Solución Desafío - Métodos y atributos

Requerimiento 1

Archivo pizza.py

1. Crear la clase Pizza

```
# Define la clase
class Pizza:
```

2. Agrega atributos de clase con sus valores por defecto

```
class Pizza:
    # Agrega atributo tamaño
    tamanio = "familiar"

    # Agrega atributo precio
    precio = 10000
```

Requerimiento 2

Archivo pizza.py

1. Agregar método estático con 2 parámetros

```
    @staticmethod
    def validar_elemento(solicitado, posibles):
```

2. Escribir cuerpo del método. Hay muchas formas de lograr lo solicitado, esta es una de ellas (más simple de escribir).

```
    @staticmethod
    def validar_elemento(solicitado, posibles):
        # Uso de "in" para validar que 1er argumento se encuentre en el 2do
        # Como mejora se pasa el primer argumento a minúscula
        return solicitado.lower() in posibles
```

Requerimiento 3

Archivo ingredientes.py (opcional)

1. (Opcional) Se define variables con valores posibles para cada ingrediente

```
vegetales = ["aceitunas", "tomate", "champiñones"]
proteicos = ["vacuno", "pollo", "carne vegetal"]
masas = ["tradicional", "delgada"]
```

Archivo pizza.py

2. (Opcional, en caso de haber realizado el paso anterior). Se agrega import de los ingredientes al comienzo del archivo (antes de definir la clase).

```
from ingredientes import proteicos, vegetales, masas
```

3. Definir método no estático para realizar pedido (recibe solo argumento “self”).

```
def pedir(self):
```

4. Almacenar en atributos de la clase la entrada del usuario para cada uno

```
def pedir(self):
    # Almacena proteico
    self.proteico = input(
        "\nIngrese qué proteína desea."
        "Las posibilidades son:\n- "
        "Vacuno\n- Pollo\n- Carne vegetal\n")

    # Define atributo vegetales como lista
    self.vegetales = []

    # Almacena primer vegetal
    self.vegetales.append(input(
        "\nIngrese el primer ingrediente vetegal.
        "Las opciones son:\n-Tomate\n-Aceitunas\n"
        "-Champiñones\n"))

    # Almacena segundo vegetal
    self.vegetales.append(input(
        "\nIngrese el segundo ingrediente"
        "vetegal.\n"))

    # Almacena tipo de masa
    self.tipo_de_masa = input(
```

```
            "\nIngrese tipo de masa."
            "Las posibilidades son:\n- Tradicional\n- Delgada\n")
```

5. (A continuación de línea anterior) Almacena en atributo no estático si es una pizza válida, aplicando método del requerimiento 2 para cada ingreso del usuario.

```
        self.es_una_pizza_valida = self.validar_elemento(self.proteico, proteicos) \
            and self.validar_elemento(self.vegetales[0], vegetales) \
            and self.validar_elemento(self.vegetales[1], vegetales) \
            and self.validar_elemento(self.tipo_de_masa, masas)
```

Requerimiento 4

Archivo evaluaciones.py

1. Importar clase Pizza del archivo `pizza.py`

```
from pizza import Pizza
```

2. Imprimir atributos de clase de la clase importada

```
print(f"Todas las pizzas tienen un tamaño {Pizza.tamanio} "
      f"y un valor de {Pizza.precio}")
```

3. Llamar a método de la clase importada para validar que “salsa de tomate” se encuentra en la lista [`"salsa de tomate", "salsa bbq"`]

```
print(Pizza.validar_elemento("salsa de tomate", ["salsa de tomate", "salsa bbq"]))
```

4. Crear objeto de tipo Pizza y llamar a método que realiza pedido

```
pizza = Pizza()
pizza.pedir()
```

5. Imprimir ingredientes del objeto creado (luego de los ingresos del usuario)

```
print(f"\nVegetales: {pizza.vegetales}\n"
      f"Proteína: {pizza.proteico}\n"
      f"Masa: {pizza.tipo_de_masa}\n"
      f"¿Es una pizza válida?: {pizza.es_una_masa_valdida}\n")
```

6. Imprimir valor de atributo no estático desde la clase sin instanciar

```
print(f"¿La clase es una pizza válida?: {Pizza.es_una_pizza_valida}"
```