** para a implementação.
- Assegure que os métodos de configuração no Builder sejam encadeáveis (chaining).
- As pizzas predefinidas devem ser criadas utilizando o Diretor.
- A pizza personalizada deve ser criada sem o uso do Diretor, diretamente pelo Builder.

---

## Entregáveis:

1. Código completo com a implementação do padrão Builder para o problema proposto.
2. Uma saída no console que demonstre:
    - A criação de uma pizza Margherita.
    - A criação de uma pizza Pepperoni.
    - A criação de uma pizza personalizada pelo cliente.