

# Programación Orientado a Objetos (Herencia y Relaciones)

PAREJAS Alejandro Oquendo Bedoya

Yeison Orozco Vasco

Docente: Alejandro Rodas Vásquez

Universidad Tecnológica de Pereira

1 de noviembre de 2025

## Introducción

Una de las claves para realizar este proyecto es aplicar el concepto de modularidad en la construcción de la Arquitectura de Software que soporta la aplicación.

## 1. Requerimientos Funcionales

Usted ha sido contratado para realizar un sistema de facturación para una tienda agrícola. Donde cada factura (o Pedido) está compuesto de los productos que serán comprados.

Esta tienda solamente maneja Productos de Control (Fertilizantes y Controles de plagas) y medicina para animales de granja, precisamente antibióticos.

Los Productos de Control tendrán como características un registro ICA, el nombre del producto y la frecuencia de aplicación (es decir, cada cuanto periodo se aplica el producto. Cada 15 días, cada 30 días, etc) así como también el valor del producto. Tenga en cuenta que el Control de Plagas y el Control de Fertilizantes son *un tipo de* Productos de Control, donde el primero tiene como característica un periodo de carencia (es el tiempo legalmente establecido, expresado usualmente en número de días que debe transcurrir entre la última aplicación de un fitosanitario y la cosecha) y el segundo la fecha de la última aplicación de este Producto.

Por otro lado, en la tienda se venden antibióticos para bovinos y porcinos donde las características de este producto son: nombre del producto, dosis (entre 400Kg y 600Kg), tipo de animal al que se le puede aplicar (Bovinos, caprinos o porcinos) y precio.

Tenga en cuenta que al ser una tienda agrícola los Clientes (con atributos nombre y cédula) son habituales por lo tanto el mismo cliente puede tener dentro de su historial de

compras, muchas Pedidos (o Facturas) asociadas. Una Factura como tal debe tener fecha en que se realizó la factura y el valor total de la compra.

Tenga en cuenta que todos los atributos de las clases son *obligatorios*. Con esta información puede diseñar los *casos de prueba*.

## 2. Requerimientos de la Arquitectura de Software

Esta aplicación debe de ser construida bajo los siguientes parámetros arquitectónicos:

1. Los componentes para separar responsabilidades (Modelo, Test).
2. Utilicen el concepto de *Módulos y NameSpace*.
3. Cada *Clase* debe de estar en un archivo separado dentro del *Componente de Modelo*.

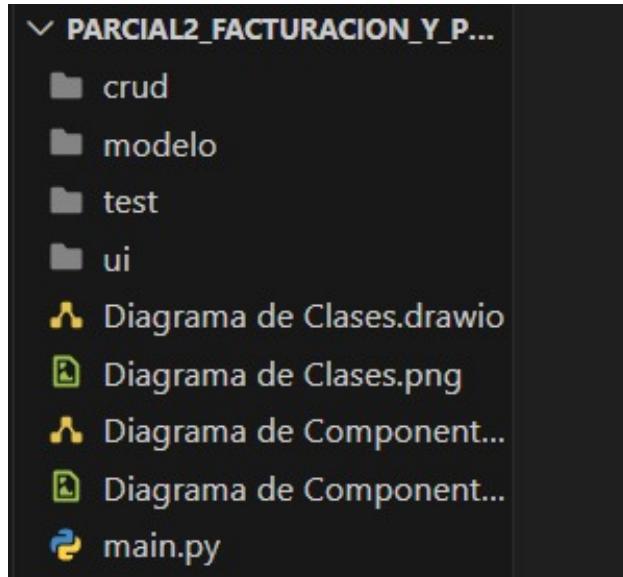


fig. 1: Arquitectura del proyecto

## 3. ¿Cómo realizo la entrega?

1. Usted debe de entregar el código fuente en su repositorio de *github*.
2. Pantallazos donde se corrobore que las pruebas unitarias han pasado.
3. Pantallazos donde se corrobore el uso del **debug**. En estas imágenes debe de observar la composición del objeto. Es decir, se evidencia que el objeto *x* tiene *asociado n instancias* del objeto *y*. Lo mismo con la herencia.
4. Realizar el Diagrama de Clases y Diagrama de Componentes.

## 4. Evidencias

Repositorio de Git:

[https://github.com/Yeyei121/Parcial2\\_Facturacion\\_y\\_pruebas.git](https://github.com/Yeyei121/Parcial2_Facturacion_y_pruebas.git)

### 4.1. Diagrama de Clases

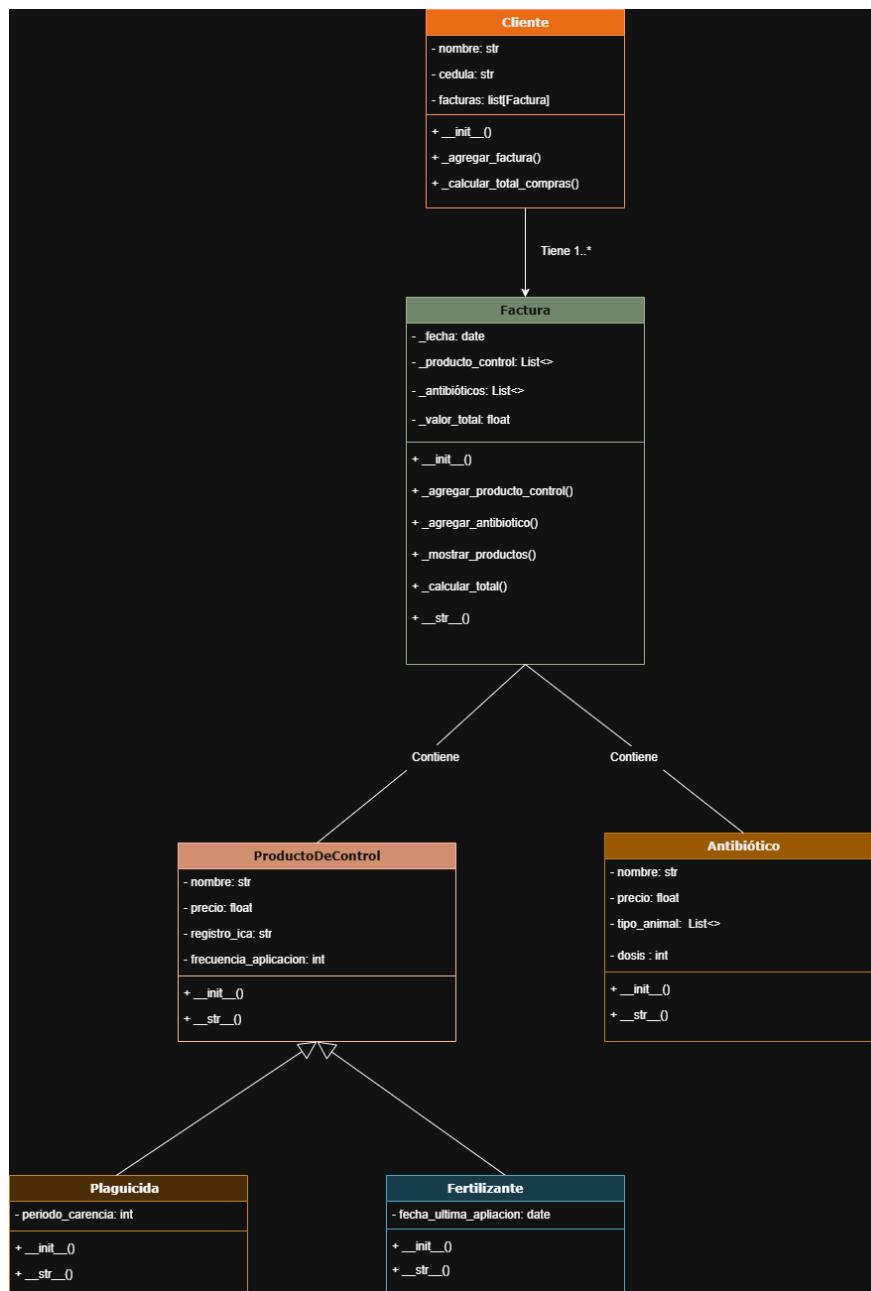


fig. 2: Diagrama de Clases

## 4.2. Diagrama de Componentes

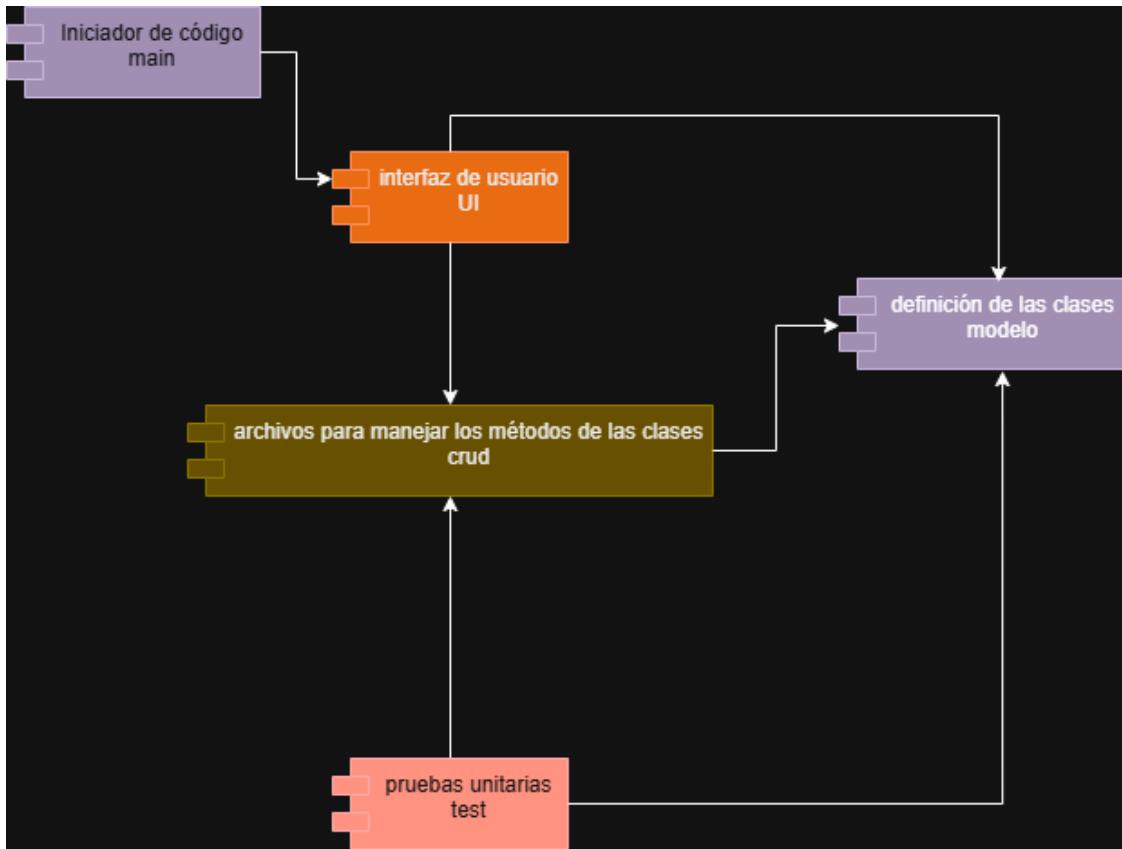


fig. 3: Diagrama de Componentes

## 4.3. Pantallazos Pruebas Unitarias

```
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

PS C:\Users\yeiso\OneDrive\Documentos\Parcial2_Facturacion_y_Pruebas> python -m unittest -v test.test_cliente

-----
Ran 0 tests in 0.000s

NO TESTS RAN
test_agregar_factura_a_cliente (test.test_cliente.TestFacturacion.test_agregar_factura_a_cliente) ... ok
test_agregar_varias_facturas (test.test_cliente.TestFacturacion.test_agregar_varias_facturas) ... ok
test_creacion_cliente (test.test_cliente.TestFacturacion.test_creacion_cliente) ... ok
test_obtener_facturas_vacias (test.test_cliente.TestFacturacion.test_obtener_facturas_vacias) ... ok

-----
Ran 4 tests in 0.000s

OK
PS C:\Users\yeiso\OneDrive\Documentos\Parcial2_Facturacion_y_Pruebas>
```

fig. 4: Pruebas unitarias para clientes

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

```
PS C:\Users\yeiso\OneDrive\Documentos\Parcial2_Facturacion_y_Puebas> python -m unittest -v test.test_facturas
test_agregar_antibiotico (test.test_facturas.TestFacturaCRUD.test_agregar_antibiotico)
Prueba agregar un antibiótico a la factura. ... ok
test_agregar_producto_control (test.test_facturas.TestFacturaCRUD.test_agregar_producto_control)
Prueba agregar un producto control a la factura. ... ok
test_buscar_factura (test.test_facturas.TestFacturaCRUD.test_buscar_factura)
Prueba la búsqueda de una factura por fecha. ... ok
test_calcular_total (test.test_facturas.TestFacturaCRUD.test_calcular_total)
Prueba el cálculo del total de la factura. ... ok
test_crear_factura (test.test_facturas.TestFacturaCRUD.test_crear_factura)
Prueba la creación de una factura. ... ok
test_eliminar_factura (test.test_facturas.TestFacturaCRUD.test_eliminar_factura)
Prueba la eliminación de una factura. ... ok

-----
Ran 6 tests in 0.001s
```

fig. 5: Pruebas unitarias para facturas

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

```
Yeison Orozco Vasco - Cédula: 1087989970
Carlos Alberto Morales - Cédula: 1077643789

--- Menú Principal ---
1. Crear cliente
2. Ver todos los clientes
3. Buscar cliente por cédula
4. Agregar factura a un cliente
5. Agregar producto a una factura
6. Salir
Seleccione una opción: 5
Ingrese la cédula del cliente: 1087989970

--- Facturas del Cliente ---
1. Factura del 2025-11-01 - Total: $100.00 - 1 productos
Seleccione el número de la factura: |
```

fig. 6: Interfaz del programa

## 4.4. Pantallazos Debug

```

main.py debug_file.py

debug_file.py > ...
1 from datetime import date
2 from modelo.factura import Factura
3 from modelo.producto_control import ProductoControl
4 from modelo.antibiotico import Antibiotico
5
6 factura = Factura(date(2025, 11, 1)) factura = <modelo.factura.Factura object at 0x000001D953EA63C0>
7 pi = ProductoControl("Producto Control X", 50000, "ICA123", 30)
8 ai = Antibiotico("Antibiotico A", 75000, 500, "Bovinos")
9 factura.agregar_producto_control(pi) pi = <modelo.producto_control.ProductoControl object at 0x000001D953EA6660>
10 factura.agregar_antibiotico(ai) ai = <modelo.antibiotico.Antibiotico object at 0x000001D953EA67B0>
11 factura.agregar_antibiotico(ai) ai = <modelo.antibiotico.Antibiotico object at 0x000001D953EA67B0>
12
13 print("Debug aqui")

```

CALL STACK: debug\_file.py

BREAKPOINTS: debug\_file.py

PROBLEMS, OUTPUT, DEBUG CONSOLE, TERMINAL, PORTS

PS C:\Users\yeiso\OneDrive\Documentos\Parcial2\_Facturacion\_y\_Puebas & "c:\Users\yeiso\AppData\Local\Programs\Python\Python311\python.exe" "c:\Users\yeiso\vscode\extensions\ms-python.debugpy-2025.14.1-win32-x64\bundled\libs\debugpy\launcher" "52188" -- -- c:\Users\yeiso\OneDrive\Documentos\Parcial2\_Facturacion\_y\_Puebas\debug\_file.py'

fig. 7: Debug1

```

RUN AND DEBUG
No Configurations ... No Configurations

debug_file.py > ...
1 from datetime import date
2 from modelo.factura import Factura
3 from modelo.producto_control import ProductoControl
4 from modelo.antibiotico import Antibiotico
5
6 factura = Factura(date(2025, 11, 1)) factura = <modelo.factura.Factura object at 0x000001D953EA63C0>
7 pi = ProductoControl("Producto Control X", 50000, "ICA123", 30)
8 ai = Antibiotico("Antibiotico A", 75000, 500, "Bovinos")
9 factura.agregar_producto_control(pi) pi = <modelo.producto_control.ProductoControl object at 0x000001D953EA6660>
10 factura.agregar_antibiotico(ai) ai = <modelo.antibiotico.Antibiotico object at 0x000001D953EA67B0>
11 factura.agregar_antibiotico(ai) ai = <modelo.antibiotico.Antibiotico object at 0x000001D953EA67B0>
12
13 print("Debug aqui")

```

RUN AND DEBUG: Run and Debug (Ctrl+Shift+D)

VARIABLES: debug\_file.py

CALL STACK: debug\_file.py

BREAKPOINTS: debug\_file.py

PROBLEMS, OUTPUT, DEBUG CONSOLE, TERMINAL, PORTS

PS C:\Users\yeiso\OneDrive\Documentos\Parcial2\_Facturacion\_y\_Puebas & "c:\Users\yeiso\AppData\Local\Programs\Python\Python311\python.exe" "c:\Users\yeiso\vscode\extensions\ms-python.debugpy-2025.14.1-win32-x64\bundled\libs\debugpy\launcher" "52188" -- -- c:\Users\yeiso\OneDrive\Documentos\Parcial2\_Facturacion\_y\_Puebas\g\_file.py'

fig. 8: Debug2

```
RUN AND DEBUG No Configurations ... debug.pyw ...
VARIABLES
  ✓ locals
    a1 = <modelo.antibiotico.Antibiotico object at 0x000001D953E467B>
      special variables
        _precio = 75000
        _dosis = 500
        _nombre = 'Antibiotico A'
        _precio = 75000
        _tipo_animal = 'Bovinos'
      ✓ Factura = <modelo.factura.Factura object at 0x000001D953E463CD>
        special variables
          facturas = <list> [None, None]
WATCH
```

fig. 9: Debug3 (Antibióticos)

```
debug_file.py > ...
1 from datetime import date
2 from modelo.factura import Factura
3 from modelo.producto_control import ProductoControl
4 from modelo.antibiotico import Antibiotico
5
6
7 factura = Factura(date(2025, 11, 1)) factura = <modelo.factura.Factura object at 0x000001D953EA63C0>
8 pi = ProductoControl("Producto Control X", 50000, "ICA123", 30)
9 al = Antibiotico("Antibiotico A", 75000, 500, "Bovinos")
10 factura.agregar_producto_control(pi) pi = <modelo.producto_control.ProductoControl object at 0x000001D953EA6780>
11 factura.agregar_antibiotico(al) al = <modelo.antibiotico.Antibiotico object at 0x000001D953EA67B0>
12
13 print("Debug aquí")
```

fig. 10: Debug4 Instancia de factura

```
debug_file.py U X
debug_file.py > ...
1  from datetime import date
2  from modelo.factura import Factura
3  from modelo.producto_control import ProductoControl
4  from modelo.antibiotico import Antibiotico
5
6
7  factura = Factura(date(2025, 11, 1))
8  p1 = ProductoControl("Producto Control X", 50000, "ICA123", 30)
9  a1 = Antibiotico("Antibiótico A", 75000, 500, "Bovinos")
10 factura._agregar_producto_control(p1)
11 factura._agregar_antibiotico(a1)
12
13 print("Debug aquí")
```

fig. 11: Código Debug