# Preguntas

# 1

Las clases sirven para empaquetar funcionalidades y crear objetos sin tener que repetir código. Si tuviéramos una receta de distintos pasteles, cada uno tendría sus propias características, como por ejemplo sabor, pero tendría elementos comunes. Así, a partir de una misma receta podríamos crear distintos pasteles, aunque cambiando ingredientes. Pasando a código esto sería algo así:

class Pastel:

def \_\_init\_\_(self, ingrediente):

self.ingrediente = ingrediente

Con esto tenemos la receta, vamos a crear los objetos/ingredientes:

pastel\_chocolate = Pastel("chocolate")

pastel\_fresa = Pastel("fresa")

Solo con esto me ahorro repetir código, y además, si quiero cambiar algo, solo tengo que cambiarlo en la propia clase.

# 2

El método que se ejecuta al crear una clase es el método \_\_init\_\_(), también llamado constructor. Este método se ejecuta automáticamente cada vez que se crea un objeto de la clase. Su sintaxis es como sigue:

class NombreClase:

def \_\_init\_\_(self, parámetros):

self.parámetros = parámetros

# 3

# Los verbos API, también llamados verbos HTTP (aunque no siempre son verbos), sirven para indicar a la API qué datos queremos obtener, y cómo queremos obtenerlos. Los más importantes son GET, POST y PUT. El método GET solicita una representación de un recurso específico, el método POST crea un nuevo recurso Y el método PUT actualiza un recurso existente.

# 4

# MongoDB es una base NoSQL (Not Only SQL) que almacena datos fuera de las estructuras relacionales tradicionales, y almacena estructuras de datos usando pares key-value y colecciones. En lugar de la típica estructura tabular de una base de datos relacional, las bases de datos NoSQL alojan datos dentro de una estructura de datos, como un documento JSON. Dado que este diseño de base de datos no relacional no requiere un esquema, ofrece una rápida escalabilidad para gestionar grandes conjuntos de datos normalmente no estructurados.

# 5

# Una API (Application Programming Interface) es una interfaz de programación que define cómo un conjunto de aplicaciones y servicios intercambian información. Una API es una interfaz que permite a los desarrolladores de software comunicarse entre sí y con los sistemas externos. Es una especie de "puente" entre el código del desarrollador y el código de los sistemas externos. Por ejemplo, un programa podría comunicarse con una API de un instituto meteorológico para obtener información sobre el clima actual.

# 6

# Postman es una interfaz gráfica para crear y probar APIs, así como para enviar peticiones HTTP a servidores web.

# 7

# El polimorfismo es, en programación orientada a objetos, la capacidad de un objeto de tener diferentes formas o representaciones dependiendo del contexto. En código podríamos tener lo siguiente:

class Gato():

def sonido(self):

return "miau"

class Perro():

def sonido(self):

return "guau"

gato = Gato()

perro = Perro()

print(perro.sonido()) # imprime guau porque hace referencia al objeto perro

print(gato.sonido()) # imprime miau porque hace referencia al objeto gato

# 8

# Un método dunder es un método especial de Python que se utiliza para implementar operaciones en una clase y se llaman automáticamente dentro de esta. Se llaman así porque tienen dos guiones bajos al inicio y al final de su nombre. \_\_init\_\_ es un ejemplo.

# 9

# Un decorador es una función que toma una función como argumento y devuelve una función con más funcionalidades sin modificar lo ya existente.

def brillitos(funcion):

funcion()

def saludar():

print("Hola")

brillitos(saludar)