Tarefa03.py > ...

```python
# Nome: Victor Mariano dos Santos Gomes
# RA: 2171392221041

# Nome: Susana Jesus Reis Alves
# RA: 2171392211029


# Implementação do padrão Bridge para tipos de massas de pizza

# Classe base para massas de pizza
class MassaPizza:
    def preparar(self):
        pass

    def assar(self):
        pass

    def cortar(self):
        pass

    def embalar(self):
        pass

# Subclasse representando massa fina
class MassaFina(MassaPizza):
    def preparar(self):
        return "Preparando massa fina"

    def assar(self):
        return "Assando em alta temperatura"

    def cortar(self):
        return "Cortando em pedaços triangulares"

    def embalar(self):
        return "Embalando na caixa tradicional"

# Subclasse representando massa grossa
class MassaGrossa(MassaPizza):
    def preparar(self):
        return "Preparando massa grossa"
```

```
        return "Preparando massa grossa"

    def assar(self):
        return "Assando em temperatura média"

    def cortar(self):
        return "Cortando em pedaços quadrados"

    def embalar(self):
        return "Embalando na caixa especial"

# Implementação do padrão Composite para criar pedidos de pizza compostos

# Classe para criar pedidos de pizza compostos
class PedidoPizza:
    def __init__(self):
        self.pizzas = []  # Lista para armazenar as pizzas no pedido

    def adicionar_pizza(self, pizza):
        self.pizzas.append(pizza)  # Adiciona uma pizza ao pedido

    def preparar(self):
        return "\n".join([pizza.preparar() for pizza in self.pizzas])  # Preparação de todas as pizzas

    def assar(self):
        return "\n".join([pizza.assar() for pizza in self.pizzas])  # Cozimento de todas as pizzas

    def cortar(self):
        return "\n".join([pizza.cortar() for pizza in self.pizzas])  # Corte de todas as pizzas

    def embalar(self):
        return "\n".join([pizza.embalar() for pizza in self.pizzas])  # Embalagem de todas as pizzas

# Implementação do padrão Factory para criar objetos de pizza
class FabricaMassaFina(FabricaPizza):
    def criar_pizza(self):
        return MassaFina()

class FabricaMassaGrossa(FabricaPizza):
    def criar_pizza(self):
        return MassaGrossa()
```

```python
# Exemplo de uso com as fábricas específicas
fabrica_massa_fina = FabricaMassaFina()
fabrica_massa_grossa = FabricaMassaGrossa()
pedido = PedidoPizza()

# Cria pizzas usando as fábricas específicas
massa_fina_pizza = fabrica_massa_fina.criar_pizza()
massa_grossa_pizza = fabrica_massa_grossa.criar_pizza()

# Adiciona as pizzas ao pedido
pedido.adicionar_pizza(massa_fina_pizza)
pedido.adicionar_pizza(massa_grossa_pizza)

# Imprime as etapas para cada pizza no pedido
print("Preparação:")
print(pedido.preparar())

print("\nCozimento:")
print(pedido.assar())

print("\nCorte:")
print(pedido.cortar())

print("\nEmbalagem:")
print(pedido.embalar())


```