1.1 Para definir una clase en Python, se utiliza la siguiente sintaxis:

class NombreClase:

# Definición de atributos y métodos de la clase

Pass

Se usa la palabra clave class seguida del nombre de la clase, y luego se define el contenido de la clase mediante la indentación. Los atributos y métodos de la clase se definen dentro del cuerpo de la clase.

1.2 En Python, un atributo de clase es un atributo que se comparte entre todas las instancias de la clase. Se define dentro de la clase, pero fuera de cualquier método. Los atributos de clase se pueden acceder a través de la clase misma o a través de las instancias de la clase. Por otro lado, un atributo de instancia es un atributo específico de cada instancia de la clase. Cada instancia puede tener sus propios valores para los atributos de instancia.

La diferencia clave entre un atributo de clase y uno de instancia es que los atributos de clase se comparten entre todas las instancias y se pueden acceder a través de la clase, mientras que los atributos de instancia son específicos de cada instancia y solo se pueden acceder a través de la instancia.

1.3 En Python, el constructor se define mediante el método especial \_\_init\_\_() en una clase. El constructor se llama automáticamente cuando se crea una nueva instancia de la clase y se utiliza para inicializar los atributos de la instancia. La sintaxis para definir un constructor es la siguiente:

class NombreClase:

def \_\_init\_\_(self, parametro1, parametro2):

self.atributo1 = parametro1

self.atributo2 = parametro2

Dentro del constructor, se pueden asignar los valores de los parámetros a los atributos de instancia utilizando self.

1.4 Un decorador en Python es una función especial que se utiliza para modificar el comportamiento de una función o método existente. Los decoradores se colocan justo encima de la definición de la función o método y se indican con el símbolo @. Los decoradores se utilizan para agregar funcionalidad adicional a una función o método sin modificar su código original.

Por ejemplo, el decorador @staticmethod se utiliza para definir un método estático, el decorador @classmethod se utiliza para definir un método de clase y el decorador @property se utiliza para definir un método getter o setter.

1.5 Un método de instancia es un método que opera en una instancia específica de una clase y tiene acceso a los atributos de esa instancia a través de self. Un método de clase es un método que opera en la clase misma en lugar de en una instancia específica. Tiene acceso a los atributos de clase a través de cls y se define utilizando el decorador @classmethod. Un método estático es un método que no tiene acceso a los atributos de instancia ni a los atributos de clase. Se define utilizando el decorador @staticmethod y no recibe automáticamente self o cls.

1.6 Ejemplo de atributos y métodos en Python:

class MiClase:

atributo\_clase = "Atributo de clase"

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

self.atributo\_instancia = parametro

def metodo\_instancia(self):

print("Método de instancia")

@classmethod

def metodo\_clase(cls):

print("Método de clase")

@staticmethod

def metodo\_estatico():