**UNIVERSIDAD DE LAS FUERZAS ARMADAS “ESPE’’**

**NOMBRE:** Solange Villegas

**FECHA:** 06/30/2024

**NRC:** 14419

**INFORME 1**

**Herencia**

La herencia es un concepto fundamental en programación orientada a objetos que permite que una clase herede las propiedades y el comportamiento de otra clase. La clase de la que se hereda se denomina clase principal o superclase, mientras que la clase que realiza la herencia se denomina clase secundaria o subclase.

La herencia permite la reutilización de código y facilita la creación de una jerarquía de clases que comparten atributos y métodos comunes. La clase secundaria hereda todos los campos y métodos de la clase principal y también puede agregar nuevos campos y métodos o anular los heredados de la clase principal.

Tipos de herencia:

1. **Herencia única** : una clase secundaria hereda de una clase principal única.
2. **Herencia múltiple** : una clase secundaria hereda de varias clases primarias. (No es compatible con todos los lenguajes de programación)
3. **Herencia multinivel** : una clase secundaria hereda de una clase principal que a su vez hereda de otra clase principal.
4. **Herencia jerárquica** : una clase principal tiene varias clases secundarias que heredan de ella.
5. **Herencia híbrida** : una combinación de herencia múltiple y multinivel.

**Polimorfismo**

El polimorfismo es otro concepto esencial en programación orientada a objetos que permite tratar objetos de diferentes clases como objetos de una superclase común. Esto permite escribir código que pueda funcionar con objetos de diferentes clases sin conocer su tipo de clase específico.

Tipos de polimorfismo:

1. **Sobrecarga de métodos** : múltiples métodos con el mismo nombre pero diferentes parámetros.
2. **Anulación de métodos** : una clase secundaria proporciona una implementación específica de un método ya definido en su clase principal.
3. **Sobrecarga de operadores** : redefinir el comportamiento de los operadores para una clase.

Beneficios de la herencia y el polimorfismo:

1. **Reutilización de código** : la herencia permite la reutilización de código al permitir que las clases secundarias hereden atributos y métodos comunes de las clases principales.
2. **Mayor flexibilidad** : el polimorfismo permite una mayor flexibilidad en la programación al permitir que objetos de diferentes clases sean tratados como objetos de una superclase común.
3. **Mantenimiento más fácil** : la herencia y el polimorfismo facilitan la modificación y la ampliación del código existente al permitir realizar cambios en el nivel de la clase principal.

Ejemplos del mundo real:

1. **Jerarquía de clases de vehículos** : una clase de vehículo puede ser la clase principal, y las clases secundarias, como automóvil, camión y motocicleta, heredan de ella.
2. **Jerarquía de clases de formas** : una clase de formas puede ser la clase principal, y las clases secundarias como Círculo, Rectángulo y Triángulo heredan de ella.

En conclusión, la herencia y el polimorfismo son conceptos fundamentales en programación orientada a objetos que permiten la reutilización del código, la flexibilidad y un mantenimiento más sencillo. Son herramientas esenciales que cualquier desarrollador de software debe dominar y sus aplicaciones son numerosas en escenarios del mundo real.