UNIDAD 6
ACTIVIDAD PRÁCTICA
CADENAS EN PROFUNDIDAD Y ARCHIVOS

U-tad
CENTRO UNIVERSITARIO DE TECNOLOGÍA Y ARTE DIGITAL

Centro adscrito a
Universidad Camilo José Cela

ÍNDICE

ORGANIZACIÓN DE LA ACTIVIDAD PRÁCTICA

Nombre de la práctica	
Tipo de tarea	Individual
Entregables	Archivo zip con los diferentes ejercicios del alumno (Revisar apartado: "Como realizar mi entrega")

DESCRIPCIÓN DE LA ACTIVIDAD PRÁCTICA

- Ejercicio 1 (10 puntos)

  Se pide crear un programa que pueda administrar los datos de los personajes de un videojuego. Para ello contamos con un fichero que contiene la información de estos personajes, en el caso de que no exista se debe pedir al usuario la información de los personajes y guardarla en dicho fichero. Los datos que incluye son:

  - Nombre
  - Clase del personaje (Únicamente pueden ser de los tipos: guerrero, mago, arquero, paladín y rogue)
  - Nivel
  - Vida
  - Poder de ataque
  - Capacidad de defensa
  - Habilidad de magia

  El formato del fichero:
  Por cada línea, se tienen los datos de un personaje:

```
Aragorn,Guerrero,10,100,50,30,70
Gandalf,Paladín,8,120,40,20,90
Legolas,Arquero,9,90,70,20,50
Frodo,Rogue,6,80,30,10,40
```

  Se debe mostrar por pantalla todos los personajes leídos del fichero, la media de los niveles y se deben guardar en un fichero aquellos personajes cuyo nivel sea mayor que 7.

  Ejemplo de ejecución:

```
Lista de personajes:
Nombre: Aragorn, Clase: Mago, Nivel: 10, HP: 100, ATK: 50, DEF: 30, MAG: 70
Nombre: Gandalf, Clase: Paladin, Nivel: 8, HP: 120, ATK: 40, DEF: 20, MAG: 90
Nombre: Legolas, Clase: Arquero, Nivel: 9, HP: 90, ATK: 70, DEF: 20, MAG: 50
Nombre: Frodo, Clase: Rogue, Nivel: 6, HP: 80, ATK: 30, DEF: 10, MAG: 40

Estadisticas de los personajes:
Media de nivel de todos los personajes: 8.25
```

  Ejemplo de fichero de salida:

```
Aragorn,Guerrero,10,100,50,30,70
Gandalf,Paladín,8,120,40,20,90
Legolas,Arquero,9,90,70,20,50
```

RECOMENDACIONES E INDICACIONES PARA LA ENTREGA

En este ejercicio está permitido usar funciones derivadas de la librería <string.h>

- En concreto funciones:
  - int strcmp(const char *s1, const char *s2);
  - size_t strlen(const char *s);
  - char *strcpy(char *dest, const char *orig);
  - size_t strcspn(const char *s, const char *reject);
  - char *strcat(char *dest, const char *src);
  - char *strstr(const char *haystack, const char *needle);
  - char *strtok(char *string1, const char *string2);

Se pueden usar el resto de las funciones vistas en los temas 1 , 2 , 3 y 4, incluidas en <stdlib.h> y <stdio.h>

- atoi, atof,strtol,strtof,malloc, realloc,free, etc...

- Claridad en el código:

Asegúrate de que tu código sea claro y esté bien estructurado. Sigue las buenas prácticas de programación, como el uso de indentaciones y comentarios, para facilitar su lectura y comprensión.

- Uso de identificadores descriptivos:

En la implementación en C, utiliza nombres de variables que describan claramente su propósito (por ejemplo, exponente en lugar de solo exp).

- Revisa los datos de entrada:

Antes de implementar los algoritmos en C, verifica que tienes claro cuáles son los posibles datos que un usuario puede introducir en tu programa y asegúrate de que no da errores al introducir datos no admitidas como letras cuando se soliciten números.

RÚBRICA DE CORRECCIÓN

La rúbrica para corregir el ejercicio seguirá los criterios listados a continuación, que tienen distintos pesos respecto al total de la nota.

Criterios	Excelente	Satisfactorio	No satisfactorio	Insuficiente

Ejercicio 1: Conversor	De 3 a 2.25 puntos: Programa funcional y correcto, con buen uso de condicionales y bucles y funciones, estructuras, argc/argv, lógica clara y sin errores importantes. Y revisión de parámetros de entrada.	De 2.24 a 1.5 puntos: Programa funcional con algunos errores menores en la lógica o en la implementación, pero produce resultados correctos.	De 1.49 a 0.75 punto: Programa con errores significativos que no produce el resultado correcto, aunque hay un intento de implementación.	De 0.74 a 0 puntos: Programa incorrecto o muy incompleto, sin funcionalidad significativa.
Ejercicio 2: Gladiadores	De 7 a 5.25 puntos: Programa funcional y correcto, con buen uso de condicionales y bucles y funciones, estructuras, argc/argv, lógica clara y sin errores importantes. Y revisión de parámetros de entrada.	De 5.24 a 3.5 puntos: Programa funcional con algunos errores menores en la lógica o en la implementación, pero produce resultados correctos.	De 3.49 a 1.75 punto: Programa con errores significativos que no produce el resultado correcto, aunque hay un intento de implementación.	De 1.74 a 0 puntos: Programa incorrecto o muy incompleto, sin funcionalidad significativa.

COMO REALIZAR LA ENTREGA

Para realizar le entrega de esta práctica se pide al alumno entregas un archivo zip con el nombre de NombreAlumno_PrimerApellido_PracticaTema6.zip.

El zip debe contener los siguientes archivos:

- Un archivo llamado ejercicio1.c. Este archivo debe contener el código compilable del ejercicio 1.

Además, se solicita al alumno que todos los archivos vayan iniciados con un comentario multilínea con indicando su nombre y el número del ejercicio.