Untitled

# Práctica 2. Clases y Objetos, métodos y atributos

class Persona: def **init**(self, nombre, apellido, edad): # Creación de la clase self.nombre = nombre self.apellido = apellido self.edad = edad self.\_\_cuenta = None # Atributo privado

def asignar\_cuenta(self, cuenta):  
 self.\_\_cuenta = cuenta  
 print(f"{self.nombre} ahora tiene una cuenta bancaria")  
   
def consultar\_saldo(self):  
 if self.\_\_cuenta:  
 print(f"El saldo de {self.nombre} es: ${self.\_\_cuenta.mostrar\_saldo()}")  
 else:  
 print(f"{self.nombre} no tiene una cuenta bancaria asignada.")  
  
# Nuevos métodos para operaciones bancarias  
def depositar\_dinero(self, cantidad):  
 if self.\_\_cuenta:  
 self.\_\_cuenta.depositar(cantidad)  
 else:  
 print(f"{self.nombre} no tiene una cuenta bancaria asignada.")  
  
def retirar\_dinero(self, cantidad):  
 if self.\_\_cuenta:  
 self.\_\_cuenta.retirar(cantidad)  
 else:  
 print(f"{self.nombre} no tiene una cuenta bancaria asignada.")  
   
def presentarse(self):  
 print(f"Hola, mi nombre es {self.nombre} {self.apellido} y tengo {self.edad} años.")  
   
def cumplir\_anios(self):  
 self.edad += 1  
 print(f"¡Feliz cumpleaños! {self.nombre} ahora tiene {self.edad} años.")

class CuentaBancaria: def **init**(self, numero\_cuenta, saldo): self.numero\_cuenta = numero\_cuenta self.\_\_saldo = saldo # Atributo privado

def mostrar\_saldo(self):  
 return self.\_\_saldo  
  
def depositar(self, cantidad):  
 if cantidad > 0:  
 self.\_\_saldo += cantidad  
 print(f"Se han depositado ${cantidad} a la cuenta. Nuevo saldo: ${self.\_\_saldo}")  
 else:  
 print("La cantidad a depositar debe ser positiva.")  
   
def retirar(self, cantidad):  
 if 0 < cantidad <= self.\_\_saldo:  
 self.\_\_saldo -= cantidad  
 print(f"Se han retirado ${cantidad} de la cuenta. Nuevo saldo: ${self.\_\_saldo}")  
 else:  
 print("Fondos insuficientes o cantidad inválida.")

# Creación de objetos

estudiante1 = Persona(“Juan”, “Perez”, 20) estudiante2 = Persona(“Maria”, “Gomez”, 22) cuenta1 = CuentaBancaria(“0001”, 1000)

# Operaciones corregidas

print(“=== OPERACIONES BANCARIAS ===”) estudiante1.presentarse() estudiante1.asignar\_cuenta(cuenta1) estudiante1.consultar\_saldo() estudiante1.depositar\_dinero(500) # Ahora funciona correctamente estudiante1.retirar\_dinero(200) # Ahora funciona correctamente estudiante1.consultar\_saldo()

print(“”) estudiante2.presentarse() estudiante2.consultar\_saldo() # Mostrará que no tiene cuenta

print(“”) estudiante1.cumplir\_anios() estudiante1.presentarse()

# EJERCICIO 1. Clase Coche corregida

print(“=== OPERACIONES CON COCHES ===”)

class Coche: def **init**(self, marca, modelo, anio): self.marca = marca self.modelo = modelo self.anio = anio self.velocidad = 0 # Inicializar velocidad aquí

def datos(self):  
 print(f"El coche es {self.marca} {self.modelo} del año {self.anio}.")  
   
def frenar(self, decremento):  
 if decremento > self.velocidad:  
 self.velocidad = 0  
 else:  
 self.velocidad -= decremento  
 print(f"El coche {self.marca} {self.modelo} ha frenado a {self.velocidad} km/h.")  
   
def acelerar(self, incremento):  
 if incremento > 0:  
 self.velocidad += incremento  
 print(f"El coche {self.marca} {self.modelo} ha acelerado a {self.velocidad} km/h.")  
 else:  
 print("El incremento debe ser positivo.")  
  
def estado\_velocidad(self):  
 print(f"Velocidad actual: {self.velocidad} km/h")

# Crear objetos Coche

automovil1 = Coche(“Toyota”, “Corolla”, 2020) automovil2 = Coche(“Honda”, “Civic”, 2019)

automovil1.datos() automovil2.datos()

print(“— Pruebas de velocidad —”) automovil1.acelerar(120) automovil1.frenar(70) automovil1.estado\_velocidad()

print(“”) automovil2.acelerar(150) automovil2.frenar(90) automovil2.estado\_velocidad()