

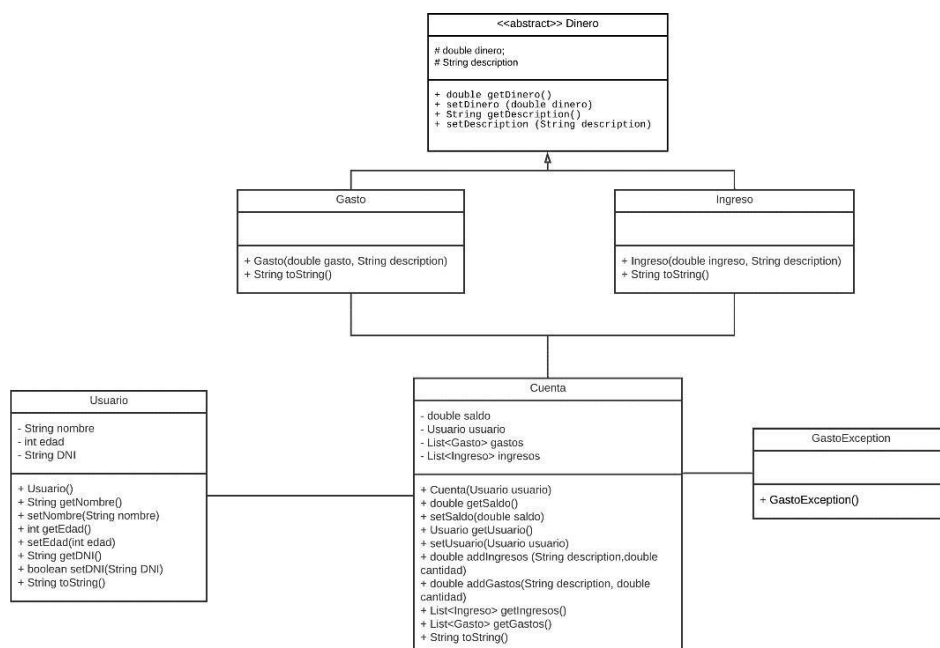
Introducción a python:

En esta práctica se evaluarán vuestros conocimientos en programación orientada a objetos. Leed bien el enunciado ya que debéis crear las clases y las funciones con los nombres que se indican en este documento.

La práctica consiste en crear una aplicación de gestión de gastos personales. A través de un menú interactivo por la consola, introducir ingresos y gastos para así poder llevar un pequeño control de nuestra economía.

El siguiente diagrama de clases representa la estructura que debéis realizar:

DIAGRAMA DE CLASES (está planteado para java)



Importante: Recordad que es obligatorio que **todas las funciones y métodos tengan el mismo nombre y parámetros** que se indican en el diagrama de clases anterior.

Clase Usuario

Esta clase será la encargada de gestionar un único usuario. Éste se creará al inicio del programa leyendo datos por el teclado.

Ejemplo de datos correctos:

Nombre: Alberto

Edad: 23

El DNI deberá tener un formato concreto y devolverá un booleano conforme es correcto o no. Si el DNI es correcto quedará asignado.

Formato correcto:

- Los primeros 8 caracteres solo podrán ser numéricos.
 - El ultimo caracteres deberá ser una letra entre la A y la Z.
 - El guion entre los números y la letra es opcional admitiendo ambas posibilidades.
- DNI: 78844112L
DNI: 78844112-L

Tendrá una función toString con la que devolver su contenido.

Clase Dinero:

Es una clase abstracta. En este caso, en Python, las clases abstractas se definen como ABC (Abstract Base Class)

Para crear una clase abstracta hay que importar ABC y abstractmethod del módulo abc.

Ejemplo:

```
from abc import ABC, abstractmethod

# 1. La clase hereda de ABC
class Animal(ABC):
    def __init__(self, nombre):
        self.nombre = nombre

    # 2. Marcamos el método que debe ser obligatorio en las clases hijas
    @abstractmethod
    def hacer_sonido(self):
        """Método que obliga a las clases hijas a definir un sonido."""
        pass

class Perro(Animal):
    def hacer_sonido(self):
        return "Guau!"
```

Clases Gasto e Ingreso:

Las clases Gasto e Ingreso heredarán de Dinero y tendrán un único constructor en el que se inicializarán los valores recibidos por parámetros. Además, tendrán una función toString con la que devolver su contenido.

Clase Cuenta:

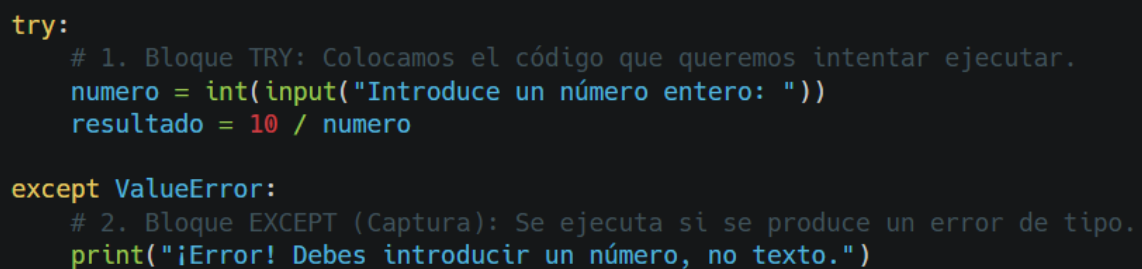
Clase donde se gestionarán todos los movimientos de dinero tanto ingresos como gastos.

Inicialmente (en el constructor) se recibirá el usuario que es dueño de la cuenta y el saldo inicial será de 0€.

Al añadir un nuevo ingreso se sumará al saldo de la cuenta teniendo en esta variable nuestro dinero real, la función devolverá el saldo de la cuenta.

Al añadir un nuevo gasto se debe comprobar si se dispone de saldo suficiente, en caso contrario se deberá lanzar una nueva excepción del tipo GastoException(), pero el programa no debe finalizar. Si se dispone de saldo suficiente se restará el importe del gasto y se devolverá el saldo de la cuenta.

En Python trabajaremos con **try/except** para las excepciones. Podemos trabajar con la directiva **raise** para lanzar la excepción cuando se produzca:



```
try:
    # 1. Bloque TRY: Colocamos el código que queremos intentar ejecutar.
    numero = int(input("Introduce un número entero: "))
    resultado = 10 / numero

except ValueError:
    # 2. Bloque EXCEPT (Captura): Se ejecuta si se produce un error de tipo.
    print("¡Error! Debes introducir un número, no texto.")
```

Tendrán una función toString con la que devolverá el usuario y su saldo.

Main:

La clase main será la que se ejecute al iniciar el programa y seguirá unos pasos definidos:

1. Creación del usuario y sus datos, el DNI no se establecerá hasta que se introduzca uno correcto, el orden de los datos será:
 - a. Nombre
 - b. Edad
 - c. DNI
2. Creación de la cuenta
3. Visualización del menú con las instrucciones tal y como se muestra en la siguiente figura:

```
Realiza una nueva acción
1 Introduce un nuevo gasto
2 Introduce un nuevo ingreso
3 Mostrar gastos
4 Mostrar ingresos
5 Mostrar saldo
0 Salir
```

4. Cada acción realizará una operación donde se deberán de solicitar los datos si los necesitase.
5. Al finalizar la aplicación se deberá mostrar el mensaje (**importante que sea igual al que se indica**):

Fin del programa.

Gracias por utilizar la aplicación.

¡Buen trabajo!

