

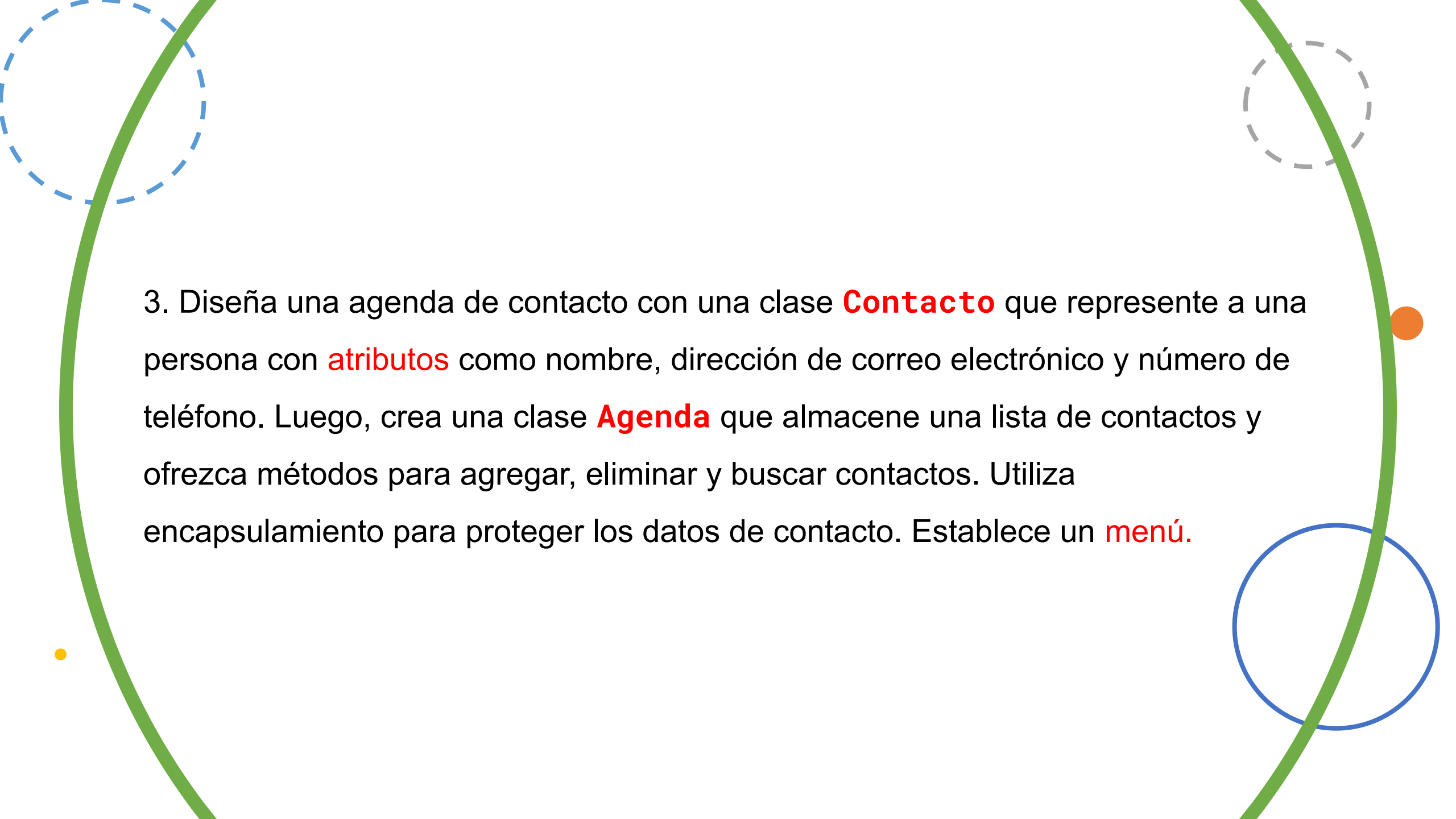


Exercicios PYTHON

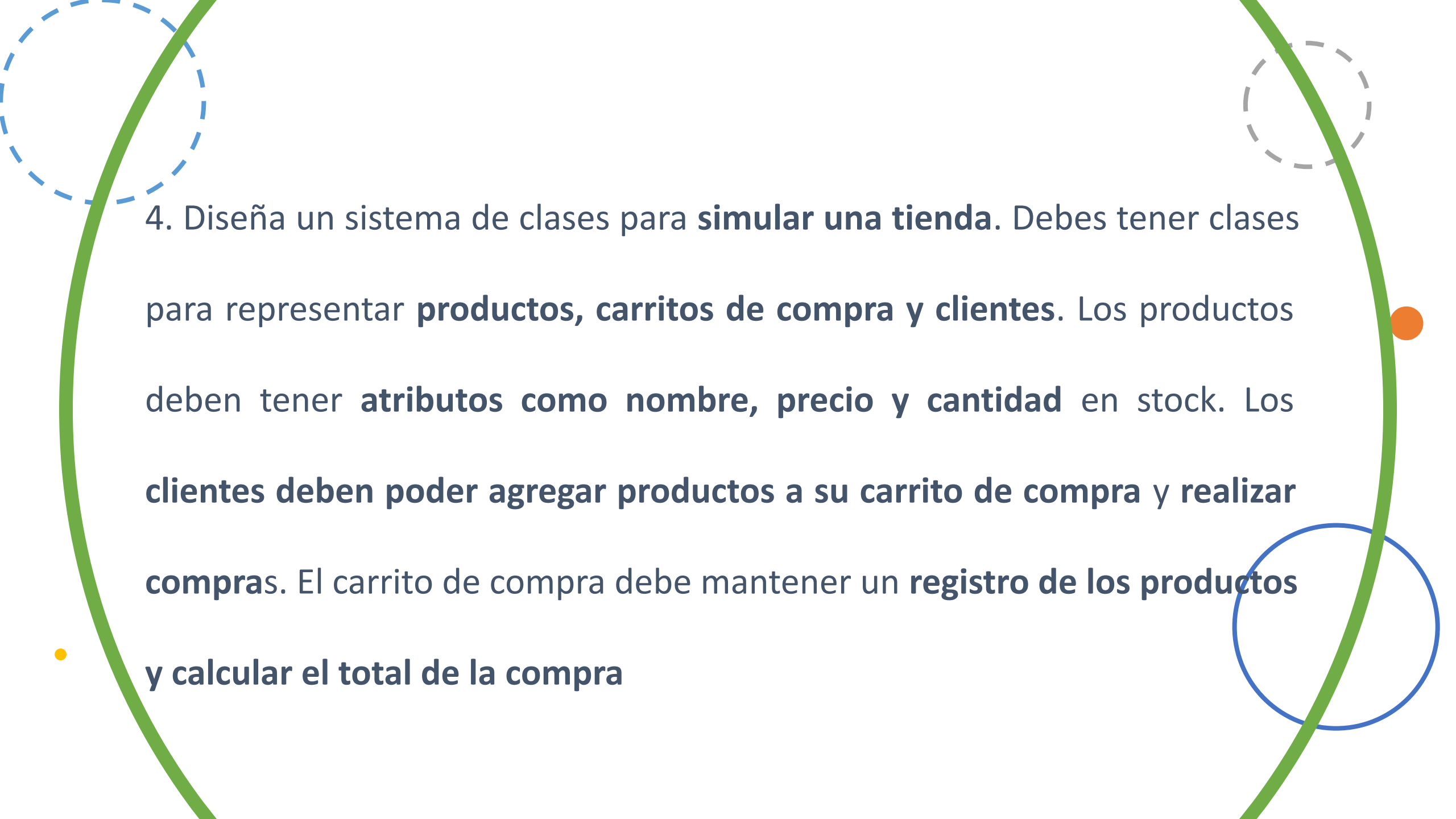
Conceptos Básicos de POO en Python

1. Crear dos clases, **Libro** y **Autor**, que estén relacionadas. La clase Libro debe tener un atributo autor, que sea una instancia de la clase Autor.

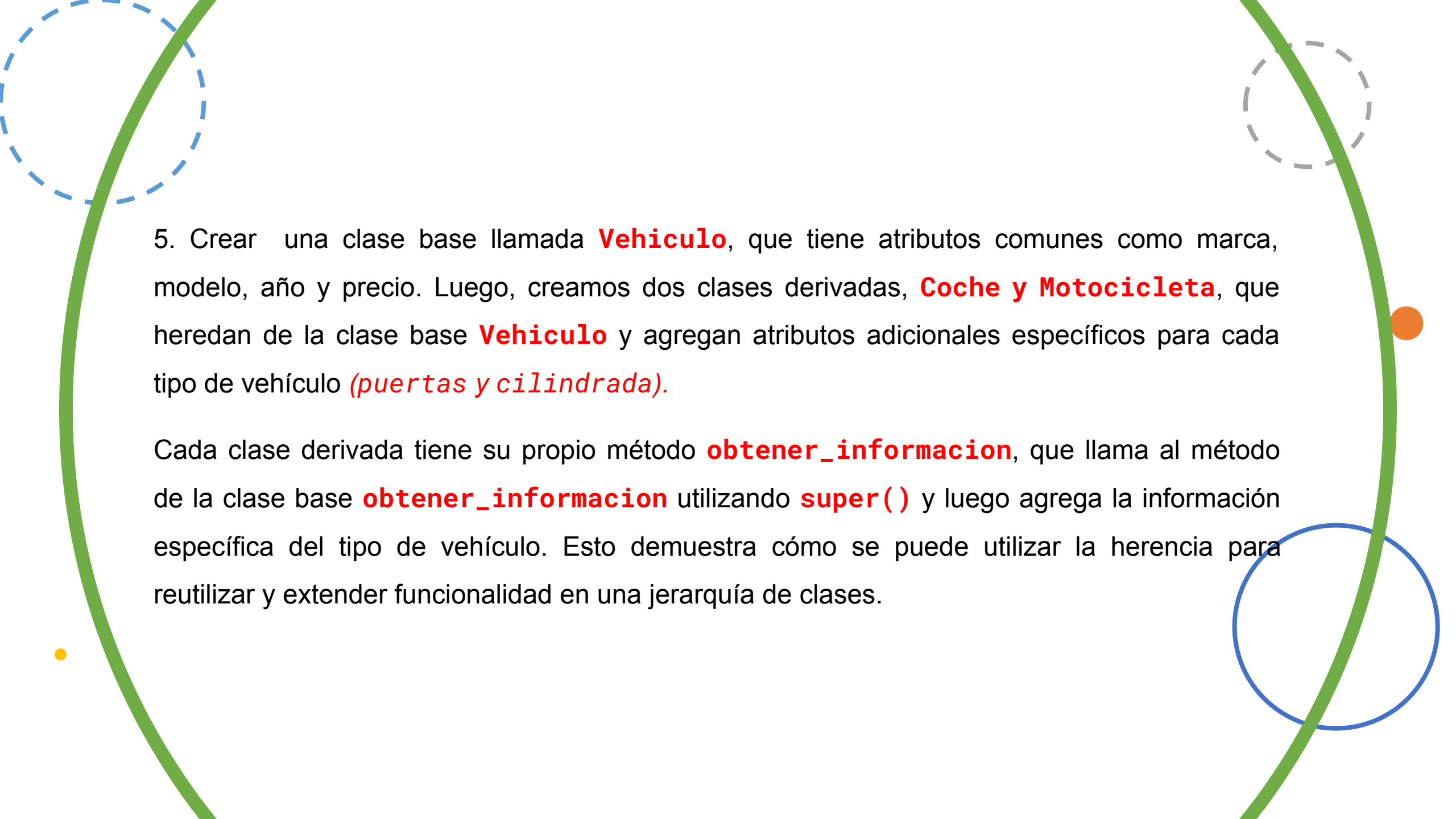
2. Crear una clase **CuentaBancaria** que tenga atributos como titular, saldo y dos métodos para **depositar** y **retirar** dinero. Si no tiene saldo debe mostrar el mensaje “saldo insuficiente”. Establece un **menú**



3. Diseña una agenda de contacto con una clase **Contacto** que represente a una persona con **atributos** como nombre, dirección de correo electrónico y número de teléfono. Luego, crea una clase **Agenda** que almacene una lista de contactos y ofrezca métodos para agregar, eliminar y buscar contactos. Utiliza encapsulamiento para proteger los datos de contacto. Establece un **menú**.

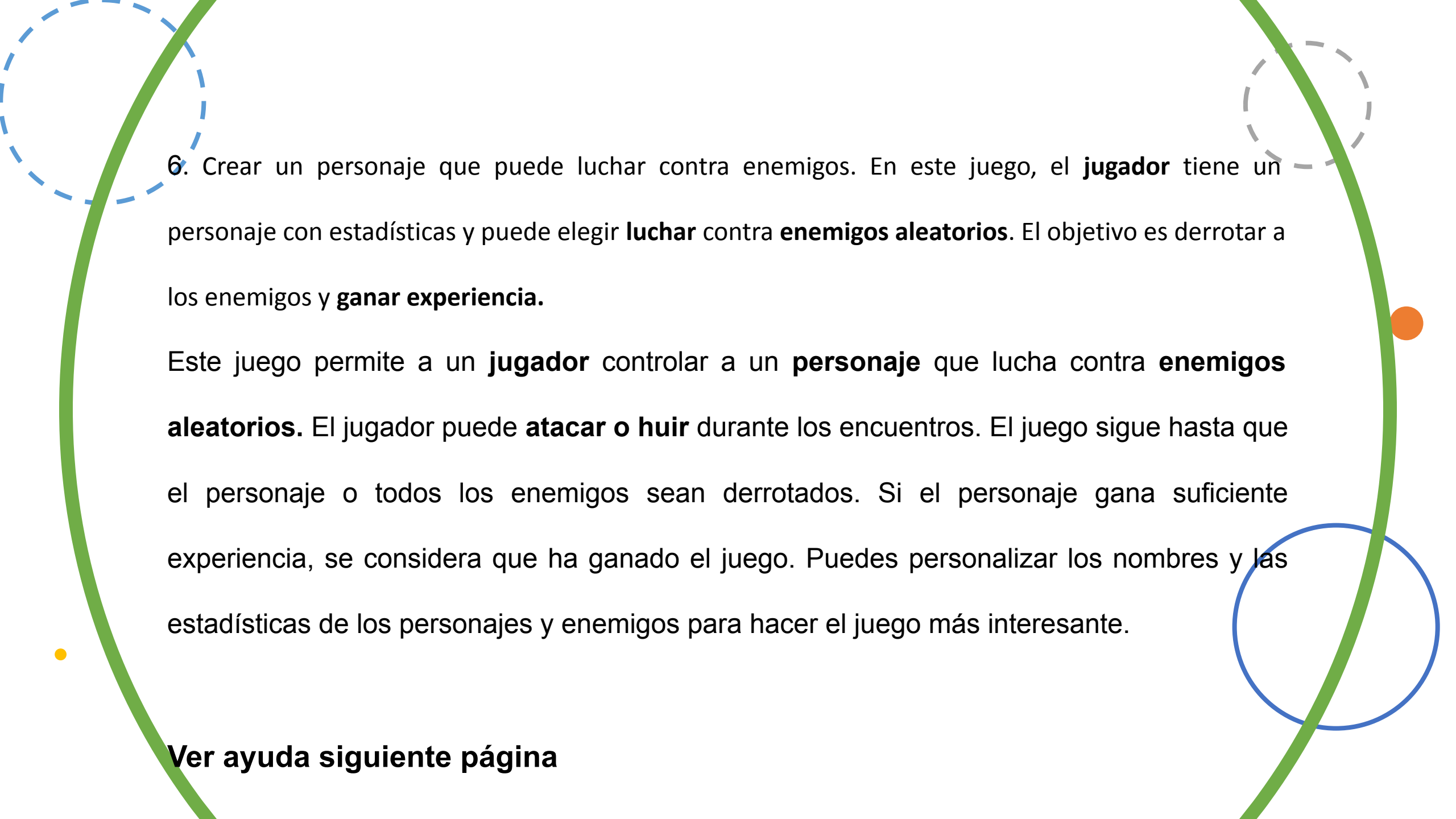


4. Diseña un sistema de clases para **simular una tienda**. Debes tener clases para representar **productos, carritos de compra y clientes**. Los productos deben tener **atributos como nombre, precio y cantidad** en stock. Los **clientes deben poder agregar productos a su carrito de compra y realizar compras**. El carrito de compra debe mantener un **registro de los productos** y calcular el total de la compra



5. Crear una clase base llamada **Vehiculo**, que tiene atributos comunes como marca, modelo, año y precio. Luego, creamos dos clases derivadas, **Coche y Motocicleta**, que heredan de la clase base **Vehiculo** y agregan atributos adicionales específicos para cada tipo de vehículo (*puertas y cilindrada*).

Cada clase derivada tiene su propio método **obtener_informacion**, que llama al método de la clase base **obtener_informacion** utilizando **super()** y luego agrega la información específica del tipo de vehículo. Esto demuestra cómo se puede utilizar la herencia para reutilizar y extender funcionalidad en una jerarquía de clases.



6. Crear un personaje que puede luchar contra enemigos. En este juego, el **jugador** tiene un personaje con estadísticas y puede elegir **luchar** contra **enemigos aleatorios**. El objetivo es derrotar a los enemigos y **ganar experiencia**.

Este juego permite a un **jugador** controlar a un **personaje** que lucha contra **enemigos aleatorios**. El jugador puede **atacar o huir** durante los encuentros. El juego sigue hasta que el personaje o todos los enemigos sean derrotados. Si el personaje gana suficiente experiencia, se considera que ha ganado el juego. Puedes personalizar los nombres y las estadísticas de los personajes y enemigos para hacer el juego más interesante.

Ver ayuda siguiente página

```
def main():
    nombre_personaje = input("Ingresa el nombre de tu personaje: ")
    personaje = Personaje(nombre_personaje)

    enemigos = [Enemigo("Orco"), Enemigo("Shawk"), Enemigo("Drago")]

    while personaje.esta_vivo():
        enemigo = random.choice(enemigos)
        print(f"\nTe encuentras con un {enemigo.nombre} enemigo.")

        while enemigo.esta_vivo() and personaje.esta_vivo():
            print("\nEstado actual:")
            personaje.mostrar_estado()
            enemigo.mostrar_estado()

            accion = input("¿Qué deseas hacer? (atacar/huir): ").lower()

            if accion == "atacar":
                personaje.atacar(enemigo)
                if enemigo.esta_vivo():
                    enemigo.atacar(personaje)
            elif accion == "huir":
                print("Escapas del combate.")
                break
            else:
                print("Acción no válida. Ingresa 'atacar' o 'huir'.")

        if enemigo.vida <= 0:
            experiencia_ganada = random.randint(10, 20)
            personaje.ganar_experiencia(experiencia_ganada)
            print(f"Has derrotado al {enemigo.nombre} enemigo y ganado {experiencia_ganada} puntos de experiencia.")
        else:
            print("Has sido derrotado por el {enemigo.nombre} enemigo.")

    print("\nEl juego ha terminado.")
    if personaje.experiencia >= 50:
        print(f"{personaje.nombre} ha ganado el juego y se ha convertido en un gran guerrero.")

if __name__ == "__main__":
    main()
```