Universidad Nacional Guillermo Brown
Programación Avanzada – Simulacro 2
Fecha: ____ / ____ / 2025

1) (2 puntos) Verdadero o Falso
   a) El acoplamiento bajo favorece que una clase se pueda reutilizar con menos dependencias.
   b) La encapsulación impide el acceso a los atributos de una clase desde fuera de sus métodos.
   c) type(objeto) devuelve la clase a la que pertenece objeto.
   d) La abstracción consiste en ocultar la complejidad interna y mostrar solo lo esencial.

2) (4 puntos) Herencia simple
Define en Python dos clases:
- Vehiculo: con atributo marca (string) y método info() que devuelve "Vehículo: <marca>".
- Coche: hereda de Vehiculo, añade atributo pasajeros (int) y redefine info() para devolver "Coche: <marca>, pasajeros: <pasajeros>".

3) (4 puntos) Decoradores básicos
   a) Explica en 1–2 líneas qué es un decorador en Python.
   b) Escribe un decorador anunciar que imprima "Ejecutando función..." antes de llamar a la función decorada y luego retorne su resultado. Aplica @anunciar a sumar(a, b).

Universidad Nacional Guillermo Brown
Programación Avanzada – Simulacro 3
Fecha: ____ / ____ / 2025

1) (2 puntos) Verdadero o Falso
   a) Un decorador puede modificar la entrada o salida de una función sin tocar su código interno.
   b) La cohesión mide cuánto están relacionadas las responsabilidades dentro de una misma clase.
   c) La herencia múltiple nunca genera conflictos al resolver métodos con el mismo nombre.
   d) type("Hola") devuelve <class 'str'>, es decir, el tipo de objeto pasado como argumento.

2) (4 puntos) Herencia simple
Crea estas dos clases en Python:
- Persona: atributo nombre y método saludar() que devuelve "Hola, soy <nombre>".
- Empleado: hereda de Persona, añade atributo salario (número) y método saludar() que añade " y gano <salario> USD".

3) (4 puntos) Decoradores y revisión imperativa
   a) Escribe un decorador mayusculas que transforme el resultado (string) de la función decorada a mayúsculas.
   b) Aplica @mayusculas a saludo() para obtener "HOLA MUNDO".