UVG UNIVERSIDAD DEL VALLE DE GUATEMALA

Guatemala 9 de noviembre de 2023
David Miranda - 231260

**Lab #4**

Polimorfismo a través de interfaces

---

1. Requisitos Funcionales:
    a. Registro de usuario: el programa debe permitir a los usuarios crear una cuenta ingresando un nombre de usuario y una contraseña. Además, deben poder seleccionar entre dos planes: uno gratuito y otro de pago (VIP).
    b. Ingreso/Salida: el programa debe permitir a los usuarios realizar reservas de vuelos. Deben poder seleccionar la fecha de viaje, el tipo de vuelo (ida y vuelta o solo ida), la cantidad de boletos y la aerolínea.
    c. Confirmación: el programa debe permitir a los usuarios confirmar su reserva ingresando el número de su tarjeta de crédito y la cantidad de cuotas en las que desean pagar.
2. Análisis:
    a. - Identificación de Clases, atributos y métodos
        i. Clases:
            1. Interfaz “ModoInterfaz”: Interface que define métodos para interactuar con el sistema, como iniciar sesión, registrar usuarios, gestionar reservas, entre otros.
            2. Usuario: Representa a un usuario del sistema.
            3. CSVManager: Gestiona la lectura y escritura de archivos CSV.
            4. MenuPrincipal: Clase que contiene el método principal (main) para ejecutar el programa.
            5. ReservaPremium: Clase que extiende la clase Reserva y agrega una columna adicional.
            6. Reserva: Clase que representa una reserva en el sistema.
            7. Confirmacion: Representa la confirmación de un vuelo con atributos asociados.
            8. ImplementacionModoInterfaz: Clase que implementa la interfaz ModoInterfaz y gestiona la lógica del sistema.
    b. - Atributos de las clases
        i. Interfaz “ModoInterfaz”:

1. No hay atributos en la interfaz, solo métodos que representan acciones.

ii. Clase Usuario:
    1. username (String): Propósito: Almacena el nombre de usuario del usuario.
    2. password (String): Propósito: Almacena la contraseña del usuario.
    3. tipo (String): Propósito: Almacena el tipo de usuario (gratis, VIP).

iii. Clase CSVManager: no tiene atributos.

iv. Clase MenuPrincipal: no tiene atributos.

v. Clase ReservaPremium: otraColumna (String): Propósito: Almacena información adicional específica de la reserva premium.

vi. Clase Reserva:
    1. nombreUsuario (String): Propósito: Almacena el nombre de usuario asociado a la reserva.
    2. fecha (String): Propósito: Almacena la fecha de la reserva.
    3. tipoVuelo (boolean): Propósito: Indica si el vuelo es de ida y vuelta (true) o solo ida (false).
    4. cantidadBoletos (int): Propósito: Almacena la cantidad de boletos reservados.
    5. aerolinea (String): Propósito: Almacena el nombre de la aerolínea.
    6. tipoVueloString (String):Propósito: Almacena una representación en cadena del tipo de vuelo (ida/vuelta o ida).

vii. Clase Confirmacion:
    1. numeroTarjeta (String): Número de tarjeta asociado a la confirmación.
    2. cuotas (int): Número de cuotas para el pago.
    3. claseVuelo (String): Clase del vuelo (por ejemplo, primera clase, económica).
    4. numeroAsiento (String): Número de asiento asignado.
    5. cantidadMaletas (int): Cantidad de maletas para el vuelo.

viii. Clase ImplementacionModoInterfaz:
    1. usuarios (List<Usuario>): Lista de usuarios en el sistema.
    2. reservas (List<Reserva>): Lista de reservas en el sistema.
    3. confirmaciones (List<Confirmacion>): Lista de confirmaciones en el sistema.

c. Métodos de las clases:

i. Interfaz “ModoInterfaz”:
    1. login(String username, String password): Usuario
        a. Propósito: Iniciar sesión y devolver un objeto Usuario si las credenciales son correctas.
    2. buscarUsuarioPorUsername(String username): Usuario
        a. Propósito: Buscar un usuario por nombre de usuario y devolverlo.
    3. registroUsuario(String username, String password, String tipo): void
        a. Propósito: Registrar un nuevo usuario con nombre de usuario, contraseña y tipo (gratis, VIP).
    4. cambiarPassword(String nuevaPassword): void
        a. Propósito: Cambiar la contraseña del usuario.
    5. cambiarTipoUsuario(): void
        a. Propósito: Cambiar el tipo de usuario, recibir el nuevo tipo como parámetro.
    6. reservacion(String fechaVuelo, boolean tipoVuelo, int cantidadBoletos, String aerolinea, String username): void
        a. Propósito: Realizar una reserva con la información proporcionada.
    7. confirmacion(String numeroTarjeta, int cuotas, String claseVuelo, String numeroAsiento, int cantidadMaletas, String nombreUsuario): void
        a. Propósito: Confirmar un vuelo con la información proporcionada.
    8. itinerario(): String
        a. Propósito: Generar y devolver el itinerario del usuario.
    9. guardarUsuario(): void
        a. Propósito: Guardar la información del usuario en algún formato (CSV).
    10. guardarReserva(): void
        a. Propósito: Guardar la información de la reserva en algún formato (CSV).
    11. leerUsuario(): void
        a. Propósito: Leer la información de los usuarios, desde un archivo.
    12. leerReserva(): void
        a. Propósito: Leer la información de las reservas, desde un archivo.

13. guardarConfirmacion(Confirmacion confirmacion, String nombreUsuario): void
    a. Propósito: Guardar la confirmación de vuelo, en algún formato (CSV).
14. modoPerfilNoPremium(Usuario usuario, Scanner scanner): void
    a. Propósito: Manejar el modo perfil para usuarios no premium.
15. modoPerfilPremium(Usuario usuario, Scanner scanner): void
    a. Propósito: Manejar el modo perfil para usuarios premium.

ii. Métodos de Usuario:
1. getTipo(): String; Propósito: Devuelve el tipo de usuario.
2. getUsername(): String; Propósito: Devuelve el nombre de usuario.
3. setUsername(String username): void; Propósito: Establece el nombre de usuario.
4. getPassword(): String; Propósito: Devuelve la contraseña.
5. setPassword(String password): void; Propósito: Establece la contraseña.
6. setTipo(String tipo): void; Propósito: Establece el tipo de usuario.
7. cambiarTipoUsuario(String nuevoTipoCliente): void; Propósito: Cambia el tipo de usuario.

iii. Métodos de CSVManager:
1. escribirCSV(String nombreArchivo, String contenido): void; Propósito: Escribe el contenido en un archivo CSV con el nombre proporcionado.
2. leerCSV(String nombreArchivo): String; Propósito: Lee el contenido de un archivo CSV y lo devuelve como una cadena.
3. actualizarTipoUsuarioEnCSV(String archivoCSV, String username, String nuevoTipo): void; Propósito: Actualiza el tipo de usuario en un archivo CSV.

iv. Métodos de MenuPrincipal:
1. main(String[] args): void; Propósito: Método principal que maneja el flujo del programa. Realiza acciones como la creación de usuarios, login, y menús para usuarios premium y no premium.

2. menuPremium(ImplementacionModoInterfaz modo, Usuario usuario, Scanner scanner): void; Propósito: Menú para usuarios premium que incluye opciones relacionadas con reservas, confirmación y perfil premium.
3. menuNoPremium(ImplementacionModoInterfaz modo, Usuario usuario, Scanner scanner): void; Propósito: Menú para usuarios no premium que incluye opciones relacionadas con reservas, confirmación y perfil no premium.
4. modoReservasPremium(ImplementacionModoInterfaz modo, Usuario usuario, Scanner scanner): void; Propósito: Método que maneja el modo de reservas para usuarios premium, recopilando información y creando la reserva.
5. modoConfirmacionPremium(ImplementacionModoInterfaz modo, Usuario usuario, Scanner scanner): void; Propósito: Método que maneja el modo de confirmación para usuarios premium, recopilando información y creando la confirmación.
6. modoPerfilPremium(ImplementacionModoInterfaz modo, Usuario usuario, Scanner scanner): void; Propósito: Método que maneja el modo de perfil para usuarios premium.
7. modoReservasNoPremium(ImplementacionModoInterfaz modo, Usuario usuario, Scanner scanner, int cuotas): void; Propósito: Método que maneja el modo de reservas para usuarios no premium, recopilando información y creando la reserva.
8. modoConfirmacionNoPremium(ImplementacionModoInterfaz modo, Usuario usuario, Scanner scanner, int cuotas): void; Propósito: Método que maneja el modo de confirmación para usuarios no premium, recopilando información y creando la confirmación.
9. aplicarDescuento(int cuotas): double; Propósito: Método que aplica descuento según el número de cuotas y devuelve el factor de descuento.

v. Métodos de ReservaPremium:
    1. Constructor ReservaPremium(String nombreUsuario, String fecha, boolean tipoVuelo, int cantidadBoletos, String aerolinea, String otraColumna): void; Propósito: Constructor para la clase ReservaPremium que inicializa sus atributos.

2. getOtraColumna(): String; Propósito: Devuelve el valor de la columna adicional.
3. setOtraColumna(String otraColumna): void; Propósito: Establece el valor de la columna adicional.

vi. Clase Reserva:

1. Constructor Reserva(): void; Propósito: Constructor vacío de la clase Reserva.
2. Constructor Reserva(String nombreUsuario, String fecha, boolean tipoVuelo, int cantidadBoletos, String aerolinea): void; Propósito: Constructor que inicializa los atributos de la clase Reserva.
3. Constructor Reserva(String fechaVuelo, boolean tipoVuelo2, int cantidadBoletos2, String aerolinea2, String username): void; Propósito: Constructor alternativo para la clase Reserva.
4. getTipoVueloString(): String; Propósito: Devuelve la representación en cadena del tipo de vuelo.
5. setTipoVueloSring(String tipoVueloString): void; Propósito: Establece la representación en cadena del tipo de vuelo.
6. getNombreUsuario(): String; Propósito: Devuelve el nombre de usuario asociado a la reserva.
7. getFecha(): String; Propósito: Devuelve la fecha de la reserva.
8. setFecha(String fecha): void; Propósito: Establece la fecha de la reserva.
9. isTipoVuelo(): boolean; Propósito: Devuelve true si el vuelo es de ida y vuelta, false si es solo ida.
10. setTipoVuelo(boolean tipoVuelo): void; Propósito: Establece el tipo de vuelo.
11. getCantidadBoletos(): int; Propósito: Devuelve la cantidad de boletos reservados.
12. setCantidadBoletos(int cantidadBoletos): void; Propósito: Establece la cantidad de boletos reservados.
13. getAerolinea(): String; Propósito: Devuelve el nombre de la aerolínea.
14. setAerolinea(String aerolinea): void; Propósito: Establece el nombre de la aerolínea.

vii. Métodos de Confirmacion:

1. Confirmacion(String numeroTarjeta, int cuotas, String claseVuelo, String numeroAsiento, int cantidadMaletas): Constructor que inicializa los atributos.

2. Métodos getters y setters para cada atributo.

viii. Métodos de ImplementacionModoInterfaz:

1. login(String username, String password): Usuario;
    a. Propósito: Iniciar sesión y devolver un objeto Usuario si las credenciales son correctas.
    b. Parámetros:
        i. username (String): Nombre de usuario proporcionado.
        ii. password (String): Contraseña proporcionada.
        iii. Devuelve: Un objeto Usuario si las credenciales son correctas, o null si no se encuentra el usuario.
2. buscarUsuarioPorUsername(String username): Usuario
    a. Propósito: Buscar un usuario por nombre de usuario y devolverlo.
    b. Parámetros:
        i. username (String): Nombre de usuario a buscar.
        ii. Devuelve: Un objeto Usuario si se encuentra, o null si no.
3. registroUsuario(String username, String password, String tipo): void
    a. Propósito: Registrar un nuevo usuario con nombre de usuario, contraseña y tipo (gratis, VIP).
    b. Parámetros:
        i. username (String): Nombre de usuario del nuevo usuario.
        ii. password (String): Contraseña del nuevo usuario.
        iii. tipo (String): Tipo de usuario a asignar (gratis, VIP).
4. cambiarPassword(String nuevaPassword): void
    a. Propósito: Cambiar la contraseña del usuario.
    b. Parámetros:
        i. nuevaPassword (String): Nueva contraseña a establecer.
5. cambiarTipoUsuario(): void
    a. Propósito: Cambiar el tipo de usuario, recibir el nuevo tipo como parámetro.

6. reservacion(String fechaVuelo, boolean tipoVuelo, int cantidadBoletos, String aerolinea, String username): void
    a. Propósito: Realizar una reserva con la información proporcionada.
    b. Parámetros:
        i. fechaVuelo (String): Fecha del vuelo a reservar.
        ii. tipoVuelo (boolean): Tipo de vuelo (ida y vuelta o solo ida).
        iii. cantidadBoletos (int): Cantidad de boletos a reservar.
        iv. aerolinea (String): Nombre de la aerolínea.
        v. username (String): Nombre de usuario asociado a la reserva.
7. confirmacion(String numeroTarjeta, int cuotas, String claseVuelo, String numeroAsiento, int cantidadMaletas, String nombreUsuario): void
    a. Propósito: Confirmar un vuelo con la información proporcionada.
    b. Parámetros:
        i. numeroTarjeta (String): Número de tarjeta para la confirmación.
        ii. cuotas (int): Número de cuotas para el pago.
        iii. claseVuelo (String): Clase del vuelo.
        iv. numeroAsiento (String): Número de asiento asignado.
        v. cantidadMaletas (int): Cantidad de maletas para el vuelo.
        vi. nombreUsuario (String): Nombre de usuario asociado a la confirmación.
8. guardarConfirmacion(Confirmacion confirmacion, String nombreUsuario): void
    a. Propósito: Guardar la confirmación de vuelo, en algún formato (CSV).
    b. Parámetros:
        i. confirmacion (Confirmacion): Objeto Confirmacion a guardar.
        ii. nombreUsuario (String): Nombre de usuario asociado a la confirmación.
9. itinerario(): String

a. Propósito: Generar y devolver el itinerario del usuario.
b. Devuelve: Una cadena que representa el itinerario del usuario.

10. guardarUsuario(): void
    a. Propósito: Guardar la información del usuario en algún formato (CSV).
11. guardarReserva(): void
    a. Propósito: Guardar la información de la reserva en algún formato (CSV).
12. leerUsuario(): void
    a. Propósito: Leer la información de los usuarios, desde un archivo.
13. leerReserva(): void
    a. Propósito: Leer la información de las reservas, desde un archivo.
14. modoPerfilNoPremium(Usuario usuario, Scanner scanner): void
    a. Propósito: Manejar el modo perfil para usuarios no premium.
15. modoPerfilPremium(Usuario usuario, Scanner scanner): void
    a. Propósito: Manejar el modo perfil para usuarios premium.

*3.* Diseño: