Taller de Programación Orientada a Objetos (POO) en Python
Objetivo: Reforzar los conceptos de la Programación Orientada a Objetos (POO) en Python a través de preguntas teóricas y ejercicios prácticos.

Parte 1: Preguntas Teóricas (Conceptos Clave)

1. **¿Qué es la Programación Orientada a Objetos (POO) y por qué es útil?**
○ Explica con tus propias palabras qué significa "programar con objetos".

El programar con objetos es recrear situación reales o conceptos,las cuales le damos comportamientos y propiedades para que interactúen entre sí en el programa(código).

○ Menciona una ventaja de usar POO en lugar de escribir todo el código en una secuencia.

Una de sus ventajas es la optimización del código al programar, ahorrándonos líneas de código.

2. **Diferencia entre Clase y Objeto:**
○ Imagina que vas a construir una casa. ¿Qué sería la "clase" y qué sería el "objeto" en este ejemplo?

La clase para la casa sería las bases en su construcción y los objetos los muros que se sostienen a partir de la base.

3. **¿Qué son los Atributos y los Métodos en una clase?**
○ Si tienes una clase Perro, ¿qué atributos podría tener? ¿Qué métodos podría realizar?

Los Atributos son propiedades y los métodos son comportamientos

Los atributos que puede tener la clase perro son “la raza”,”el color”,”nombre”.
Y los métodos pueden ser el “ladrar”,”correr” o “comer”.

4. **¿Cuál es la función del método especial __init__ en Python?**
○ ¿Cuándo se llama este método y para qué se utiliza?

El método __init__ en Python le da inicio a atributos de objetos con valores iniciales.

5. **Explica el concepto de Herencia en POO.**
○ Da un ejemplo sencillo de cómo una clase puede "heredar" características de otra.

La Herencia en POO permite darle atributos y métodos de una clase Padre a una clase Hija.

Ejemplo:

```
class Animal:
    def comer(self):
        print("El animal come")

class Perro(Animal):
    def ladrar(self):
        print("Guau")
```

En este caso el perro hereda de la clase animal el método comer.

6. **¿Qué significa Encapsulamiento y cómo se logra en Python?**
○ ¿Por qué es importante "proteger" los datos de un objeto?
○ Menciona cómo se indican los atributos "privados" en Python.

El encapsulamiento significa proteger los datos de un objeto y que este no tenga cambios indebidos.

El protegerlos evita errores y que se modifique de forma ordenada.

Los atributos Privados se indican con dos guiones abajo : “__”

7. **Define Polimorfismo con un ejemplo.**
○ Piensa en diferentes animales que hacen un "sonido". ¿Cómo se relaciona esto con el polimorfismo?

Estos se relacionan con un mismo método en este caso el “sonido”, aunque tendrán diferentes sonidos estos pueden ser sus “comportamientos”

8. **¿Qué es la Abstracción en POO?**
○ ¿Por qué es útil ocultar los detalles complejos y mostrar solo lo esencial?
○ ¿Puedes crear un objeto directamente de una "clase abstracta"? ¿Por qué sí o por qué no?

La Abstracción en POO es el solo mostrar lo importante y el ocultar todo el proceso interno.

Para facilitar el uso del código y necesidad de entender su funcionamiento interno

No se puede porque estarían incompletas y necesitaría de otras clases llamen sus métodos.

9. **¿Qué son los Métodos Mágicos (Dunder Methods) en Python?**
○ Menciona un ejemplo de un método mágico y para qué se utiliza.

Los métodos Mágicos en Python permiten modificar comportamientos internos, en este caso se escriben con doble guion bajo en ambos lados “__method__”

Un ejemplo de método mágico es el “__str__” el cual define como se muestra un objeto al usar print().