CURSO DE PROGRAMACIÓN FULL-STACK

Python / Guía 9

Ejercicios Programación Orientada a Objetos II

Ejercicios de aprendizaje

1. Métodos de clase y Factory Method:

    - Crea una clase Vehículo con una variable de clase "modelo"dos variables de instancia: "nombre" y "precio".
    - Crea un método de clase "con_rebaja" que sea un patrón Factory Method, el cual debe crear un objeto con un descuento del 20%.
    - Crea un método de instancia "mostrar", imprimirá el nombre y el precio.
    - Crea la clase Auto que heredará de Vehículo.
    - Crea un método de instancia cualquiera en Auto.
    - Crea una instancia de Auto. Invoca mostrar()
    - Crea otra instancia de Auto, utilizando el patrón Factory Method.
    - Invoca mostrar()

2. Continuando el ejercicio anterior, crea un método estático en Vehículo que sirva para validar que el precio ingresado sea mayor a 1000 y menor a 10.000.

3. A partir de las siguientes pruebas unitarias, realiza "ingeniería inversa" creando el módulo "figuras" y creando las dos clases correspondientes **usando dataclasses**. No debe fallar ninguna prueba.

```python
import pytest
import figuras


def test_herencias():
    assert issubclass(figuras.Cuadrado, figuras.Rectángulo)


def test_Rectángulo():
    rectángulo = figuras.Rectángulo(5, 10)
    # getters
    assert rectángulo.base == 5
    assert rectángulo.área == 50
    assert rectángulo.altura == 10
    assert rectángulo.diagonal == 11.180339887498949
    # setters
    rectángulo.base = 5
    rectángulo.altura = 10
    # errores
    with pytest.raises(ValueError):
        rectángulo.base = 0

    with pytest.raises(ValueError):
        rectángulo.altura = 0


def test_Cuadrado():
    cuadrado = figuras.Cuadrado(9)
    # getters
    assert cuadrado.lado == 9
    assert cuadrado.base == 9
    assert cuadrado.altura == 9
    assert cuadrado.área == 81
    assert cuadrado.diagonal == 12.727922061357855
    # setters
    cuadrado.lado = 9
    cuadrado.base = 9
    cuadrado.altura = 9
    # errores
    with pytest.raises(ValueError):
        cuadrado.lado = 0
    with pytest.raises(ValueError):
        cuadrado.base = 0
    with pytest.raises(ValueError):
        cuadrado.altura = 0
```

Te dejo los valores de las diagonales para que copies y pegues:

```
assert rectángulo.diagonal == 11.180339887498949
assert cuadrado.diagonal == 12.727922061357855
```

Cálculo del área: base * altura
Cálculo de la diagonal: (base ** 2 + altura ** 2) ** 0.5

Trata de **identificar el error que hay en la prueba unitaria**, que puede llevar a bugs.

Ver video "Ingeniería Inversa"

4. A partir de los ejercicios hechos en las guías anteriores, y teniendo en cuenta lo aprendido hasta ahora, utilizando las mejores prácticas de modularización, funciones, y programación orientada a objetos:

   La nueva librería de la ciudad cuenta con una gran biblioteca, salas de lectura confortantes y una sala de bebidas, como el café. Cada persona que entra puede registrarse, y si está registrada, muestra su código QR para poder pasar. Con su código puede pedir libros y bebidas. Te han contratado para que crees una aplicación para el negocio. Deberá:

   1. Registrar libros (título, autor...)
   2. Registrar bebidas (nombre, precio, cantidad...)
   3. Registrar al cliente (nombre, teléfono, tarjeta...)
   4. Registrar una tarjeta de crédito para cada cliente (titular, nº tajeta...) con un límite máximo de compra.
   5. El cliente puede solicitar libros, que tendrán un precio por pedido
   6. El cliente puede comprar bebidas, que tendrán su propia precio
   7. Registrar qué libros están prestados y cuáles están disponibles
   8. Llevar un historial de ventas

   Menú principal:

- Registro de Clientes
- Registro de Libros
- Registro de Bebidas
- Préstamos de Libros
- Ventas de Bebidas

Clientes:

- Alta cliente
- Modificar cliente
- Baja cliente
- Listar clientes

Libros:

- Alta libro
- Modificar libro
- Baja libro
- Listar libros

Bebidas:

- Alta bebida
- Modificar bebida
- Baja bebida
- Listar bebidas

Préstamos de libros:

- Prestar a cliente
- Cliente devuelve libro
- Lista de libros disponibles
- Lista de libros prestados

Venta de bebidas:

- Vender a cliente
- Ver historial de ventas

Recuerda empezar por lo más simple, y una vez que vayan funcionando las partes simples del código, seguir con las pruebas unitarias. Luego haz

crecer el código de manera prolija, en partes. No uses aún conceptos que no has aprendido.

Utiliza atributos protegidos o privados cuando realmente los necesites. De la misma forma que las propiedades. Recuerda que son utilidades propias del programador, y no tanto al usuario final, salvo que el usuario sea un desarrollador y tenga que crear instancias de clase por sí mismo, o llamar los atributos y métodos para manejar las clases.