Práctica Programación 2

Los siguientes ejercicios le ayudarán a comprender de manera efectiva los conceptos fundamentales de herencia, abstracción y polimorfismo. Debe implementarlos en el lenguaje de programación C++, preferiblemente en papel y lápiz para que practique para el examen.

Ejercicio 1: Herencia

Cree una jerarquía de clases para vehículos.

1. Defina una clase base `Vehículo` con propiedades como `marca`, `modelo` y `año`.
2. Derive dos subclases de `Vehículo` — `Coche` y `Motocicleta`.
3. Cada subclase debe tener una propiedad adicional única para ella, por ejemplo, `Coche` podría tener `numeroDePuertas`, mientras que `Motocicleta` podría tener `tipo` (por ejemplo, cruiser, deportiva).
4. Implemente un método en la clase `Vehículo` que devuelva una descripción del vehículo. Sobrescriba este método en ambas subclases para proporcionar una descripción más detallada.

Ejercicio 2: Abstracción

Cree una clase abstracta para un Dispositivo Electrónico.

Defina una clase abstracta `DispositivoElectronico` con un método abstracto `duracionBateria()` que devuelva la duración de la batería del dispositivo y otro método `especificaciones()`.

Implemente dos subclases: `Smartphone` y `Tablet`.

1. En la clase `Smartphone`, implementa el método `duracionBateria()` para devolver una duración de batería específica, por ejemplo, 12 horas. Implementa `especificaciones()` para devolver una cadena que contenga información como “Pantalla: 6.1 pulgadas, Procesador: Snapdragon 888”.

2. En la clase `Tablet`, implementa `duracionBateria()` para devolver una duración de batería diferente y mayor, como 15 horas. Implementa `especificaciones()` para devolver una cadena con información como “Pantalla: 10.5 pulgadas, Procesador: A14 Bionic”.

Ejercicio 3: Polimorfismo

Construye una aplicación simple que demuestre el polimorfismo con animales.

1. Cree una clase base `Animal` con un método `sonido()`.
2. Cree subclases `Perro` y `Gato` que sobrescriban el método `sonido()` para devolver "Guau" y "Miau" respectivamente.
3. Escriba una función que acepte una lista de objetos `Animal` y llame al método `sonido()` para cada uno, demostrando el polimorfismo en acción.

Ejercicio 4: Combinando Herencia y Polimorfismo

Diseña un sistema de gestión escolar.

1. Cree una clase base `Persona` con propiedades como `nombre`, `edad`, y métodos como `presentarse()`.
2. Derive dos subclases `Estudiante` y `Profesor`.
3. `Estudiante` debe tener propiedades adicionales como `grado` y un método `estudiar()`, mientras que `Profesor` debe incluir `asignatura` y un método `enseñar()`.
4. Demuestre el polimorfismo creando un método que acepte una lista de objetos `Persona` (estudiantes y profesores) y llame al método `presentarse()` para cada uno.

Ejercicio 5: Clases Abstractas con Polimorfismo

Cree un sistema de pedidos de comercio electrónico con métodos de pago.

1. Defina una clase abstracta `Pago` con un método abstracto `procesarPago()`.
2. Cree subclases como `PagoConTarjetaDeCrédito`, `PagoConPaypal` y `PagoEnEfectivo`, Implementendo el método `procesarPago()`.
3. Escriba una función `finalizarCompra()` que acepte una lista de objetos `Pago` y recorra la lista, llamando a `procesarPago()` para cada uno.
4. Prueba tu Implementeción con varias instancias de métodos de pago.