UNIVERSIDAD TECMILENIO

Mauricio Estrada De la Garza
AL02976904
14/11/2023
Programación Orientada a Objetos
Reto 10

Reto 10: Implementación de los conceptos fundamentales de Programación en Python.

Objetivo de la actividad:

El alumno desarrollará un programa en el cual se implementa los conceptos:

1. **Uso correcto de variables y manipulación de datos.**
1. **Uso correcto de datos de entrada / salida (input/print).**
1. **Creación y uso correcto de colecciones.**
1. **Creación y uso correcto de Funciones.**
1. **Uso correcto de condicionales.**
1. **Uso correcto de ciclos.**
1. **Uso correcto de flujos de ejecución.**
1. **Documentación precisa y concreta dentro del programa.**

Requerimientos para la actividad:

Cualquier IDE que verifique la sintaxis de python.

Instrucciones para el alumno:

- **Crea una clase llamada Banco que represente una entidad bancaria. La clase debe tener los siguientes atributos:**
- **Nombre del banco**
- **Saldo total del banco (inicializado en 0)**
- **Define un constructor __init__ para la clase Banco que inicialice el nombre del banco.**
- **Agrega un método mostrar saldo a la clase Banco que imprima el nombre del banco y el saldo total.**

- **Crea una clase secundaria llamada Cuenta que represente una cuenta bancaria. La clase Cuenta debe tener los siguientes atributos:**

- **Nombre del titular de la cuenta**
- **Saldo de la cuenta**
- **Tipo de Cuenta (Ahorro, Cheques, Inversión, etc., se vale inventar)**
- **Define un constructor __init__ para la clase Cuenta que inicialice el nombre del titular de la cuenta y el saldo de la cuenta. Asegúrate de que no se puedan crear objetos de la clase Cuenta con un saldo inicial negativo.**
- **Agrega un método depositar a la clase Cuenta que permita depositar una cantidad de dinero en la cuenta. Actualiza el saldo de la cuenta.**
- **Agrega un método retirar a la clase Cuenta que permita retirar una cantidad de dinero de la cuenta. Asegúrate de que no sea posible retirar más dinero del disponible en la cuenta y maneja la excepción correspondiente.**

- **Crea una clase Cliente que represente a un cliente bancario. La clase Cliente debe tener los siguientes atributos:**
- **Nombre del cliente.**
- **Una lista de objetos de la clase Cuenta que el cliente posee.**
- **Define un constructor __init__ para la clase Cliente que inicialice el nombre del cliente y su lista de cuentas.**

- **Escribe un programa principal que:**
- **Crea una instancia de la clase Banco.**
- **Crea varias instancias de la clase Cuenta.**
- **Crea una instancia de la clase Cliente con una lista de cuentas.**
- **Realiza depósitos y retiros en las cuentas del cliente y muestra el saldo actual de cada cuenta.**
- **Muestra el saldo total del banco después de todas las transacciones.**

Conceptos a calificar:

1. **Uso correcto de variables y manipulación de datos.**
1. **Creación y uso correcto de clases y objetos.**
1. **Creación y uso correcto de métodos de clases.**
1. **Uso correcto de condicionales para manejar excepciones.**
1. **Uso correcto de flujos de ejecución.**
1. **Creación y definición de clases con atributos y métodos.**
1. **Uso de objetos y construcción de instancias de las clases Banco, Cuenta y Cliente.**

1. **Reflexión: Discutir sobre la flexibilidad de objetos y clases en python versus otros lenguajes de programación.**

Entregables:

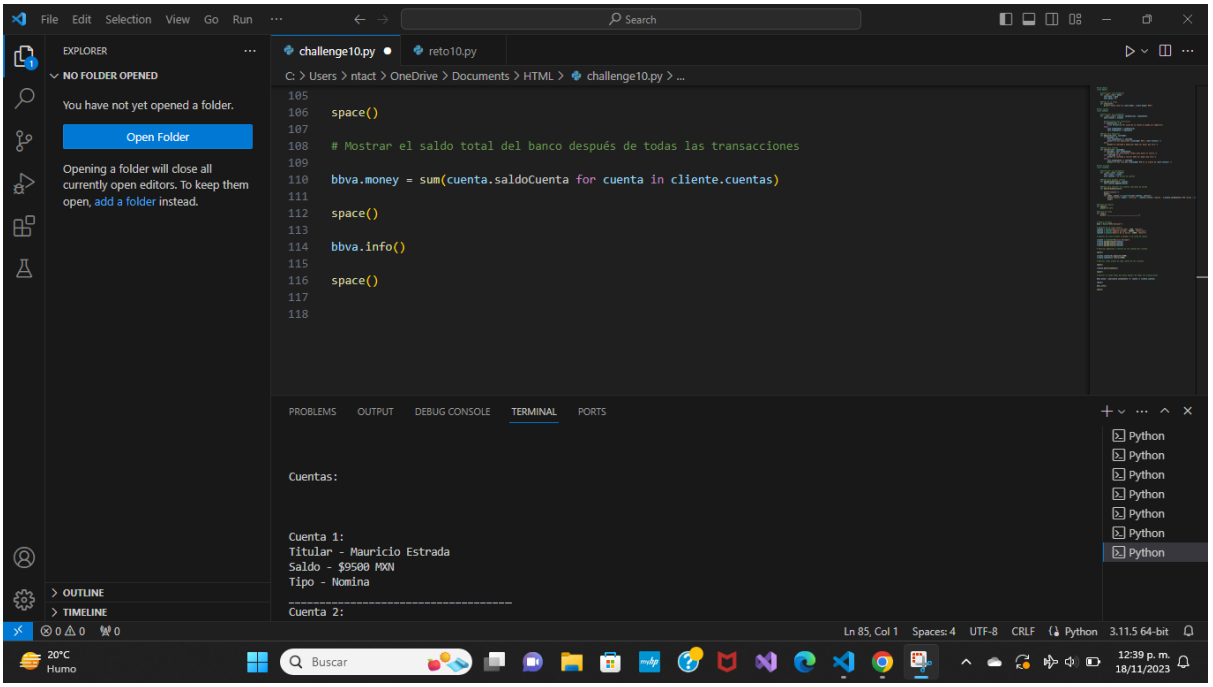
- **Código Fuente, archivos con extensión .py (Recuerda, cualquier otro tipo de archivo no es permitido, i.e. pyd, .class, .zip, .jar, etc)**
- **Compilación sin errores.**
- **Cumplimiento de Especificaciones.**
- **Reporte con las pantallas de ejecución respectivas por cada rubro.**
- **¡¡¡¡¡ Reflexión individual por integrante del equipo. !!!!!**

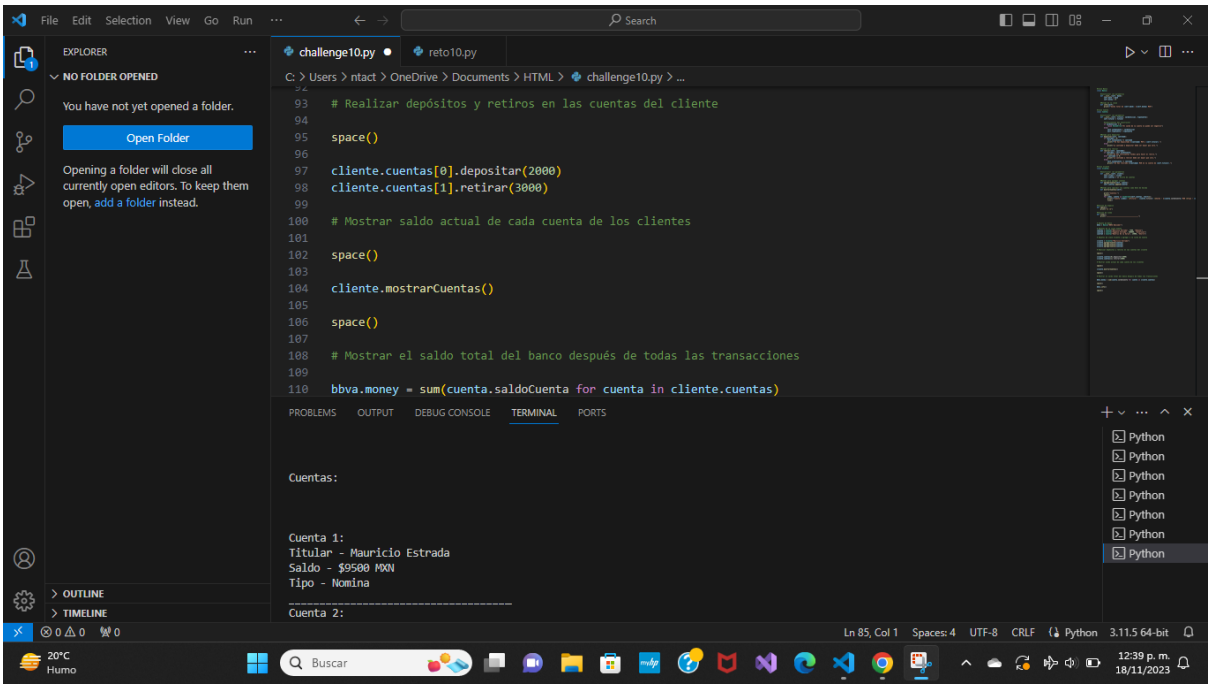
Importante: Solo se revisa la última entrega

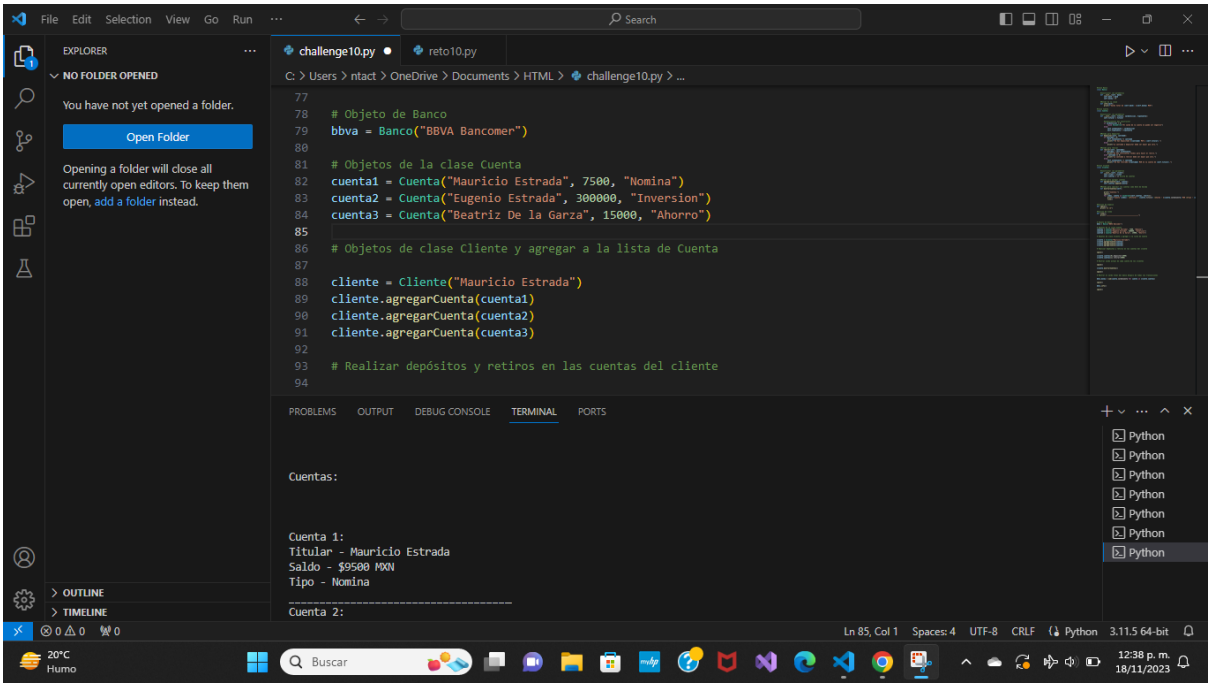
*** Sin reporte y/o código fuente, ningún porcentaje es considerado ⇒ Calificación NE**

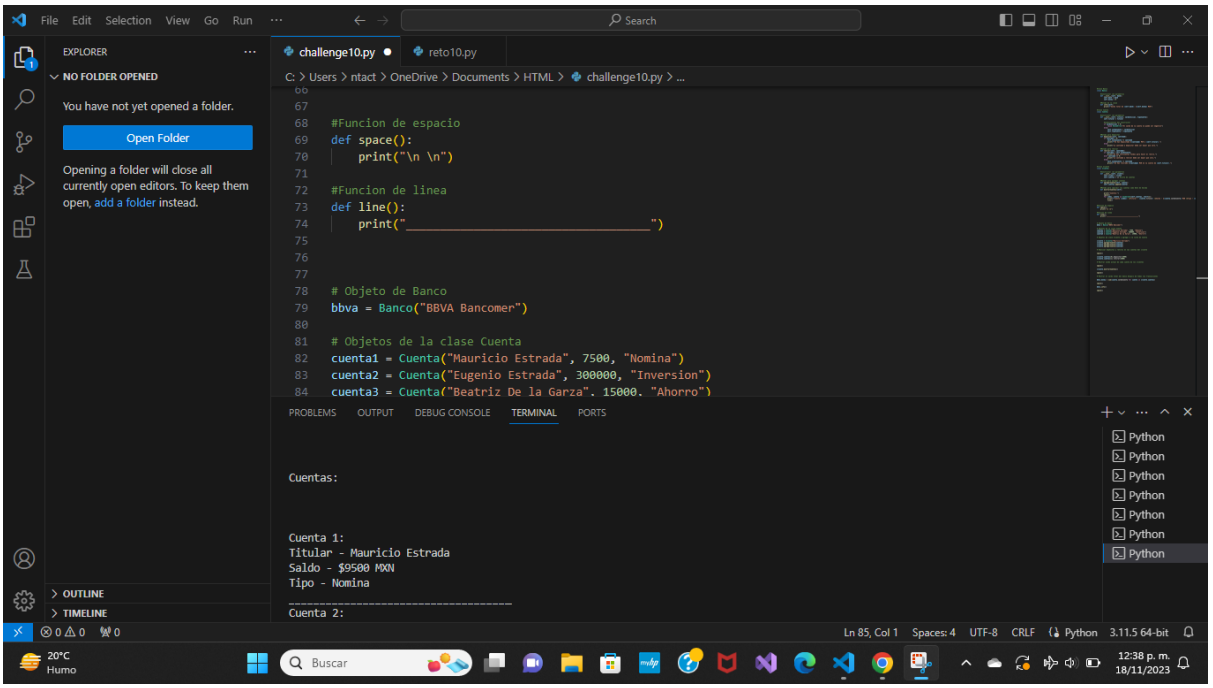
Restricción: Equipos de no más de tres personas, no menos de dos. 😀

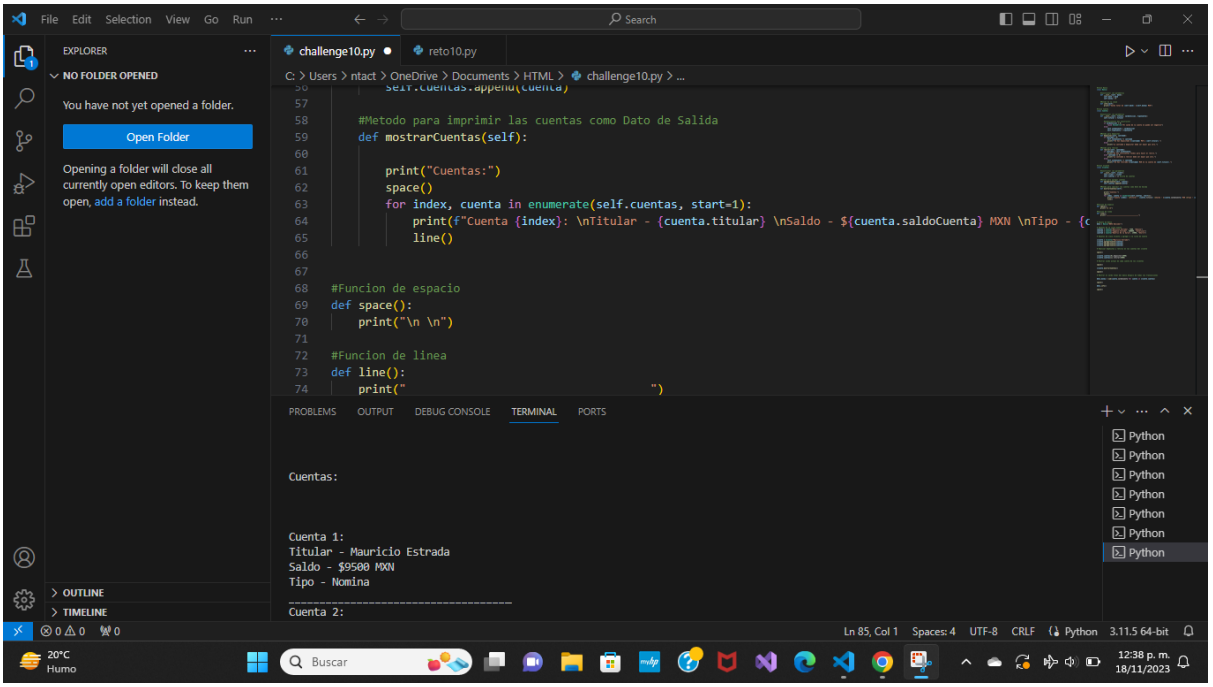
CÓDIGO:

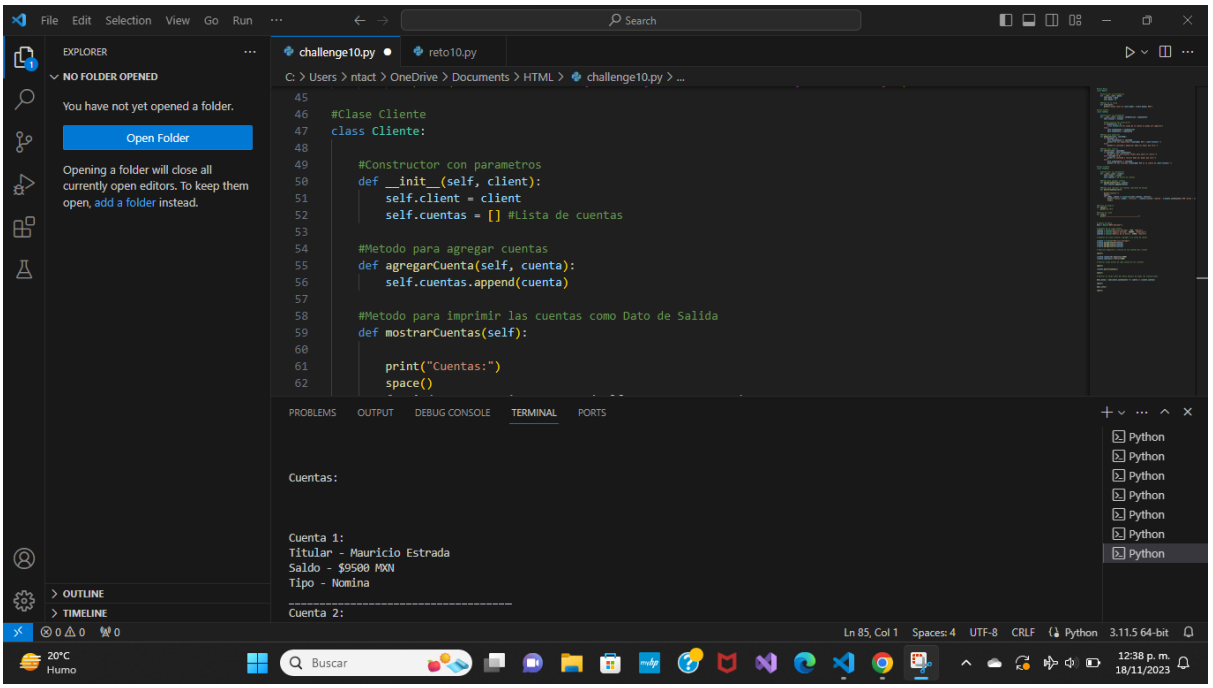

File Edit Selection View Go Run
Search
EXPLORER
NO FOLDER OPENED
You have not yet opened a folder.
Open Folder
Opening a folder will close all currently open editors. To keep them open, add a folder instead.
challenge10.py
reto10.py
C: > Users > ntact > OneDrive > Documents > HTML > challenge10.py > ...
space()
# Mostrar el saldo total del banco después de todas las transacciones
bbva.money = sum(cuenta.saldoCuenta for cuenta in cliente.cuentas)
space()
bbva.info()
space()
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS
Cuentas:
Cuenta 1:
Titular - Mauricio Estrada
Saldo - $9500 MXN
Tipo - Nomina
Cuenta 2:
OUTLINE
TIMELINE
Ln 85, Col 1 Spaces: 4 UTF-8 CRLF Python 3.11.5 64-bit
20°C Humo
Buscar
12:39 p. m. 18/11/2023

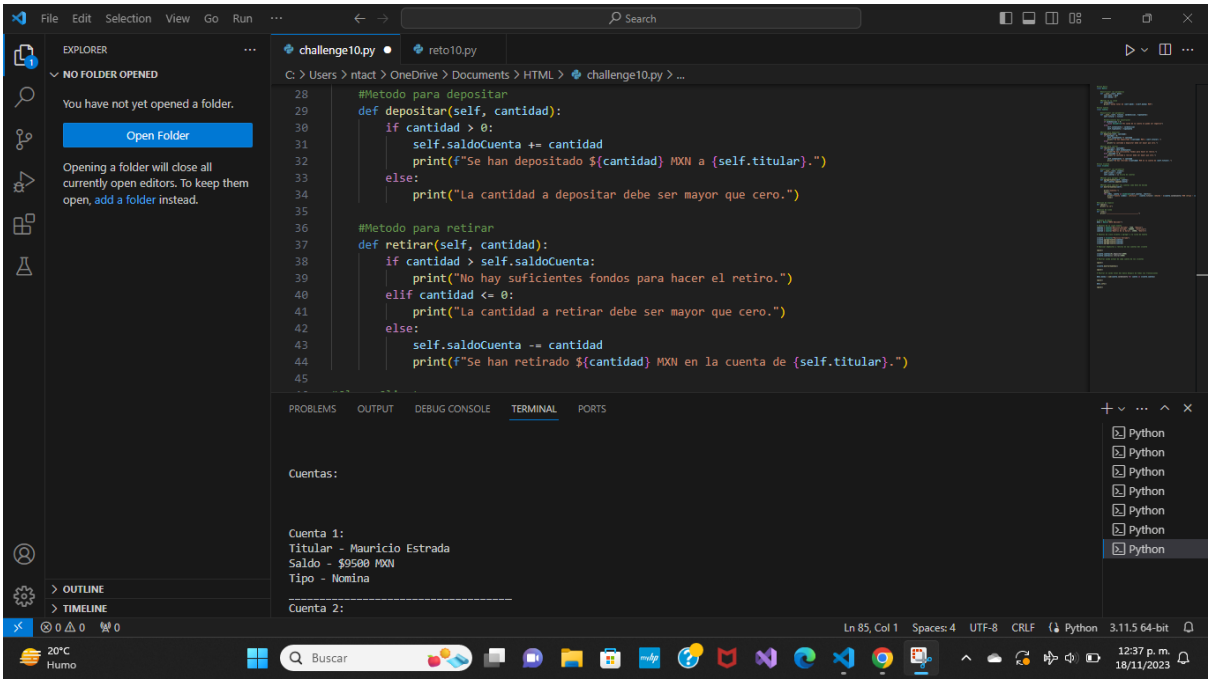

File Edit Selection View Go Run
Search
EXPLORER
NO FOLDER OPENED
You have not yet opened a folder.
Open Folder
Opening a folder will close all currently open editors. To keep them open, add a folder instead.
challenge10.py
reto10.py
C: > Users > ntact > OneDrive > Documents > HTML > challenge10.py > ...
# Realizar depósitos y retiros en las cuentas del cliente
space()
cliente.cuentas[0].depositar(2000)
cliente.cuentas[1].retirar(3000)
# Mostrar saldo actual de cada cuenta de los clientes
space()
cliente.mostrarCuentas()
space()
# Mostrar el saldo total del banco después de todas las transacciones
bbva.money = sum(cuenta.saldoCuenta for cuenta in cliente.cuentas)
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS
Cuentas:
Cuenta 1:
Titular - Mauricio Estrada
Saldo - $9500 MXN
Tipo - Nomina
Cuenta 2:
OUTLINE
TIMELINE
Ln 85, Col 1 Spaces: 4 UTF-8 CRLF Python 3.11.5 64-bit
20°C Humo
Buscar
12:39 p. m. 18/11/2023

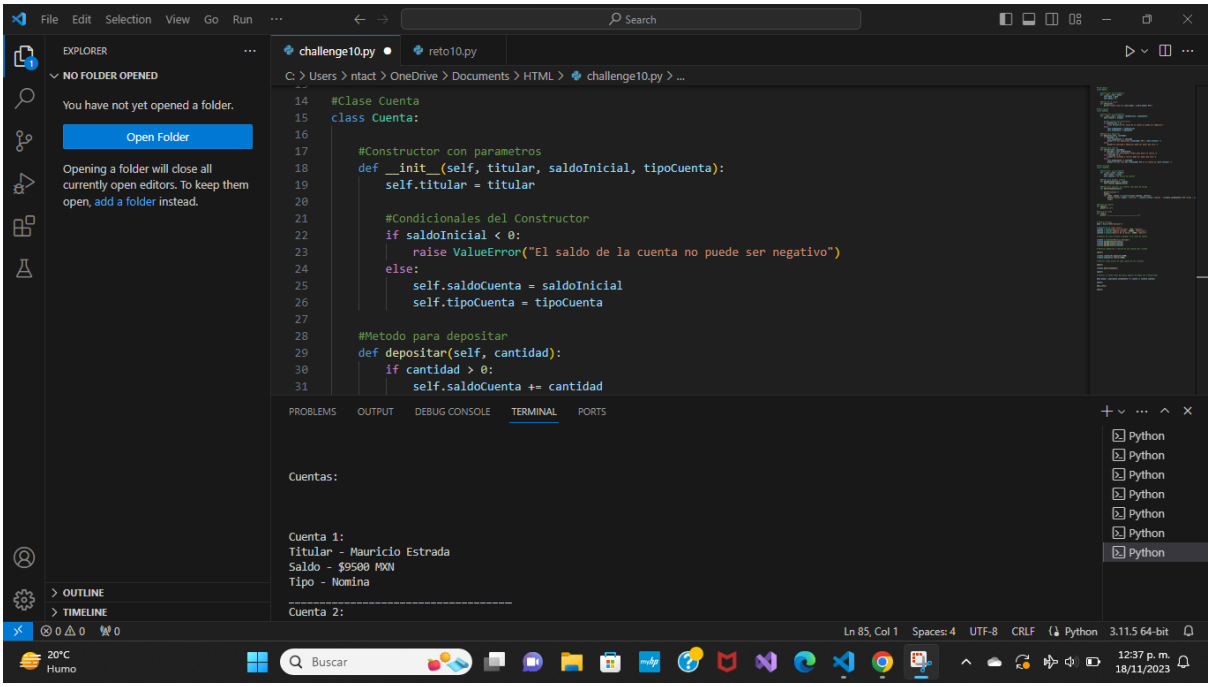

File Edit Selection View Go Run
Search
EXPLORER
NO FOLDER OPENED
You have not yet opened a folder.
Open Folder
Opening a folder will close all currently open editors. To keep them open, add a folder instead.
challenge10.py
reto10.py
C: > Users > ntact > OneDrive > Documents > HTML > challenge10.py > ...
77
78 # Objeto de Banco
79 bbva = Banco("BBVA Bancomer")
80
81 # Objetos de la clase Cuenta
82 cuenta1 = Cuenta("Mauricio Estrada", 7500, "Nomina")
83 cuenta2 = Cuenta("Eugenio Estrada", 300000, "Inversion")
84 cuenta3 = Cuenta("Beatriz De la Garza", 15000, "Ahorro")
85
86 # Objetos de clase Cliente y agregar a la lista de Cuenta
87
88 cliente = Cliente("Mauricio Estrada")
89 cliente.agregarCuenta(cuenta1)
90 cliente.agregarCuenta(cuenta2)
91 cliente.agregarCuenta(cuenta3)
92
93 # Realizar depósitos y retiros en las cuentas del cliente
94
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS
Python
Cuentas:
Cuenta 1:
Titular - Mauricio Estrada
Saldo - $9500 MXN
Tipo - Nomina
Cuenta 2:
OUTLINE
TIMELINE
Ln 85, Col 1 Spaces: 4 UTF-8 CRLF Python 3.11.5 64-bit
20°C Humo
Buscar
12:38 p. m. 18/11/2023

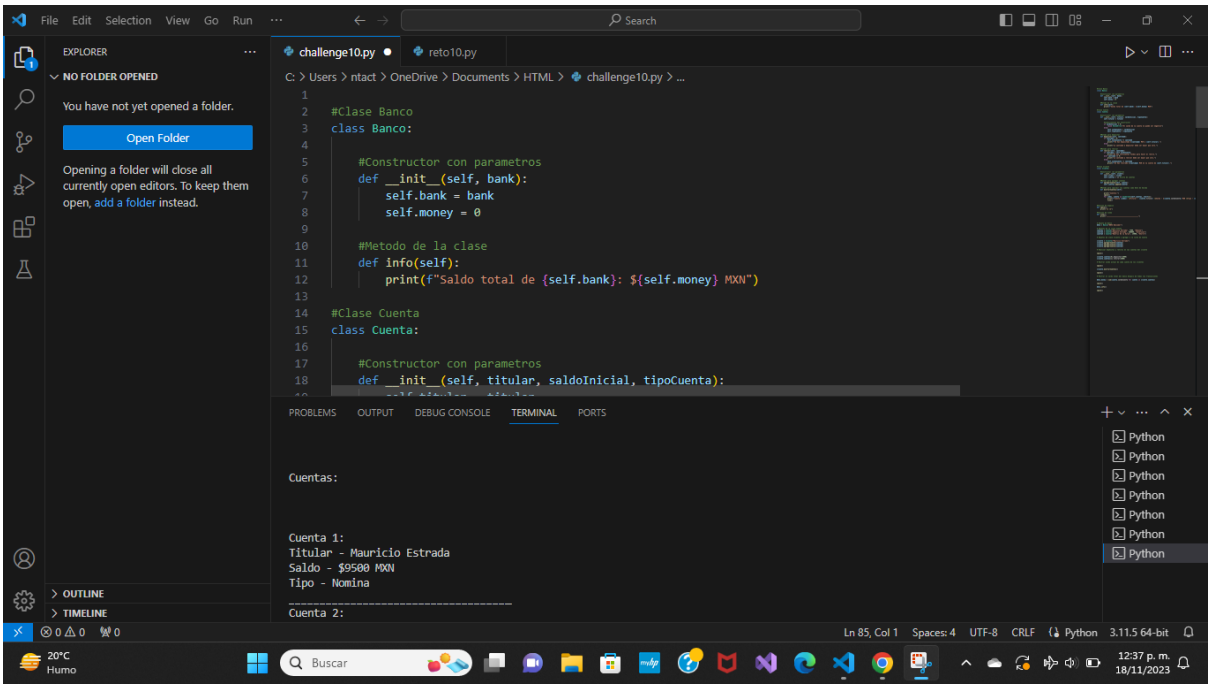

File Edit Selection View Go Run
Search
EXPLORER
NO FOLDER OPENED
You have not yet opened a folder.
Open Folder
Opening a folder will close all currently open editors. To keep them open, add a folder instead.
challenge10.py
reto10.py
C: > Users > ntact > OneDrive > Documents > HTML > challenge10.py > ...
67
68 #Funcion de espacio
69 def space():
70 print("\n \n")
71
72 #Funcion de linea
73 def line():
74 print("_____")
75
76
77
78 # Objeto de Banco
79 bbva = Banco("BBVA Bancomer")
80
81 # Objetos de la clase Cuenta
82 cuenta1 = Cuenta("Mauricio Estrada", 7500, "Nomina")
83 cuenta2 = Cuenta("Eugenio Estrada", 300000, "Inversion")
84 cuenta3 = Cuenta("Beatriz De la Garza", 15000, "Ahorro")
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS
Python
Cuentas:
Cuenta 1:
Titular - Mauricio Estrada
Saldo - $9500 MXN
Tipo - Nomina
Cuenta 2:
OUTLINE
TIMELINE
Ln 85, Col 1 Spaces: 4 UTF-8 CRLF Python 3.11.5 64-bit
20°C Humo
Buscar
12:38 p. m. 18/11/2023


challenge10.py
reto10.py
C: > Users > ntact > OneDrive > Documents > HTML > challenge10.py > ...
#Metodo para imprimir las cuentas como Dato de Salida
def mostrarCuentas(self):
print("Cuentas:")
space()
for index, cuenta in enumerate(self.cuentas, start=1):
print(f"Cuenta {index}: \nTitular - {cuenta.titular} \nSaldo - ${cuenta.saldoCuenta} MXN \nTipo - {c
line()
#Funcion de espacio
def space():
print("\n \n")
#Funcion de linea
def line():
print("
")
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS
Cuentas:
Cuenta 1:
Titular - Mauricio Estrada
Saldo - $9500 MXN
Tipo - Nomina
Cuenta 2:
Ln 85, Col 1 Spaces: 4 UTF-8 CRLF Python 3.11.5 64-bit
20°C Humo
Buscar
12:38 p. m. 18/11/2023

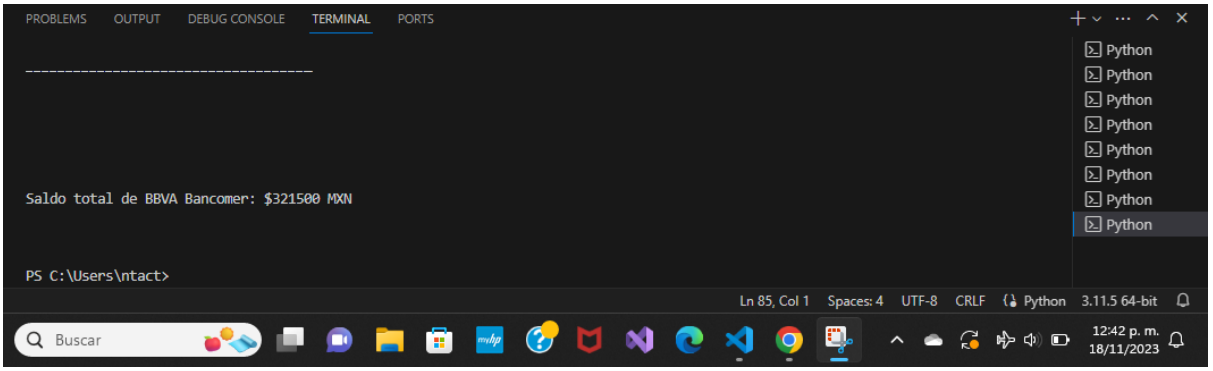

challenge10.py
reto10.py
C: > Users > ntact > OneDrive > Documents > HTML > challenge10.py > ...
#Clase Cliente
class Cliente:
#Constructor con parametros
def __init__(self, client):
self.client = client
self.cuentas = [] #Lista de cuentas
#Metodo para agregar cuentas
def agregarCuenta(self, cuenta):
self.cuentas.append(cuenta)
#Metodo para imprimir las cuentas como Dato de Salida
def mostrarCuentas(self):
print("Cuentas:")
space()
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS
Cuentas:
Cuenta 1:
Titular - Mauricio Estrada
Saldo - $9500 MXN
Tipo - Nomina
Cuenta 2:
Ln 85, Col 1 Spaces: 4 UTF-8 CRLF Python 3.11.5 64-bit
20°C Humo
Buscar
12:38 p. m. 18/11/2023

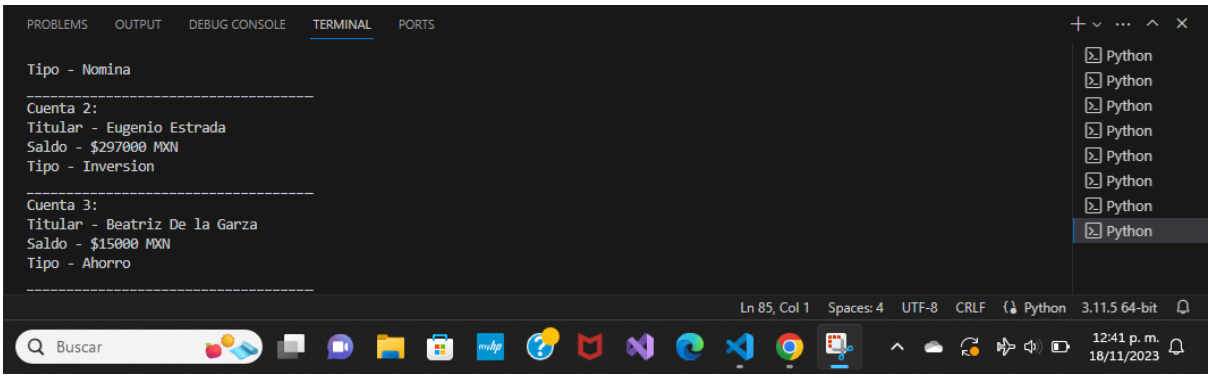

File Edit Selection View Go Run
Search
EXPLORER
NO FOLDER OPENED
You have not yet opened a folder.
Open Folder
Opening a folder will close all currently open editors. To keep them open, add a folder instead.
challenge10.py
reto10.py
C: > Users > ntact > OneDrive > Documents > HTML > challenge10.py > ...
#Metodo para depositar
def depositar(self, cantidad):
if cantidad > 0:
self.saldoCuenta += cantidad
print(f"Se han depositado ${cantidad} MXN a {self.titular}.")
else:
print("La cantidad a depositar debe ser mayor que cero.")
#Metodo para retirar
def retirar(self, cantidad):
if cantidad > self.saldoCuenta:
print("No hay suficientes fondos para hacer el retiro.")
elif cantidad <= 0:
print("La cantidad a retirar debe ser mayor que cero.")
else:
self.saldoCuenta -= cantidad
print(f"Se han retirado ${cantidad} MXN en la cuenta de {self.titular}.")
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS
Python
Cuentas:
Cuenta 1:
Titular - Mauricio Estrada
Saldo - $9500 MXN
Tipo - Nomina
Cuenta 2:
OUTLINE
TIMELINE
Ln 85, Col 1 Spaces: 4 UTF-8 CRLF Python 3.11.5 64-bit
20°C Humo
Buscar
12:37 p. m. 18/11/2023

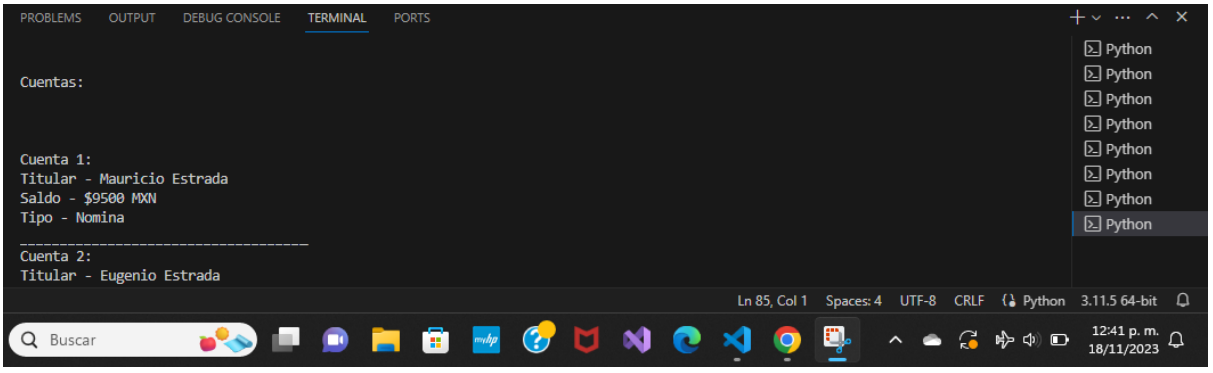

File Edit Selection View Go Run
Search
EXPLORER
NO FOLDER OPENED
You have not yet opened a folder.
Open Folder
Opening a folder will close all currently open editors. To keep them open, add a folder instead.
challenge10.py
reto10.py
C: > Users > ntact > OneDrive > Documents > HTML > challenge10.py > ...
#Clase Cuenta
class Cuenta:
#Constructor con parametros
def __init__(self, titular, saldoInicial, tipoCuenta):
self.titular = titular
#Condicionales del Constructor
if saldoInicial < 0:
raise ValueError("El saldo de la cuenta no puede ser negativo")
else:
self.saldoCuenta = saldoInicial
self.tipoCuenta = tipoCuenta
#Metodo para depositar
def depositar(self, cantidad):
if cantidad > 0:
self.saldoCuenta += cantidad
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS
Cuentas:
Cuenta 1:
Titular - Mauricio Estrada
Saldo - $9500 MXN
Tipo - Nomina
Cuenta 2:
OUTLINE
TIMELINE
Ln 85, Col 1 Spaces: 4 UTF-8 CRLF Python 3.11.5 64-bit
20°C Humo
Buscar
12:37 p. m. 18/11/2023

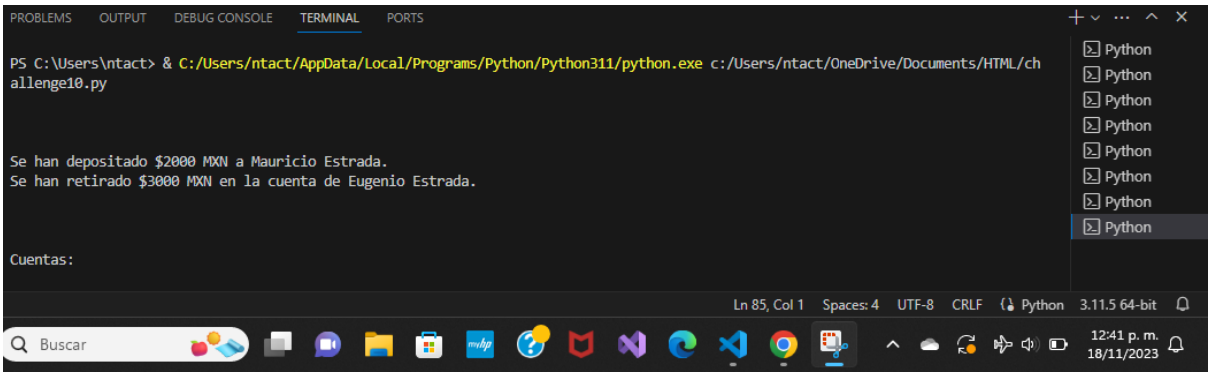

File Edit Selection View Go Run
Search
EXPLORER
NO FOLDER OPENED
You have not yet opened a folder.
Open Folder
Opening a folder will close all currently open editors. To keep them open, add a folder instead.
challenge10.py
reto10.py
C: > Users > ntact > OneDrive > Documents > HTML > challenge10.py > ...
#Clase Banco
class Banco:
#Constructor con parametros
def __init__(self, bank):
self.bank = bank
self.money = 0
#Metodo de la clase
def info(self):
print(f"Saldo total de {self.bank}: ${self.money} MXN")
#Clase Cuenta
class Cuenta:
#Constructor con parametros
def __init__(self, titular, saldoInicial, tipoCuenta):
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS
Cuentas:
Cuenta 1:
Titular - Mauricio Estrada
Saldo - $9500 MXN
Tipo - Nomina
Cuenta 2:
OUTLINE
TIMELINE
Ln 85, Col 1 Spaces: 4 UTF-8 CRLF Python 3.11.5 64-bit
20°C Humo
Buscar
12:37 p. m. 18/11/2023

PRUEBAS:









REPORTE:

- Lo primero que hice en el programa fue crear la clase Banco, le implemente el constructor con parametros, el atributo del saldo lo declare como 0 para que pueda personalizar el atributo como quiera y declararle cualquier valor en el Main, luego implemente un método para que imprimiera el saldo total del Banco.

- Después hice la clase de cuenta, los parámetros son el nombre del titular, el saldo y el tipo de cuenta, dentro del constructor hice una condicional, donde si el saldo inicial es menor a 0 entonces la cuenta no pueda funcionar en el programa. de lo contrario se implementan también los demás atributos de la clase. Implemente un método para depositar, e hice una condición donde si la cantidad a depositar es menor a cero no se pueda ejecutar la función. También hice un metodo para retirar dinero de las cuentas, e implementa las siguientes condicionales, si la cantidad a retirar era mayor que el saldo de la cuenta, entonces no se podría ejecutar la función, en el "elif" si la cantidad era menor o igual a 0, tampoco se podrá ejecutar la función, para que se ejecute la función la cantidad a retirar siempre tiene que ser mayor a cero pero también la cantidad debe ser menor o igual al saldo de la cuenta.

- La última clase que implemente fue la de cliente, en el constructor agregue una lista donde se guardará la información de cada cliente. Hice un método para poder guardar la información en la lista cada vez que se creara una nueva cuenta, similar a la función de "add" en Java, solamente que en Python se llama "append". Y por último, implementé una función para que se imprimiera la información de las cuentas como dato de salida, al igual que en Java, para mostrar la información de una lista, necesitas usar un ciclo "FOR" y así funcione bien el programa, eso fue lo que hice exactamente.

- Ya solamente agregue las funciones que siempre pongo en los programas. Después instancie un objeto de la clase Banco y le puse "BBVA" porque pues donde tengo mi cuenta de nómina, después instancie 3 objetos de la clase Cuenta, la primera cuenta esta a mi nombre, y las otras 2 a nombre de mis padres, a las 3 cuentas les pase los parámetros del tipo de cuenta y el saldo además del nombre de titular. Después a los 3 objetos los agregue en la lista que declare en la clase Cliente. Luego mande a llamar el método donde deposito y retiro saldo en las cuentas de los objetos que instancie, y después mandé a llamar el método para que mostrara la información de las cuentas ya actualizadas después de implementar el método de depositar y retirar. Y ya por último solamente imprime como dato de salida todas las transacciones que se habían hecho en el programa y el saldo total del Banco usando el ciclo FOR al igual que en el método para mostrar las cuentas de la lista de clientes.

REFLEXIÓN:

-La verdad no esta muy complejo la sintaxis de Python pero prefiero programar con Objetos en Java ya que te ahorrar más tiempo con el IDE implementando los constructores, getters y setters, y cambio en Python tienes que hacer todo eso manual, y puede que te tardes mas.