#importar

import re

from datetime import datetime, timedelta

#precios

ESPACIOS\_TOTALES = 64

PRECIO\_HORA = 7000

PRECIO\_CUARTO = 1500

#listas

usuarios = []

vehiculos\_registrados = {}

vehiculos\_retirados = []

administradores = {"admin": "1234"}

#nombre del cliente validando restricciones

def validar\_nombre(nombre):

    errores = []

    if len(nombre) < 3:

        errores.append("El nombre debe tener al menos 3 letras.")

    contiene\_numero = False

    for char in nombre:

        try:

            int(char)

            contiene\_numero = True

            break

        except ValueError:

            continue

    if contiene\_numero:

        errores.append("El nombre no debe contener números.")

    return errores

#apellido del cliente validando restricciones

def validar\_apellido(apellido):

    errores = []

    if len(apellido) < 3:

        errores.append("El apellido debe tener al menos 3 letras.")

    contiene\_numero = False

    for char in apellido:

        try:

            int(char)  # Intentamos convertir el carácter a número

            contiene\_numero = True

            break      # Si lo logramos, hay un número, salimos del ciclo

        except ValueError:

            continue   # Si no es un número, seguimos con el siguiente carácter

    if contiene\_numero:

        errores.append("El apellido no debe contener números.")

    return errores

#cedula del cliente validando restricciones

def validar\_documento(documento):

    errores = []

    if not documento.isdigit():

        errores.append("El documento solo debe contener números.")

    if not 3 <= len(documento) <= 15:

        errores.append("El documento debe tener entre 3 y 15 dígitos.")

    return errores

#placa del vehiculo validando restricciones

def validar\_placa(placa):

    errores = []

    if len(placa) != 6:

        errores.append("La placa debe tener exactamente 6 caracteres.")

    elif not (placa[:3].isalpha() and placa[3:].isdigit()):

        errores.append("La placa debe tener 3 letras seguidas de 3 números.")

    return errores

#registro del vehiculo

def registrar\_vehiculo():

    if len(vehiculos\_registrados) >= ESPACIOS\_TOTALES:

        print("Parqueadero lleno. No hay espacios disponibles.")

        return

    nombre = input("Ingrese nombre: ")

    apellido = input("Ingrese apellido: ")

    documento = input("Ingrese documento: ")

    placa = input("Ingrese placa del vehículo: ").upper()

    errores = validar\_nombre(nombre) + validar\_apellido(apellido) + \

              validar\_documento(documento) + validar\_placa(placa)

    if errores:

        print("\nErrores en el ingreso de datos:")

        for error in errores:

            print("- {}".format(error))

        return

    hora\_entrada = datetime.now()

    vehiculos\_registrados[placa] = {

        "nombre": nombre,

        "apellido": apellido,

        "documento": documento,

        "hora\_entrada": hora\_entrada

    }

    usuarios.append({"nombre": nombre, "apellido": apellido, "documento": documento})

    print("\nVehículo ingresado exitosamente.\nRecibo:\nPlaca: {}\nHora de entrada: {}\n".format(placa, hora\_entrada.strftime('%H:%M') ))

#retiro del vehiculo

def retirar\_vehiculo():

    placa = input("Ingrese la placa del vehículo a retirar: ").upper()

    if placa not in vehiculos\_registrados:

        print("\nEl vehículo no está registrado.\n")

        return

    entrada = vehiculos\_registrados.pop(placa)

    salida = datetime.now()

    tiempo\_total = salida - entrada['hora\_entrada']

    horas = tiempo\_total.seconds // 3600

    cuartos = ((tiempo\_total.seconds % 3600) + 899) // 900  # redondeo hacia arriba

    total = horas \* PRECIO\_HORA + cuartos \* PRECIO\_CUARTO

    total = max(total, PRECIO\_HORA)

    vehiculos\_retirados.append({"placa": placa, "entrada": entrada['hora\_entrada'], "salida": salida, "total": total})

    print("\nFactura de retiro:\nPlaca: {}\nHora entrada: {}\nHora salida: {}\nTotal a pagar: ${}\n".format(

    placa,

    entrada['hora\_entrada'].strftime('%H:%M'),

    salida.strftime('%H:%M'),

    total))

#ingresar menu de administrador

def menu\_administrador():

    usuario = input("Ingrese usuario administrador: ")

    clave = input("Ingrese clave: ")

    if administradores.get(usuario) != clave:

        print("\nUsuario o clave incorrectos.\n")

        return

    while True:

        print("""

--- Reportes Administrador ---

1. Total de vehículos registrados

2. Total de vehículos retirados

3. Total de vehículos sin retirar

4. Total pagado por vehículos retirados

5. Tiempo promedio de parqueo

6. Lista de usuarios

7. Vehículo con tiempo máximo y mínimo

8. Salir

        """)

        opcion = input("Seleccione una opción: ")

        if opcion == "1":

           print("Total de vehículos registrados: {}".format(len(vehiculos\_registrados)))

        elif opcion == "2":

           print(f"Total de vehículos retirados: {len(vehiculos\_retirados)}")

        elif opcion == "3":

            print(f"Total de vehículos sin retirar: ${len(vehiculos\_registrados)}")

        elif opcion == "4":

            total = sum(v["total"] for v in vehiculos\_retirados)

            print("Total pagado por vehículos retirados: ${}".format(total))

        elif opcion == "5":

            if vehiculos\_retirados:

                promedio = sum((v["salida"] - v["entrada"]).seconds for v in vehiculos\_retirados) / len(vehiculos\_retirados)

                print("Tiempo promedio de parqueo: {} minutos".format(int(promedio // 60)))

            else:

                print("No hay datos de vehículos retirados.")

        elif opcion == "6":

            for u in usuarios:

                print("{} {} - Documento: {}".format(u['nombre'], u['apellido'], u['documento']))

        elif opcion == "7":

            if not vehiculos\_retirados:

                print("No hay vehículos retirados.")

                continue

            tiempos = [(v["placa"], (v["salida"] - v["entrada"]).seconds) for v in vehiculos\_retirados]

            max\_veh = max(tiempos, key=lambda x: x[1])

            min\_veh = min(tiempos, key=lambda x: x[1])

            print("Vehículo con mayor tiempo: {} - {} minutos".format(max\_veh[0],max\_veh[1] // 60))

            print("Vehículo con menor tiempo: {} - {} minutos".format(min\_veh[0], min\_veh[1] // 60))

        elif opcion == "8":

            break

        else:

            print("Opción inválida.")

#menu principal

def menu\_principal():

    while True:

        print("""

--- Parqueadero Nido de Ruedas ---

1. Ingresar vehículo

2. Retirar vehículo

3. Módulo administrador

4. Salir

        """)

        opcion = input("Seleccione una opción: ")

        if opcion == "1":

            registrar\_vehiculo()

        elif opcion == "2":

            retirar\_vehiculo()

        elif opcion == "3":

            menu\_administrador()

        elif opcion == "4":

            print("Gracias por usar el sistema del parqueadero.")

            break

        else:

            print("Opción inválida.")

#ejecutar

if \_\_name\_\_ == "\_\_main\_\_":

    menu\_principal()