PROYECTO PROGRAMACIÓN ESTRUCTURADA Y MODULAR

INSTRUCCIONES. Cada pareja o grupo debe repartir las 7 funciones a realizar entre cada componente, se debe hacer seguimiento con repositorio en GitHub. En el fichero README.md del repositorio debe aparecer el reparto de tareas a cada integrante. Una vez creado el repositorio, invita como colaboradores al resto de participantes del grupo y al profesor: cfergar586, con permisos de edición.

ENUNCIADO DEL PROYECTO. Partiendo del fichero csv `liga.csv` con los resultados de las jornadas de liga 2015-2016, realizar un programa que muestre la tabla de clasificación al final de la liga, en el que debe aparecer el orden en que ha quedado cada equipo, los partidos ganados, los partidos empatados y los partidos perdidos, y por último los puntos conseguidos. Para realizar este programa vamos a construir varias funciones:

- `LeerPartidos()`: Función que lee el fichero CSV y devuelve los datos del mismo en una lista de diccionarios.
- `impClasificacion(liga)`:Recibe la lista de diccionarios generado a partir de la función anterior, genera los datos de la clasificación y los imprime por pantalla.

Esta función utiliza interna las siguientes funciones:

- `Equipos(datosliga)`: Función que recibe la lista de diccionarios con los datos de la liga y devuelve un conjunto con los equipos de la liga.
- `InfoEquipos(datosliga,equipos)`: Función que recibe la lista de diccionarios con los datos de la liga y el conjunto de equipos y devuelve una lista de tuplas, en cada tupla se guarda un equipo con los partidos ganados, empatados y perdidos y los puntos obtenidos.
  Esta función utiliza internamente:
  - `QuienGana(resultado)`: Función que recibe un resultado y devuelve un 0 si es un empate, un 1 si gana el equipo de casa y -1 si gana el equipo visitante.
  - `Puntos(info)`: Función que recibe una lista con los partidos ganados, empatados y perdidos y devuelve los puntos obtenidos.
- `Clasificacion(datos)`: Recibe la lista generada con la función anterior y la ordena según el número de puntos.

Las funciones deben incluir comentarios para que aparezcan con comando help(function).

Ejemplo de Ejecución

Nº	Equipo	PG	PP	PE	Puntos
1	Barcelona	29	5	4	91
2	Real Madrid	28	4	6	90
3	Ath Madrid	28	6	4	88
4	Villarreal	18	10	10	64
5	Ath Bilbao	18	12	8	62
6	Celta	17	12	9	60
7	Sevilla	14	14	10	52
8	Malaga	12	14	12	48
9	Sociedad	13	16	9	48
10	Betis	11	15	12	45
11	Las Palmas	12	18	8	44
12	Valencia	11	16	11	44
13	Espanol	12	19	7	43
14	Eibar	11	17	10	43
15	La Coruna	8	12	18	42
16	Granada	10	19	9	39
17	Sp Gijon	10	19	9	39
18	Vallecano	9	18	11	38
19	Getafe	9	20	9	36
20	Levante	8	22	8	32

RÚBRICA DE EVALUACIÓN. Proyecto: Sistema de clasificación de liga a partir de CSV. RAs evaluados: RA1, RA2, RA3

RESULTADO DE APRENDIZAJE	CRITERIOS DE EVALUACIÓN	EXCELENTE (4)	SATISFACTORIO (3)	EN DESARROLLO (2)	INICIAL (1)	OBSERVACIONES (basadas en el código)
RA1: Análisis del problema	CE1a: Documentación del problema	Documentación completa en README con análisis detallado	README contiene descripción básica	Documentación incompleta	No hay documentación	*Evaluar README.md del repositorio*
	CE1b: Estrategias de resolución	Estrategia clara que conecta todas las funciones	Estrategia definida pero falta conexión	Estrategia confusa	Sin planificación	*Código muestra buena estructura modular*
	CE1c: Análisis de dificultades	Identifica y documenta posibles problemas	Identifica algunas dificultades	Menciona dificultades superficiales	No analiza dificultades	*¿Documentaron problemas con CSV, tipos de datos?*
	CE2b: Uso de IDLE/entorno	Configura entorno correctamente	Usa entorno básico sin problemas	Dificultades con configuración	No configura entorno	*Shebang correcto, imports adecuados*
	CE2c: Bloques del programa	Estructura modular clara	Estructura adecuada pero falta modularidad	Estructura confusa	Sin estructura definida	EXCELENTE: Estructura perfecta con `if __name__ == '__main__':`

RA2: Sintaxis y entorno Python	CE2d: Operadores	Usa correctamente todos los operadores	Usa operadores básicos correctamente	Errores en uso de operadores	Uso incorrecto	EXCELENTE: Operadores aritméticos, comparación y lógicos usados correctamente
	CE2e: Tipos de datos	Implementa correctamente todos los tipos	Usa tipos básicos correctamente	Confusión entre tipos	Uso incorrecto	EXCELENTE: Uso correcto de `list`, `dict`, `tuple`, `set`, `int`, `str`
	CE2f: Verificación resultados	Incluye pruebas/ejemplos	Verifica el resultado final	Verificación incompleta	No verifica	*Faltan pruebas unitarias. Programa corre pero sin validación explícita*
RA3: Estructuras de control	CE3a: Identificación estructuras	Usa condicionales, bucles y funciones óptimamente	Usa estructuras adecuadamente	Uso limitado	No utiliza apropiadas	EXCELENTE: Uso completo de todas las estructuras
	CE3b: Sentencias condicionales	Usa if-elif-else correctamente	Implementa condicionales básicos	Errores en lógica	No implementa	EXCELENTE: Condicionales en `QuienGana()` y `InfoEquipos()` correctos
	CE3c: Bucles for/while	Usa bucles for eficientemente	Usa bucles pero podría optimizarse	Bucles con errores	No usa bucles	EXCELENTE: Bucles `for` en `LeerPartidos()`,

						`InfoEquipos()`, etc.
	CE3d: Funciones Python	Crea y usa funciones eficientemente	Usa funciones correctamente	Funciones con problemas	No usa funciones	EXCELENTE: Implementa las 7 funciones requeridas con parámetros y retornos
	CE3e: Código robusto y eficiente	Código robusto, manejo de errores	Código funcional pero frágil	Código con problemas	Código no funcional	SATISFACTORIO: Código funcional pero sin manejo de errores (ej: archivo no existe)
IMPLEMENTACIÓN ESPECÍFICA	Funciones requeridas	Las 7 funciones implementadas perfectamente	Todas implementadas con pequeños errores	Faltan 1-2 funciones	Faltan más de 2	EXCELENTE: Implementa las 7 funciones exactas
	Lógica de cálculo	Cálculo perfecto, ordenación correcta	Cálculo correcto con pequeños errores	Errores en fórmula u ordenación	No calcula bien	EXCELENTE: Cálculo de puntos correcto (3-1-0)
	Manejo CSV	Lee CSV correctamente, maneja encabezados	Lee CSV pero con pequeños problemas	Problemas con formato CSV	No lee CSV	EXCELENTE: Uso correcto de módulo `csv`, salta encabezado
	Repositorio GitHub	Repositorio organizado,	Repositorio básico con	Pocos commits,	Sin repositorio	*Evaluar repositorio GitHub*

GESTIÓN DEL PROYECTO		commits frecuentes	commits	desorganizado		
	Distribución de tareas	Distribución clara en README, equitativa	Distribución documentada	Distribución poco clara	No documentan distribución	*Revisar README.md*
	Trabajo colaborativo	Commits distribuidos, resuelven conflictos	Trabajan juntos pero commits desbalanceados	Poca colaboración evidente	No trabajan colaborativamente	*Revisar historial de commits*