La lista de pedidos en la clase Panaderia funciona como una cola. Debido a que los pedidos se agregan al final de la lista (en el método nuevo\_pedido) y se procesan desde el principio de la lista (en el método procesar\_pedido).

# La clase Pedido representa un pedido individual de un cliente

class Pedido:

    def **\_\_init\_\_**(self, cliente, panes):

        self.cliente = cliente  # El nombre del cliente que hizo el pedido

        self.panes = panes  # Un diccionario de los tipos de pan y sus cantidades

        self.completado = False  # Un indicador de si el pedido ha sido completado o no

    # Esta función se llama cuando se imprime un objeto Pedido

    def **\_\_str\_\_**(self):

        estado = 'Completado' if self.completado else 'Pendiente'

        return f"Cliente: {self.cliente}, Panes: {self.panes}, Estado: {estado}"

# La clase Panaderia representa la panadería y gestiona los pedidos

class Panaderia:

    def **\_\_init\_\_**(self, productos):

        self.pedidos = []  # Una lista para almacenar los pedidos

        self.productos = productos  # Una lista de los productos disponibles en la panadería

    # Este método permite agregar un nuevo pedido a la lista de pedidos

    def **nuevo\_pedido**(self, cliente, panes):

        pedido = Pedido(cliente, panes)

        self.pedidos.**append**(pedido)

    # Este método procesa el primer pedido pendiente en la lista de pedidos

    def **procesar\_pedido**(self):

        if not self.pedidos:

**print**("No hay pedidos pendientes.")

            return

        for pedido in self.pedidos:

            if not pedido.completado:

                pedido.completado = True

**print**(f"Pedido de {pedido.cliente} completado.")

                return

**print**("Todos los pedidos han sido completados.")

    # Este método muestra todos los pedidos en la lista de pedidos

    def **mostrar\_pedidos**(self):

        if not self.pedidos:

**print**("No hay pedidos para mostrar.")

            return

        for pedido in self.pedidos:

**print**(pedido)

    # Este método muestra todos los productos disponibles en la panadería

    def **mostrar\_productos**(self):

        for producto in self.productos:

**print**(producto)

# Crea una instancia de Panaderia con algunos productos

panaderia = Panaderia(["baguette", "pan campesino", "pan de queso", "pan de ajo", "pan de higo", "pan de banano"])

**print**("Bienvenido a La panadería 'El Pan Nuestro'")

# Un bucle para el menú interactivo

while True:

**print**("\n1. Ver productos")

**print**("2. Hacer un pedido")

**print**("3. Procesar un pedido")

**print**("4. Ver pedidos")

**print**("5. Salir")

    opcion = **input**("Elige una opción: ")

    if opcion == "1":

        panaderia.**mostrar\_productos**()

    elif opcion == "2":

        cliente = **input**("Ingresa el nombre del cliente: ")

        panes = {}

        while True:

            pan = **input**("Ingresa el tipo de pan (o 'salir' para terminar): ")

            if pan == "salir":

                break

            cantidad = int(**input**("Ingresa la cantidad de este pan: "))

            if pan in panes:

                panes[pan] += cantidad

            else:

                panes[pan] = cantidad

        panaderia.**nuevo\_pedido**(cliente, panes)

    elif opcion == "3":

        panaderia.**procesar\_pedido**()

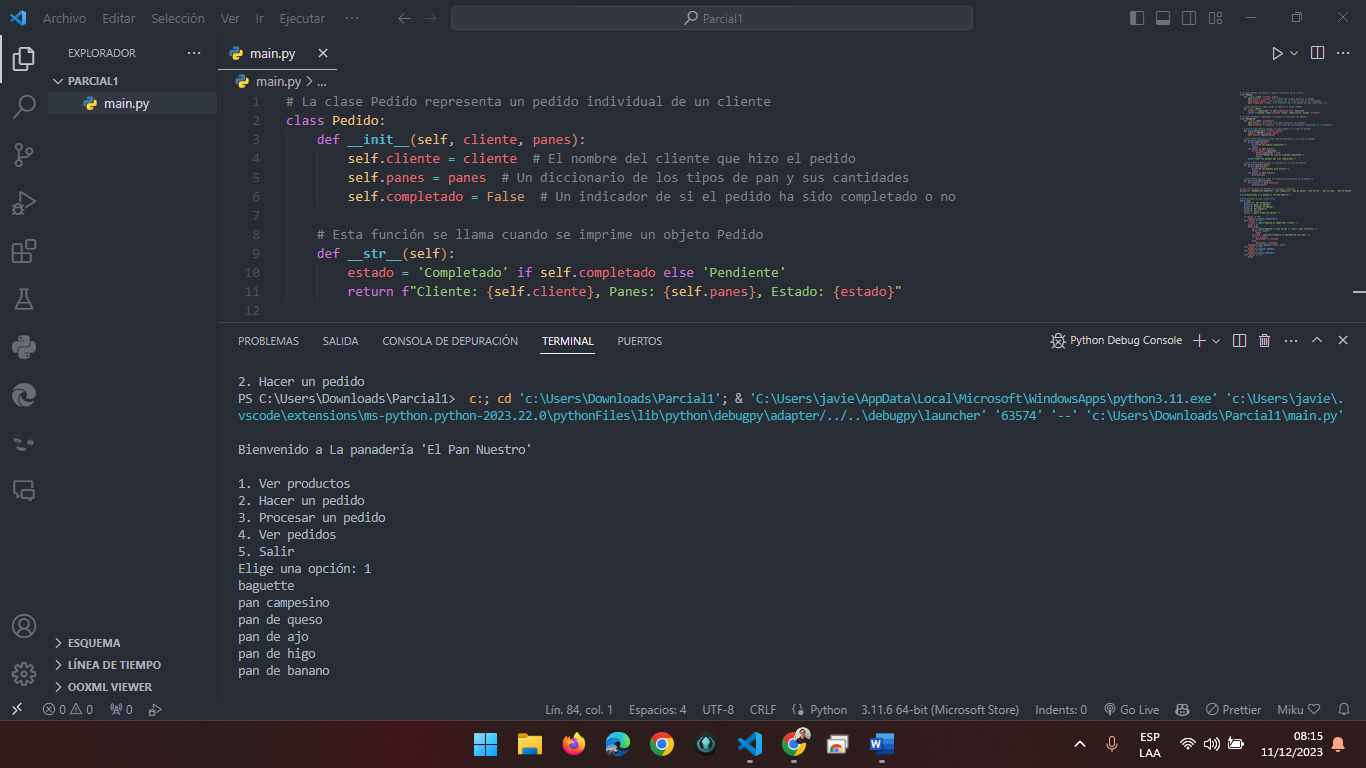
    elif opcion == "4":

        panaderia.**mostrar\_pedidos**()

    elif opcion == "5":

        break

Muestra de Funcionamiento



Menú con selección 1

Texto

Descripción generada automáticamente

Test1: Se ingreso un usuario y se hizo un pedido

Texto

Descripción generada automáticamente

Test2: Se ingreso un nuevo usuario y se hizo otro pedido

Texto

Descripción generada automáticamente

Procesamiento del primer pedido

Texto

Descripción generada automáticamente

Ver pedidos Pendientes-Completados

Texto

Descripción generada automáticamente

Salir

Texto

Descripción generada automáticamente