Evaluación Final del Módulo 4

Actividad:

Sistema de Gestión de Jugadores de Fútbol

Objetivo: Evaluar la aplicación de POO, herencia, polimorfismo y archivos en Python.

Descripción: El club "Python FC" requiere un sistema para gestionar jugadores, permitiendo registrar nombre, edad, posición y goles, listar jugadores almacenar y recuperar datos desde un archivo, aplicar herencia para distinguir capitanes y jugadores regulares, e implementar excepciones para manejar errores en la entrada.

INSTRUCCIONES:

- Revisa las instrucciones detalladas que se presentan en la página 2 de este documento.
- Revisa la rúbrica de evaluación expuesta en la página 3 de este documento.

INDICACIONES

Tipo de Entrega:	.Zip, .Rar, GitHub.
Tiempo:	90 minutos.
Número de participantes.	Individual.

KIBERNUM
IT Academy

Criterios de Evaluación :

Instrucciones:

1. Definir la clase principal:
   - Crear una clase Jugador con atributos: nombre, edad, posición y goles.
   - Implementar el método constructor __init__() y un método mostrar_info() para imprimir los datos del jugador.
2. Herencia:
   - Crear una subclase Capitán que herede de Jugador y agregue el atributo liderazgo.
   - Sobrescribir mostrar_info() para incluir el nivel de liderazgo del capitán.
3. Gestión de archivos:
   - Implementar un método para guardar jugadores en un archivo (jugadores.txt).
   - Implementar un método para cargar jugadores desde el archivo y mostrarlos en pantalla.
4. Manejo de excepciones:
   - Controlar errores con try/except cuando el usuario ingrese valores incorrectos (por ejemplo, edad en texto o goles negativos).

Criterios de Evaluación :

Rubrica

Criterio	Muy bien (90-100%)	Bien (70-89%)	Suficiente (50-69%)	Regular (0-49%)
Definición de la clase principal	La clase `Jugador` está correctamente definida con atributos, constructor y método `mostrar_info()`, sin errores.	La clase `Jugador` está bien definida, pero con pequeños errores en la implementación.	La clase `Jugador` tiene errores en los atributos o métodos, pero funciona parcialmente.	No se implementa correctamente la clase `Jugador` o no funciona.
Implementación de herencia	La subclase `Capitán` hereda correctamente de `Jugador`, añade `liderazgo` y sobrescribe `mostrar_info()`.	La subclase `Capitán` está implementada, pero con pequeños errores en la herencia o sobreescritura.	La subclase `Capitán` está incompleta o tiene errores importantes en su implementación.	No se implementa la herencia correctamente o no se incluye `Capitán`.
Gestión de archivos	Se implementan correctamente los métodos para guardar y cargar jugadores desde un archivo, asegurando su correcta lectura y escritura.	Los métodos de gestión de archivos funcionan, pero presentan pequeños errores de almacenamiento o recuperación.	La gestión de archivos tiene fallos importantes o no guarda/carga correctamente los datos.	No se implementa la gestión de archivos o no funciona.
Manejo de excepciones	Se implementan `try/except` para prevenir errores en la entrada de datos, asegurando robustez.	El manejo de excepciones está presente, pero no cubre todos los posibles errores.	Se implementan `try/except`, pero con errores o cobertura insuficiente.	No se implementa manejo de excepciones o el código no maneja errores.