**Herencia y polimorfismo ¿para qué sirven?**

**Introducción**

en este documento se busca responder a las preguntas ¿Qué es herencia y polimorfismo en programación? Y ¿para qué sirven? Para ello nos adentraremos en el concepto de ambos y para que son usados en la Programación Orientada a Objetos (POO) y los distintos problemas que podemos solucionar usando herencia y polimorfismo.

**¿Qué es la herencia en programación?**

La herencia es aquella propiedad en la cual podemos crear nuevas clases teniendo como bases creadas con anterioridad a este tipo de bases se les llamara “clase base”, estas nuevas clases heredaran las características y distintos comportamientos que ya tenían las clases ya existentes estas nuevas bases se denomina como “clase derivada”. Al momento de escribir la declaración de derivación de clases se debe incluir el nombre de la “clase base” desde la cual estamos haciendo nuestra clase derivada, el tipo de herencia (publica, privada o protegida) y “el especificador de acceso” (:), siguiendo el siguiente formato:

class nombre\_clase\_derivada :: tipo\_herencia nombre\_clase\_base

**Tipos de Herencia**

1. **Publica:** este tipo de herencia solo tiene acceso a los atributos públicos y protegidos de su clase base, permaneciendo los atributos públicos como públicos y los atributos protegidos como protegidos
2. **Privada:** en este tipo de herencia no tiene acceso a ninguno de los atributos de la clase base.
3. **Protegida:** en este tipo de herencia los atributos públicos y protegidos de la clase base se convierten en atributos protegidos de la clase derivada, y los miembros privados de la clase base se vuelven inaccesibles.
4. **Múltiple:** en este tipo de herencia la clase derivada hereda los atributos de mas de una clase base.

**Constructores**

Son elementos fundamentales en la POO cuando utilizamos herencias, ya con los constructores le damos a la herencia los atributos que componen la clase base. Después de activar el constructor de la clase base, se debe activar el constructor de la clase derivada.

**Destructores**

En la POO se usan los destructores cuando un constructor de la clase derivada ha asignado un espacio en la memoria que debe ser liberado.

**¿Qué es el polimorfismo en programación?**

El polimorfismo y la herencia son dos elementos muy comunes dentro de la Programación POO, consiste en crear distintas clases derivadas con métodos iguales pero distintos entre ellos, esto quiere decir que un objeto puede responder de una manera distinta a un método, a como respondería un objeto distinto a ese mismo método. Los tipos de respuesta de distintos objetos se califican de la siguiente manera:

1. Polimorfismo sin ligadura dinámica: esto significa que dos objetos responden de la misma manera ante un mismo método.
2. Polimorfismo con ligadura dinámica: mediante esta manera podemos hacer que un objeto en especifico responda de una manera específica ante un método.

**Uso del polimorfismo**

El polimorfismo nos permite utilizar un mismo método para distintos objetos, para esto debemos seguir las siguientes reglas:

1. Crear una jerarquía de clases junto con las distintas operaciones importantes definidas por las funciones de la clase base
2. Las implementaciones especificas se deben realizar dentro de las clases derivadas.
3. Las instancias de las clases derivadas se usan a través de una referencia o puntero.

**Ventajas del polimorfismo**

El polimorfismo nos ayuda a hacer un código de programación mas libre y flexible, además de ayudarnos con acciones de la POO tales como:

**Especialización de clases derivadas**: este es el uso más común del polimorfismo, esta clase de polimorfismo aumenta la eficiencia de la clase derivada, mientras conserva su alta flexibilidad y permite un medio uniforme para manejar distintos objetos.

**Estructuras de datos Heterogéneos:** usando polimorfismo podemos crear y manejar de una manera sencilla estructuras de datos heterogéneos, que son sencillos de diseñar, sin perder la comprobación de los elementos utilizados.

**Gestión de una jerarquía de clases:** las jerarquías de clases son colecciones de clases, con una buena estructuración que se puede entender fácilmente.

**Conclusiones**

La herencia y el polimorfismo son partes fundamentales de la POO, ya que con estos se pueden realizar distintos códigos de programación que involucren más de dos clases de objetos los cuales pueden responder de distintas maneras ante un mismo método o ante varios métodos, los cuales pueden estar incluidos en todos los objetos o en algunos de ellos.

**Bibliografía**

Joyanes Aguilar, L., Zahonero Martínez, I.(2014). Programación en C/C++ JAVA y UML. McGraw-Hill. <https://www-ebooks7-24-com.ezproxy.umng.edu.co/?il=291>