MANUAL DE USO DE CLASES

Proyecto Fast Food

Luis David Ixquiac Sac
Miguel Antonio Salguero Sandoval
José Andrés López Diaz

[Dirección de correo electrónico]

Clase menú:

1. En la clase menú contamos con un método constructor donde esta la pila que va a guardar la información de los menús que se ingresaran y que están como predeterminados.
2. Luego tenemos el método para ingresar un menú que nos pide el nombre, el complemento y el precio, luego esto es ingresado a un diccionario, seguido a eso se ingresa a la pila, y se envía el mensaje de que ha sido ingresado.
3. Luego tenemos el método que muestra los menús, en el cual utilizamos un ciclo for que recorre la pila y luego imprime los menús que tenemos para el día, en donde nos muestra el nombre, complemento y precio.
4. Luego tenemos el método para eliminar menús nos pide el índice en el cual elegimos la posición en la cual queremos eliminar el menú, en el cual con un if le decimos desde donde necesitamos, utilizando la función len tenemos el largo de la pila y utilizando la función .pop podemos quitarlo de la pila.
5. Luego tenemos el método que muestra los combos predeterminados que aparecen al iniciar el programa, estos están unidos utilizando la función .extend para añadirlos, utilizamos una opción en el menú principal para poder iniciarlos.

```python
class Menus:
    def __init__(self):
        self.pila = []

    def ingresar_menu(self, nombre, complementos, precio):
        menu = {"Nombre: ": nombre, "Complementos: ": complementos, "Precio: ": precio}
        self.pila.append(menu)
        print("El menu ha sido ingresado correctamente!")

    def mostrar_menu(self):
        print("Estos son los menús para el dia de hoy: ")
        for menu in self.pila:
            print(f'- {menu["Nombre"]}: {menu["Complementos"]} - Q.{menu["Precio"]}')

    def eliminar_menu(self, indice):
        if 1 <= indice <= len(self.pila):
            eliminar = self.pila.pop(indice-1)
            print(f'El menu "{eliminar["Nombre"]}" ha sido eliminado.')
        else:
            print("Indice no válido. No se eliminó ningún menú.")

    def menus_iniciales(self):
        menu1 = {"Nombre": "Combo #1", "Complementos": "Hamburguesa, papas fritas, refresco", "Precio": 50}
        menu2 = {"Nombre": "Combo #2", "Complementos": "Pizza, ensalada, agua", "Precio": 40}
        self.pila.extend([menu1, menu2])
        print("Se han añadido los menús predeterminados.")
```

6. Por último en esta clase tenemos un método que inicializa el menú en la cual están las opciones para ingresar un menú, mostrarlo, eliminarlo, y salir del programa.

```python
def inicializar(self):
    menu = Menus()
    menu.menus_iniciales()

    while True:
        print("1. Ingresar menu")
        print("2. Mostrar menu")
        print("3. Eliminar Menu")
        print("4. Salir")
        opcion = input("Seleccione una opcion: ")
        match opcion:
            case '1':
                nombre = input("Ingrese el nombre del combo: ")
                complementos = input("Ingrese el complemento del menu: ")
                precio = int(input("Ingrese el precio del menu: "))
                menu.ingresar_menu(nombre, complementos, precio)
            case '2':
                menu.mostrar_menu()
            case '3':
                indice = int(input("Ingrese el menu que desea eliminar: "))
                menu.eliminar_menu(indice)
            case '4':
                print("Saliendo de las opciones del menu")
                break
            case _:
                print("No ingreso una opcion valida")
```

Clase Pedidos:

1. En la clase pedidos contamos con un método constructor que tiene 3 colas en las cuales ingresan los pedidos del menú, utilizando 3 colas, en la primera ingresan los pedidos en estado pendiente, en la segunda los que están en preparación, y la última con pedido entregado.
2. Luego tenemos el método de ingreso_pedidos el cual le pide al cliente el combo que quiere del menú y nombre de quien estará luego en un diccionario se ingresan los datos y utilizando la función. put se ingresa a la cola pendiente.
3. Luego tenemos el método en_preparacion el cual tiene la función de enviar el pedido del estado de pendiente a preparación.
4. Luego tenemos el método listo_servir el cual tiene la misma función de enviar el pedido del estado en preparación a listos para servir.
5. Luego tenemos la función para mostrar los pedidos en la lista el cual funciona de modo que el ciclo for recorre la cola y busca los pedidos en la cola, el funcionamiento es el mismo en las 3 colas, luego los imprime con el nombre y el combo que el cliente eligió.

```python
class Pedidos:
    def __init__(self):
        self.pendiente = queue.Queue()
        self.preparacion = queue.Queue()
        self.entregado = queue.Queue()

    def ingreso_pedido(self, combo, nombre):
        pedido = {'Combo': combo, 'Nombre': nombre}
        self.pendiente.put(pedido)
        print(f'El pedido ha sido ingresado a la cola de pendientes')

    def en_preparacion(self):
        if not self.pendiente.empty():
            pedido = self.pendiente.get()
            self.preparacion.put(pedido)
            print(f'El pedido ha sido movido a la cola de preparación')

    def listo_servir(self):
        if not self.preparacion.empty():
            pedido = self.preparacion.get()
            self.entregado.put(pedido)
            print(f'El pedido ha sido entregado')

    def mostrar_pedidos(self):
        print('Pedidos Ingresados: ')
        for pedido in list(self.pendiente.queue):
            print(f'- Pedido: {pedido["Combo"]}, Nombre: {pedido["Nombre"]}')

        print('Pedidos en preparación: ')
        for pedido in list(self.preparacion.queue):
            print(f'- Pedido: {pedido["Combo"]}, Nombre: {pedido["Nombre"]}')

        print('Pedidos entregados: ')
        for pedido in list(self.entregado.queue):
            print(f'- Pedido: {pedido["Combo"]}, Nombre: {pedido["Nombre"]}')
```

6. Luego tenemos la clase para buscar el pedido por nombre utilizando el ordenamiento secuencial, en el cual ingresamos el nombre del cliente al inicio y utilizando la llave del diccionario busca en la cola el nombre del cliente y muestra la información acerca del pedido.

```python
# funcion utilizada en con busqueda secuencial, lo que hace es buscar por el nombre de la persona el pedido que tiene, cambio realizado en clase
def buscar_pedido(self, nombre):
    encontrado = False
    for pedido in list(self.pendiente.queue):
        if pedido['Nombre'] == nombre:
            print(f'Pedido encontrado en la cola de pendientes - Combo: {pedido["Combo"]}, Nombre: {pedido["Nombre"]}')
            encontrado = True
            break
    for pedido in list(self.preparacion.queue):
        if pedido['Nombre'] == nombre:
            print(f'Pedido encontrado en la cola de preparacion - Combo: {pedido["Combo"]}, Nombre: {pedido["Nombre"]}')
            encontrado = True
            break
    for pedido in list(self.entregado.queue):
        if pedido['Nombre'] == nombre:
            print(f'Pedido encontrado en la cola de entregados - Combo: {pedido["Combo"]}, Nombre: {pedido["Nombre"]}')
            encontrado = True
            break
    if not encontrado:
        print(f'Pedido no encontrado para el nombre: {nombre}')
```

7. Por último, tenemos el método para inicializar el menú en el cual tenemos las opciones para ingresar el pedido, la orden en preparación, la orden lista, ver los pedidos y buscar los pedidos por nombre y salir del menú.

```python
def inicializar_pedidos(self):

    objeto = Pedidos()

    while True:
        print('1. Ingreso de pedido')
        print('2. Orden en preparación')
        print('3. Orden a listos para entregar')
        print('4. Ver pedidos')
        print('5. Buscar pedido por nombre')
        print('6. Salir')

        opcion = input('Ingresa la opción que deseas: ')

        match opcion:
            case '1':
                combo = input('Ingresa el combo que quieres: ')
                nombre = input('Ingresa tu nombre para la entrega: ')
                objeto.ingreso_pedido(combo, nombre)
            case '2':
                objeto.en_preparacion()
            case '3':
                objeto.listo_servir()
            case '4':
                objeto.mostrar_pedidos()
            case '5':
                nombre_buscar = input('Ingrese el nombre para buscar el pedido: ')
                objeto.buscar_pedido(nombre_buscar)
            case '6':
                print('saliendo...')
                break
            case _:
                print('No ingresaste una opción válida')
```

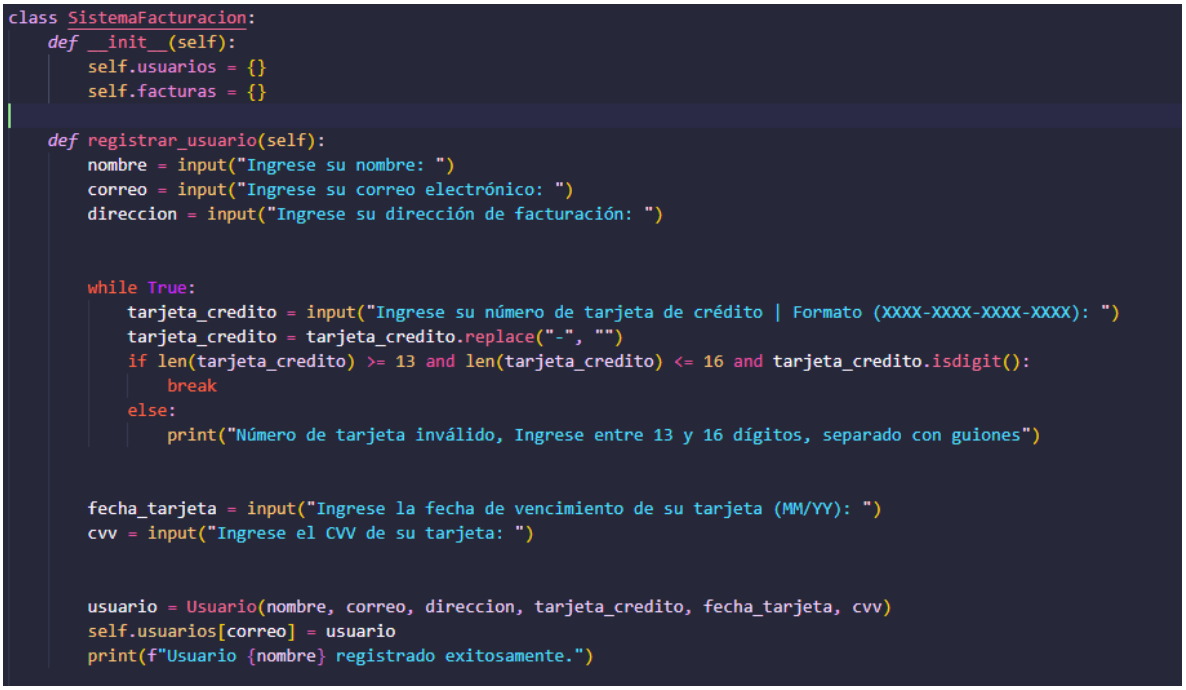
Clase Usuario:

1. Este método es el constructor de la clase usuario este recibe como parámetros la información básica de un usuario, como nombre, correo, dirección, número de tarjeta de crédito, fecha de vencimiento de la tarjeta y el CVV. Estos atributos se asignan a las variables de instancia correspondientes.

```python
class Usuario:
    def __init__(self, nombre, correo, direccion, tarjeta_credito, fecha_tarjeta, cvv):
        self.nombre = nombre
        self.correo = correo
        self.direccion = direccion
        self.tarjeta_credito = tarjeta_credito
        self.fecha_tarjeta = fecha_tarjeta
        self.cvv = cvv
```

Clase SistemaDeFacturacion:

1. El método constructor de la clase SistemaFacturacion inicializa dos diccionarios vacíos, usuarios y facturas, que se utilizan para almacenar información de usuarios y facturas para posteriormente facturar.
2. Este método permite al usuario del sistema ingresar su información, incluyendo nombre, correo, dirección y detalles de la tarjeta de crédito realiza validaciones para garantizar que la información ingresada sea correcta antes de crear una instancia de la clase Usuario y agregarla al diccionario de usuarios.
3. Genera una factura para un usuario específico. Verifica si el usuario existe en el diccionario de usuarios, y si es así, crea una factura con información como nombre, correo, dirección, monto total y una versión oculta del número de tarjeta. Luego, guarda esta información en el diccionario de facturas y también crea un archivo de texto con la información de la factura.
4. Este método es el punto de entrada principal del sistema. Muestra un menú con opciones como registrar usuario, generar factura o salir. Dependiendo de la opción seleccionada por el usuario, llama al método correspondiente.



```python
class SistemaFacturacion:
    def __init__(self):
        self.usuarios = {}
        self.facturas = {}

    def registrar_usuario(self):
        nombre = input("Ingrese su nombre: ")
        correo = input("Ingrese su correo electrónico: ")
        direccion = input("Ingrese su dirección de facturación: ")


        while True:
            tarjeta_credito = input("Ingrese su número de tarjeta de crédito | Formato (XXXX-XXXX-XXXX-XXXX): ")
            tarjeta_credito = tarjeta_credito.replace("-", "")
            if len(tarjeta_credito) >= 13 and len(tarjeta_credito) <= 16 and tarjeta_credito.isdigit():
                break
            else:
                print("Número de tarjeta inválido, Ingrese entre 13 y 16 dígitos, separado con guiones")


        fecha_tarjeta = input("Ingrese la fecha de vencimiento de su tarjeta (MM/YY): ")
        cvv = input("Ingrese el CVV de su tarjeta: ")


        usuario = Usuario(nombre, correo, direccion, tarjeta_credito, fecha_tarjeta, cvv)
        self.usuarios[correo] = usuario
        print(f"Usuario {nombre} registrado exitosamente.")
```

```python
def generar_factura(self, correo_usuario, total):
    if correo_usuario in self.usuarios:
        usuario = self.usuarios[correo_usuario]
        numero_factura = len(self.facturas) + 1


        tarjeta_oculta = "X" * (len(usuario.tarjeta_credito) - 4) + usuario.tarjeta_credito[-4:]

        self.facturas[numero_factura] = {
            'cliente': usuario.nombre,
            'total': total,
            'direccion': usuario.direccion,
            'correo': usuario.correo,
            'tarjeta_credito': tarjeta_oculta,
        }

        nombre_archivo = input("Ingrese el nombre para el archivo de texto (sin extensión): ")
        nombre_archivo_txt = f'{nombre_archivo}.txt'

        contenido = f"Restaurante: Fast Food - Proyecto\n"
        contenido += f"Cliente: {usuario.nombre}\n"
        contenido += f"Correo: {usuario.correo}\n"
        contenido += f"Dirección: {usuario.direccion}\n"
        contenido += f"Total: {total}\n"
        contenido += f"Tarjeta de Crédito: XXXX-XXXX-XXXX-{usuario.tarjeta_credito[-4:]}\n"

        with open(nombre_archivo_txt, 'w') as archivo_txt:
            archivo_txt.write(contenido)

        print(f"Factura generada para el usuario {usuario.nombre}, número de factura: {numero_factura}")
        print(f"El archivo de texto '{nombre_archivo_txt}' ha sido guardado.")
    else:
        print("---- Usuario no encontrado ----")

def inicializar_factura(self):
    sistema_facturacion = SistemaFacturacion()

    while True:
        print("\n---- Menú ----")
        print("1. Registrar Usuario")
        print("2. Generar Factura")
        print("3. Salir")

        opcion = input("Seleccione una opción: ")

        if opcion == "1":
            sistema_facturacion.registrar_usuario()
        elif opcion == "2":
            correo_usuario = input("Ingrese el correo electrónico del usuario: ")
            total = float(input("Ingrese el monto total de la factura: "))
            sistema_facturacion.generar_factura(correo_usuario, total)
        elif opcion == "3":
            print("------- FIN DE LA EJECUCION --------")
            break
        else:
            print("Opción no válida, seleccione una opción válida del menú.")
```