TIC

Módulo 1 – Unidad 2 –
Lección 1

Actividad 7

Programación Orientada a Objetos (POO)

TALENTO TECH

Universidad Tecnológica de Pereira | faceIT

Programación Orientada a Objetos (POO)

Conceptos fundamentales de POO: clases y objetos. Métodos y atributos en el contexto de la programación orientada a objetos.

La Programación Orientada a Objetos (POO) es un paradigma de programación que se basa en el concepto de "objetos".

Objeto: un objeto es una instancia concreta de una clase.
Ejemplo: si tienes una clase "Perro", un objeto sería una instancia específica de perro, como "miPerro".

Clase: una clase es un modelo o plano que define las características y comportamientos comunes de un grupo de objetos.
Ejemplo: la clase "Perro" puede tener atributos como "nombre" y métodos como "ladrar".

Programación Orientada a Objetos (POO)

Atributos: son las características o propiedades de un objeto que lo describen. Ejemplo: en la clase "Persona", los atributos pueden ser "nombre", "edad" y "altura".

Métodos: son funciones asociadas a una clase y describen el comportamiento de los objetos de esa clase. Ejemplo: en la clase "Coche", un método puede ser "arrancar".

Encapsulamiento: es el principio que consiste en ocultar los detalles internos de la implementación de un objeto y exponer solo lo necesario. Ejemplo: atributos privados que solo son accesibles mediante métodos públicos.

Programación Orientada a Objetos (POO)

Herencia: es un mecanismo que permite que una clase herede atributos y métodos de otra clase. Ejemplo: una clase "Estudiante" que hereda de la clase "Persona".

Polimorfismo: permite que un objeto pueda tomar múltiples formas, es decir, que un mismo método pueda comportarse de manera diferente en distintas clases. Ejemplo: un método "hacerSonido" que se comporte de manera diferente en las clases "Perro" y "Gato".

Estos conceptos forman la base de la Programación Orientada a Objetos y proporcionan una estructura eficiente y organizada para el desarrollo de software.

La Programación Orientada a Objetos (POO) en Python

Se basa en la creación y manipulación de objetos.

Clases y Objetos:

Clase:

es una plantilla que define las propiedades y comportamientos comunes de un conjunto de objetos. Objeto: es una instancia concreta de una clase. Cada objeto tiene sus propios atributos y puede realizar acciones específicas.

```python
class Perro:
    def __init__(self, nombre, edad):
        self.nombre = nombre
        self.edad = edad

    def ladrar(self):
        print(f"{self.nombre} está ladrando.")

mi_perro = Perro("Buddy", 3)
mi_perro.ladrar()
```

La Programación Orientada a Objetos (POO) en Python

Atributos:

Son características de un objeto y se definen en la clase, pueden ser accesibles desde fuera de la clase (public), o se pueden ocultar (private) mediante convenciones.

```python
class Persona:
    def __init__(self, nombre, edad):
        self.nombre = nombre  # Atributo público
        self.__edad = edad    # Atributo privado

persona = Persona("Juan", 25)
print(persona.nombre)  # Acceso a atributo público
print(persona.__edad)  # Error: No se puede acceder a atri
```

La Programación Orientad a Objetos (POO) en Python

Métodos:

Son funciones definidas dentro de una clase y representan el comportamiento de los objetos, pueden tener acceso a los atributos de la clase mediante self.

```python
class Rectangulo:
    def __init__(self, base, altura):
        self.base = base
        self.altura = altura

    def calcular_area(self):
        return self.base * self.altura

rectangulo = Rectangulo(5, 10)
print(rectangulo.calcular_area())
```

La Programación Orientad a Objetos (POO) en Python

Herencia:

Permite crear una nueva clase basada en una clase existente, heredando sus atributos y métodos. la nueva clase se llama subclase, y la clase existente se llama superclase.

```python
class Animal:
    def __init__(self, nombre):
        self.nombre = nombre

class Perro(Animal):
    def ladrar(self):
        print(f"{self.nombre} está ladrando.")

perro = Perro("Buddy")
print(perro.nombre)
perro.ladrar()
```

La Programación Orientada a Objetos (POO) en Python

Polimorfismo:

Permite que objetos de diferentes clases respondan al mismo método, facilita el uso de una interfaz común para objetos de distintas clases.

```python
class Gato(Animal):
    def maullar(self):
        print(f"{self.nombre} está maullando.")

gato = Gato("Whiskers")
animales = [perro, gato]

for animal in animales:
    print(animal.nombre)
```

La POO en Python facilita la creación de programas más estructurados y reutilizables al organizar la lógica en clases y objetos.

Ejercicios relacionados con Programación Orientada a Objetos (POO) en Python

- Creación de Clases y Objetos
- Métodos en Clases
- Herencia
- Polimorfism
- ***Encapsulamiento***

Estos ejercicios proporcionan práctica en la creación y manipulación de clases, el uso de herencia, la implementación de métodos y el manejo de encapsulamiento en Python.

Ejercicio Encapsulamiento

- Modifica la clase Coche para que el atributo anio sea privado.
- Agrega métodos de acceso (getter y setter) para el atributo anio.

Ejercicio Métodos en Clases

- Modifica la clase Coche para incluir un método arrancar que imprima un mensaje indicando que el coche
- está arrancado.

  Llama al método arrancar para uno de los coches creados en el Ejercicio 1.

Ejercicio Herencia

- Crea una clase Deportivo que herede de la clase Coche.
- Añade un atributo adicional a la clase Deportivo como velocidad_maxima.
- Crea un objeto de la clase Deportivo y llama al método arrancar.

Ejercicio Creación de Clases y Objetos

- Crea una clase Coche con atributos como modelo, color y anio.
- Crea dos objetos de la clase Coche con diferentes valores para los atributos.
- Imprime los detalles de ambos coches.

Ejercicio Polimor smo

- Define un método llamado desplazarse en ambas clases (Coche y Deportivo) que imprima un mensaje genérico como "se desplaza por la carretera".
- Crea una lista que contenga objetos de ambas clases y, mediante un bucle, llama al método desplazarse para cada objeto.