Actividad unidad 5. Conceptos importantes de la programación orientada a objetos.

Angélica Hernández Moreno

➔ Encapsulamiento y ocultación de datos.

Implica reunir en una misma estructura, llamada clase, tanto los datos (atributos) como los métodos (comportamientos) asociados a un objeto. También se establece un control sobre el acceso a estos elementos, de forma que solo se permite su uso mediante interacciones definidas y seguras desde fuera de la clase.

Para lograrlo, se puede anteponer doble guión bajo (`__`) al nombre de un atributo o método. Esto indica a Python que se trata de un elemento 'privado', lo que impide que sea accedido directamente desde fuera de la clase.

➔ Herencia

La herencia permite crear una clase hija que toma como base una clase padre, adquiriendo sus atributos y métodos. Esta clase hija también tiene la posibilidad de sobrescribir o modificar los elementos heredados o agregar otros nuevos. Para establecer la herencia, basta con indicar la clase padre entre paréntesis al definir la clase hija.

➔ Reutilización

En Python, la reutilización de código se realiza mediante el uso de clases. Crear una clase implica definir un objeto con atributos y métodos que dan organización y funcionalidad al programa. Gracias a las clases, es posible reutilizar código ya escrito, reduciendo repeticiones y minimizando errores, ya que se pueden crear múltiples objetos con comportamientos similares a partir de una misma estructura.

➔ Polimorfismo

Se refiere a la capacidad de los objetos para adoptar distintas formas, lo que significa que instancias de diferentes clases pueden ser manipuladas mediante una misma interfaz, aunque respondan de manera diferente según su implementación. En

Python, no es obligatorio que los objetos compartan una interfaz formal; basta con que posean los métodos que se desea invocar.

➔ **Sobrecarga de operadores**

La sobrecarga de operadores en Python permite modificar el comportamiento de operadores (por ejemplo: +, -, * o /) de modo que se adapten a clases personalizadas. Esto se logra mediante la implementación de métodos especiales dentro de la clase, los cuales están definidos con doble guión bajo al inicio y al final. Por ejemplo, para redefinir el operador "+", es necesario crear el método " __add__".

➔ **Sobrecarga de funciones**

En lenguajes como Java o C++, se permite definir varias funciones con el mismo nombre siempre que se diferencien por la cantidad o el tipo de sus parámetros. Esta técnica se conoce como sobrecarga de funciones o de métodos cuando se trata de clases. En Python, esto no es posible de forma directa, ya que el lenguaje decide qué función ejecutar según los tipos de datos de los argumentos, y Python utiliza un tipado dinámico, lo que significa que no se declara el tipo de dato al definir una variable. Sin embargo, existen diferentes estrategias para lograr un comportamiento similar al de la sobrecarga de funciones. Un ejemplo es la función incorporada "range()". Aunque Python no cuenta con soporte nativo para sobrecarga ni despacho múltiple, es posible replicar este comportamiento por otros medios.

➔ **Mensaje**

Los mensajes representan un elemento esencial, ya que permiten que los objetos se comuniquen entre sí dentro de un sistema. Básicamente, un mensaje es una instrucción o petición que un objeto envía a otro para solicitar una acción o información. Esta forma de interacción facilita la creación de sistemas complejos y flexibles, ya que promueve la colaboración y distribución de tareas entre objetos. Cada objeto puede responder a los mensajes recibidos de manera particular, ejecutando sus propios métodos y afectando así el comportamiento general del programa.

Referencias

- El Libro De Python. (s. f.). *Polimorfismo en programación*. Recuperado el 13 de julio de 2025, de El Libro De Python: https://ellibrodepython.com/polimorfismo-en-programacion
- Apuntes.de. (s. f.). *Creación y uso de clases en Python: objetos y reutilización de código*. Recuperado el 13 de julio de 2025, de Apuntes.de: https://apuntes.de/python/creacion-y-uso-de-clases-en-python-objetos-y-reutilizacion-de-codigo/#gsc.tab=0
- Cursa.app. (s. f.). *Programación orientada a objetos en Python: sobrecarga de operadores*. Recuperado el 13 de julio de 2025, de https://cursa.app/es/pagina/programacion-orientada-a-objetos-en-python-sobrecarga-de-operadores-en-python
- Recursos Python. (9 de agosto de 2018). *Sobrecarga de funciones o despacho múltiple en Python*. Recuperado el 13 de julio de 2025, de Recursos Python: https://recursospython.com/guias-y-manuales/sobrecarga-de-funciones-o-despacho-multiple/
- Gonzalez, M. (2024). *Mensajes en programación orientada a objetos*. Recuperado el 13 de julio de 2025, de Programacion365: https://programacion365.com/mensajes-en-programacion-orientada-a-objetos/