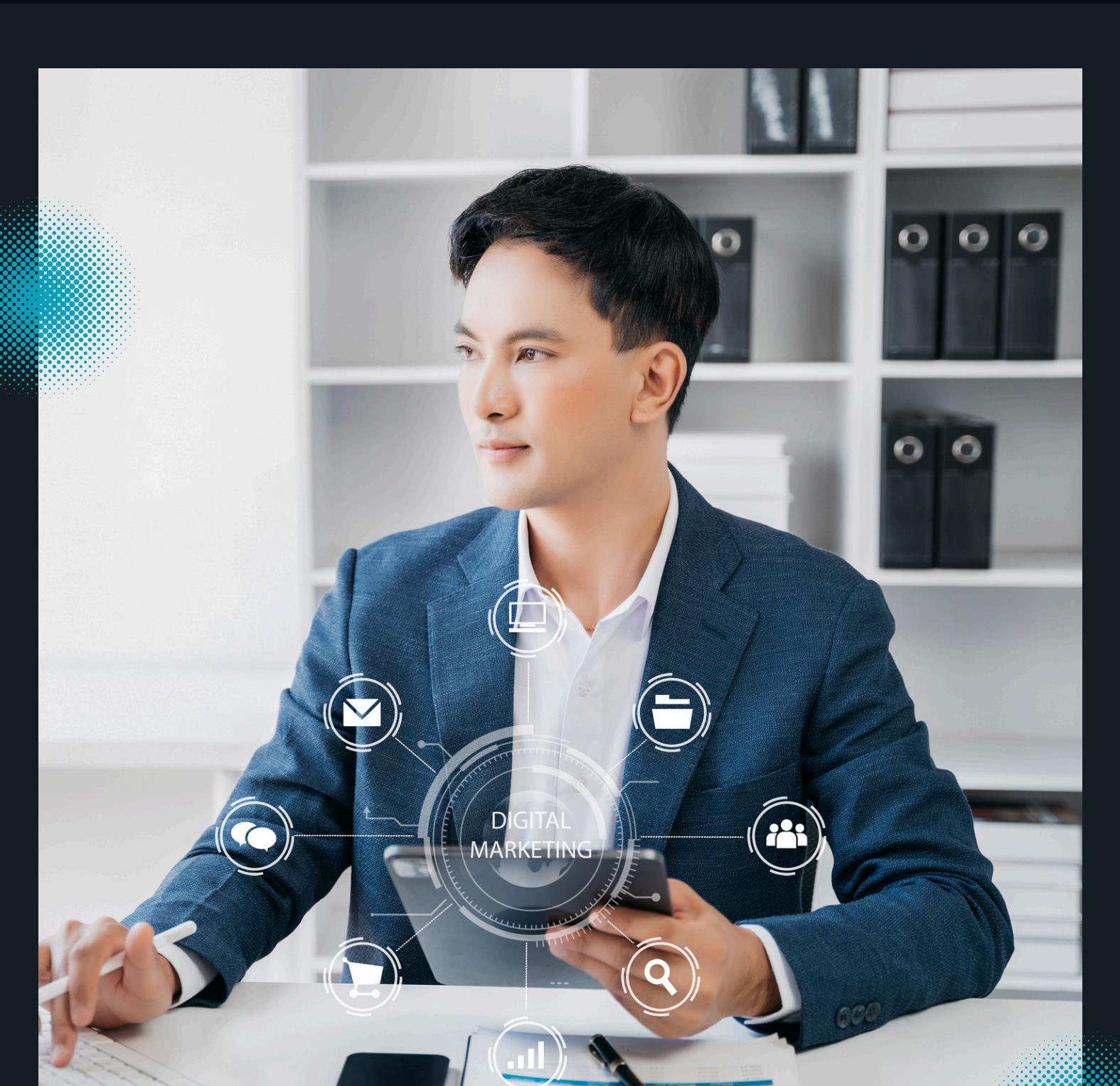


POLIMORFISMO

MARIA PAZ GALEANO
MARIANA SUAREZ
SEBASTIAN FIGUEROA
JOSE CELIS

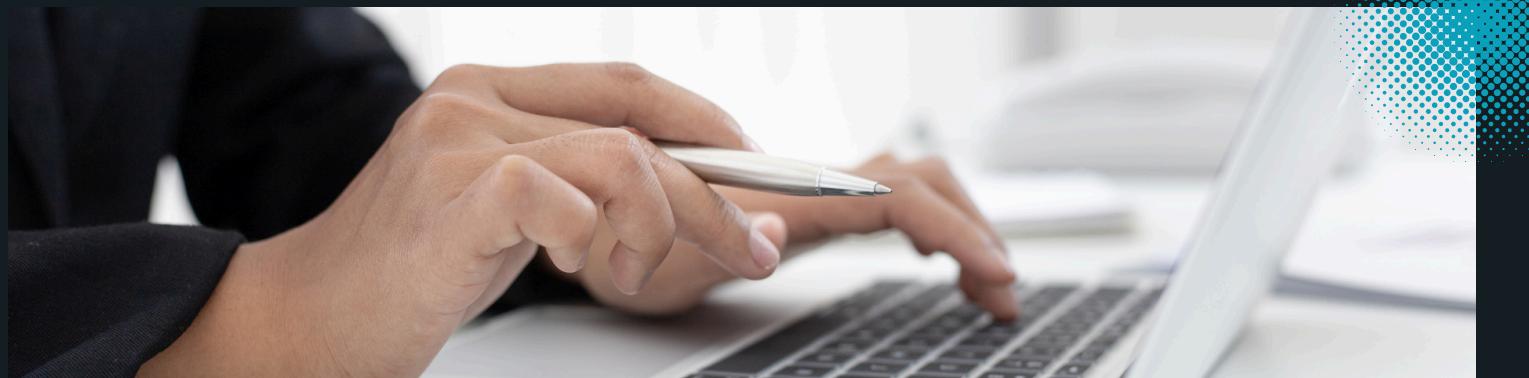


CONCEPTO

hace referencia a la capacidad de un objeto de tomar múltiples formas, lo que permite que una misma operación (como un método o función) pueda ejecutarse de diferentes maneras según el tipo de objeto que la invoque.

EJEMPLO MINECRAFT

Imagina que tienes una acción común como "atacar", pero según el personaje del juego (zombi, humano o esqueleto), esa acción se lleva a cabo de forma diferente.



CLASES Y OBJETOS

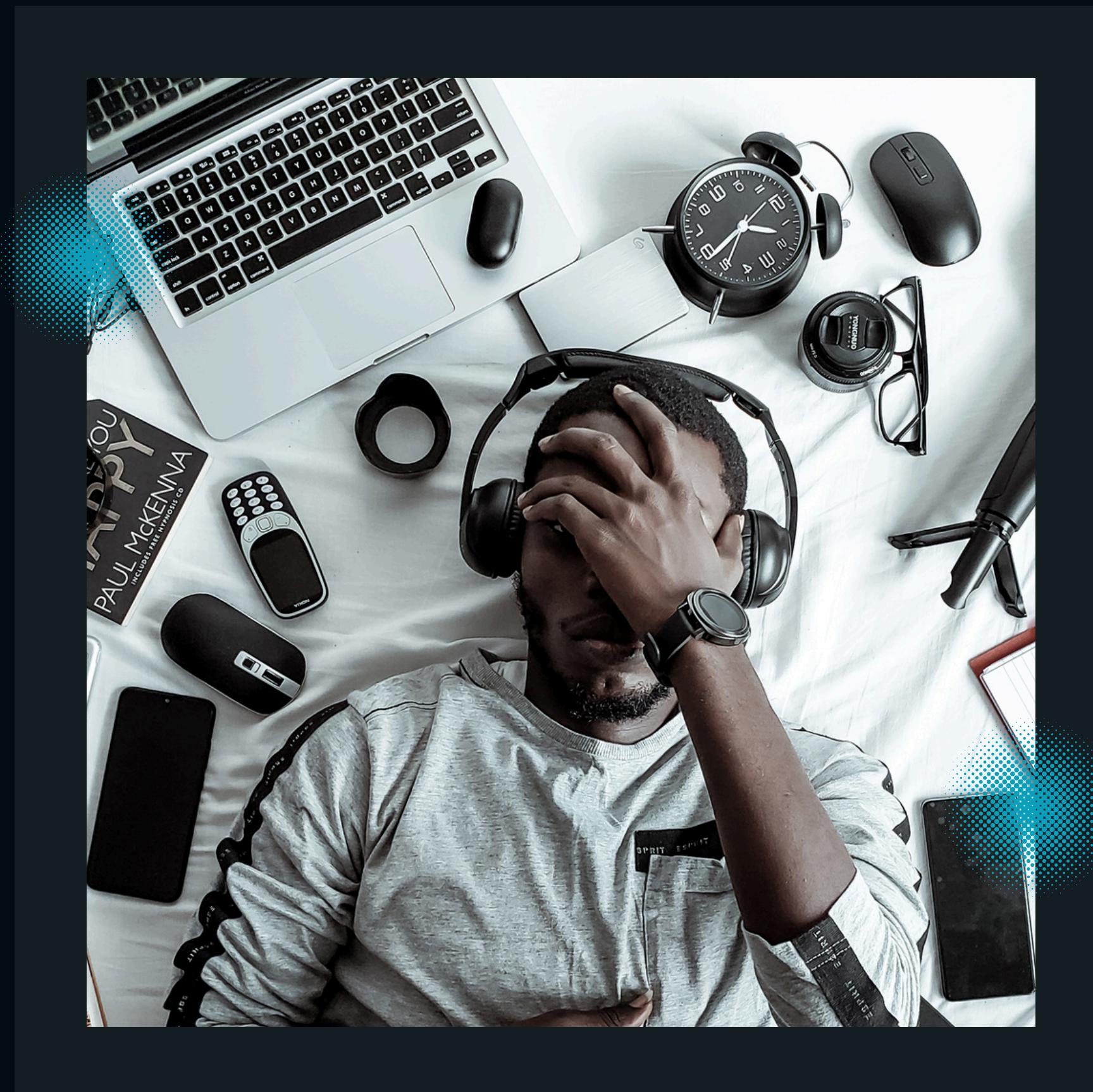
En POO, el concepto de clase se refiere a un plano o molde que define las características y comportamientos de los objetos.

Clase

Es un conjunto de atributos (variables) y métodos (funciones) que definen las características y acciones posibles de los objetos que pertenecen a esa clase.

Objeto

Es una instancia de una clase. Cuando creas un objeto, estás construyendo una entidad que sigue el plano definido por su clase.



POLIMORFISMO EN MÉTODOS

El polimorfismo se manifiesta cuando diferentes clases tienen un método con el mismo nombre, pero con implementaciones distintas. Esto permite que puedas interactuar con distintos tipos de objetos de manera uniforme, pero que cada uno reaccione de manera diferente según su clase.

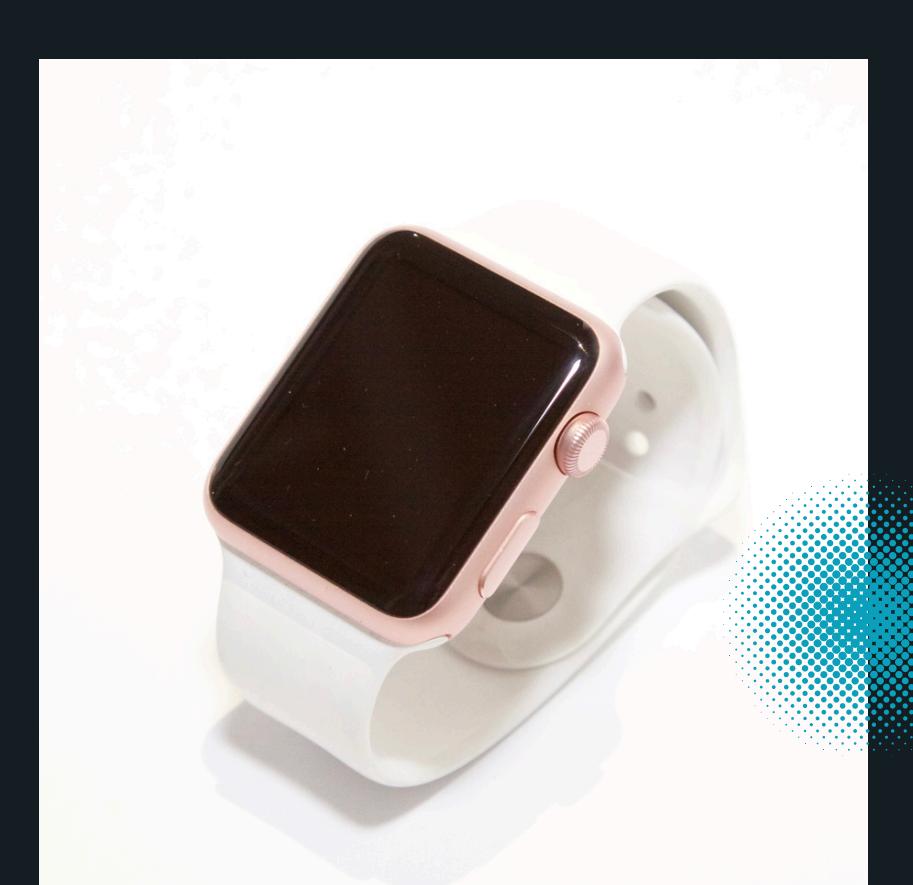
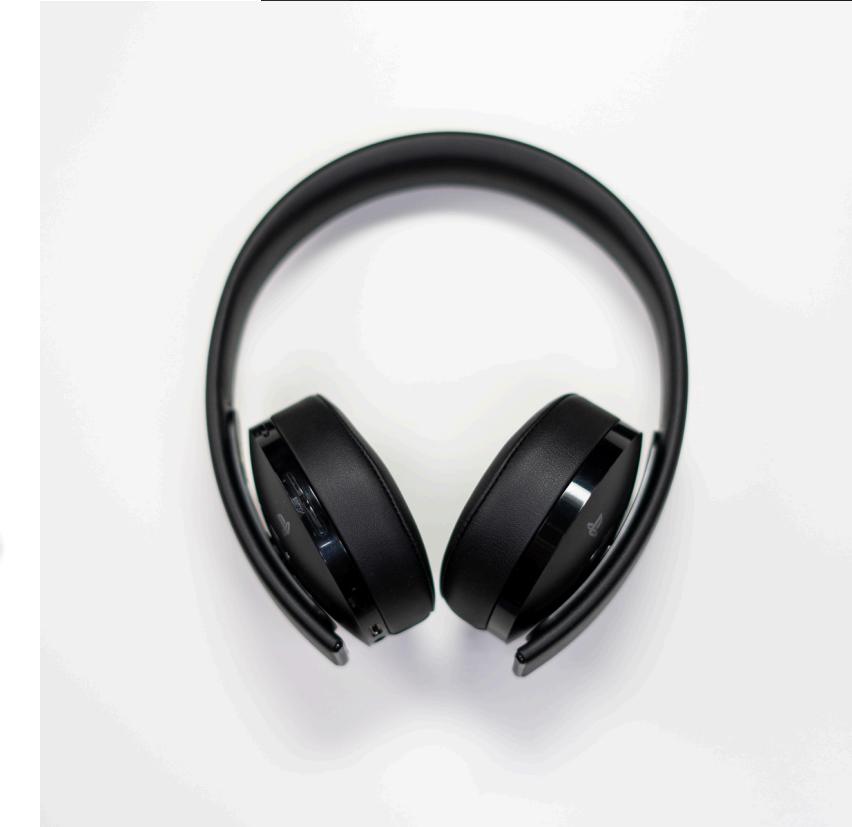
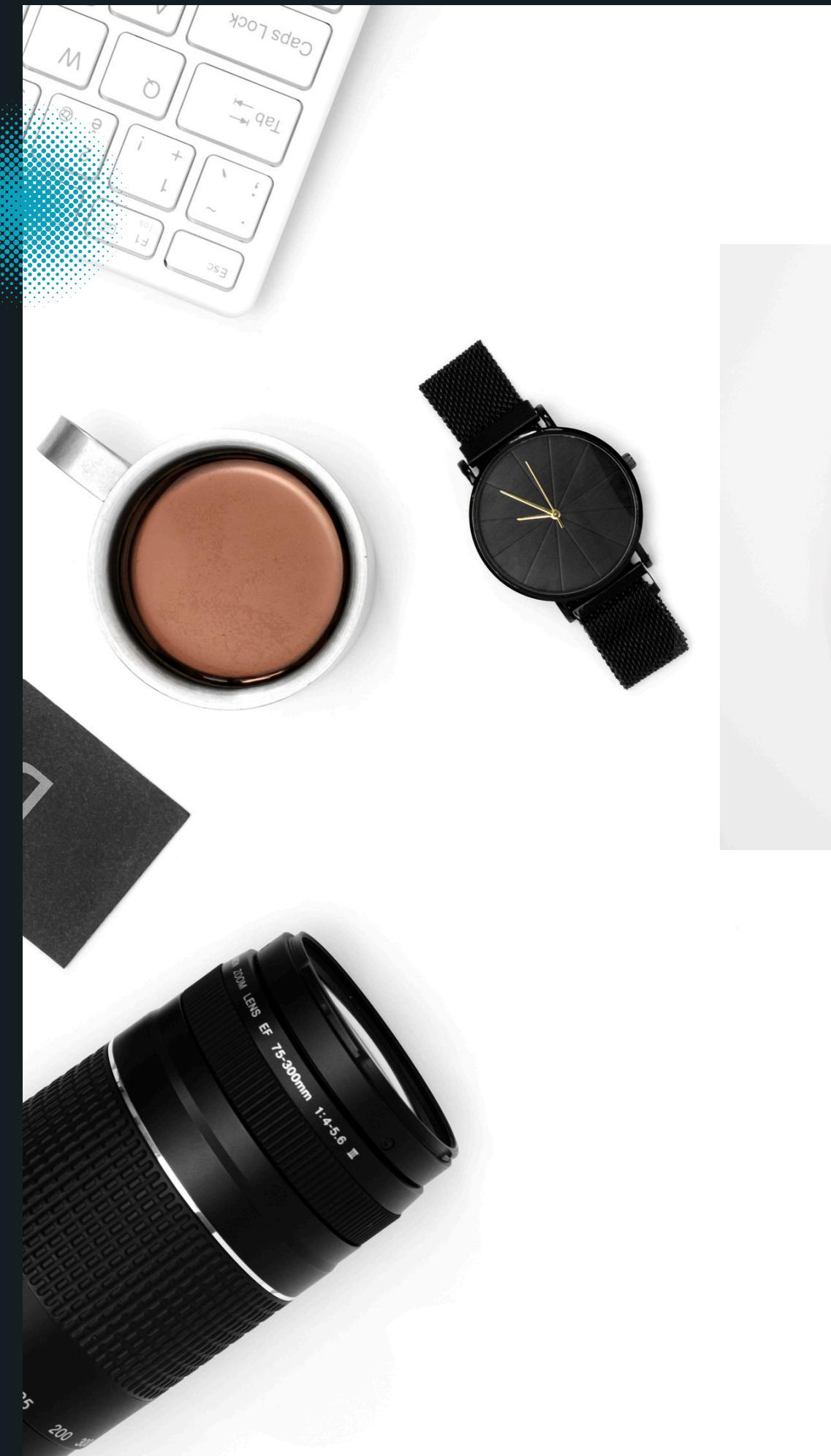
Sobreescritura de métodos:

Las subclases pueden sobreescibir un método de la clase base para adaptarlo a sus necesidades. Es aquí donde el polimorfismo entra en juego, permitiendo que las subclases definan su propio comportamiento.

Relación con la herencia

La herencia proporciona la estructura necesaria para que el polimorfismo funcione, permitiendo que las subclases implementen sus propios comportamientos mientras comparten una interfaz común. Esta relación es fundamental para diseñar sistemas de software más robustos y adaptables.

VENTAJAS



- Reusabilidad: El código escrito para la clase base puede ser reutilizado por muchas subclases.
- Flexibilidad: Se pueden crear nuevas subclases y adaptar su comportamiento sin cambiar el código principal.
- Simplicidad: Puedes escribir funciones que trabajen con objetos de múltiples clases sin preocuparte por los detalles de la implementación de cada una.

EJEMPLO PYTHON MINECRAFT

Explicación:

1. Clases de Entidades:

- Zombi y Esqueleto atacan como antes, de manera predecible.

2. Humano con o sin arma:

- El humano tiene un atributo `tiene_arma` que indica si posee un arma (True) o no (False). Dependiendo de esto, el método `atacar()` devuelve una respuesta diferente: ataca con un arma o con las manos.

3. Función polimórfica:

- La función `entidad_ataca(entidad)` sigue llamando al método `atacar()` en cualquier tipo de entidad, demostrando el polimorfismo de manera sencilla.

Este código muestra cómo el humano decide atacar dependiendo de si tiene un arma o no, manteniendo el concepto de polimorfismo claro y simple.

```
# Clase base
class Entidad:
    def atacar(self):
        pass # Método base que será sobrescrito por las subclases

# Subclase Zombi
class Zombi(Entidad):
    def atacar(self):
        return "El zombi ataca mordiendo."

# Subclase Esqueleto
class Esqueleto(Entidad):
    def atacar(self):
        return "El esqueleto ataca disparando flechas."

# Subclase Humano
class Humano(Entidad):
    def __init__(self, tiene_arma):
        self.tiene_arma = tiene_arma # True si tiene un arma, False si no

    def atacar(self):
        if self.tiene_arma:
            return "El humano ataca con un arma."
        else:
            return "El humano ataca con las manos."

# Función que demuestra el polimorfismo
def entidad_ataca(entidad):
    print(entidad.atacar())

# Crear instancias de Zombi, Esqueleto y Humano
zombi = Zombi()
esqueleto = Esqueleto()
humano_con_arma = Humano(tiene_arma=True) # El humano tiene un arma
humano_sin_arma = Humano(tiene_arma=False) # El humano no tiene un arma

# Llamar la función polimórfica
entidad_ataca(zombi)                      # El zombi ataca mordiendo.
entidad_ataca(esqueleto)                   # El esqueleto ataca disparando flechas.
entidad_ataca(humano_con_arma)             # El humano ataca con un arma.
entidad_ataca(humano_sin_arma)             # El humano ataca con las manos.
```



El zombi ataca mordiendo.
El esqueleto ataca disparando flechas.
El humano ataca con un arma.

THANK YOU