Algoritmos y Estructuras de Datos II

Parcial : Tema B - Cálculo de Mejor rankeados UCI

La unión de ciclistas internacionales publica diariamente el ranking de sus 100 mejores ciclistas. El ranking se basa en una sumatoria de puntos que cada corredor obtiene por su posición en carreras denominadas UCI.
Tenemos un sistema que muestra la tabla con los 100 mejores ciclistas, sin embargo se necesita manipular los datos para personalizarlos y mostrar mejores resultados que servirán para el posterior análisis.
En el directorio del ejercicio se encuentran los siguientes archivos:

Archivo	***Descripción***
`main.c`	Contiene la función principal del programa
`ranking.h`	Declaraciones relativas a la estructura del ranking y de funciones de carga y escritura de datos.
`ranking.c`	Implementaciones **incompletas** de las funciones
`array_helpers.h`	Declaraciones / prototipos de las funciones que manejan tablas
`array_helpers.c`	Implementaciones **incompletas** de las funciones que manejan el arreglo

Abrir el archivo **`input/uci_rank.in`** para ver cómo se estructuran los datos.

Cada línea contiene los datos de los corredores UCI.

El archivo se presenta sin una cabecera es por ello que se especifica cada posición y su significado:

@T PA PP DP NC NE PT

- **T:** tipo de carrera
- **PA:** posición actual en el ranking del corredor
- **PP:** posición previa en el ranking del corredor
- **DP:** diferencia entre la posición actual y previa
- **NC:** nombre del ciclista
- **NE:** nombre del equipo del ciclista
- **PT:** puntos totales obtenidos por el ciclista hasta el momento

Consideraciones:

- El tipo de la carrera puede ser 0 (road) o 1 (track). Este tipo de dato está definido en

ranking.h
- La diferencia es negativa cuando el ciclista bajó puestos en el ranking.
- Los demás valores serán siempre positivos.
- Los string en C se manejan de una manera particular, es por ello que el código entregado ya contiene el manejo de los mismos.
- Para la lectura de strings usando **`fscanf()`** existe el comodín **`%s`** que puede utilizar. Tener en cuenta que el parámetro asociado si es un arreglo va sin el `'&'` (pueden consultar al docente para dudas sobre esto).
- Cada corredor participa en ambos tipos de carreras, esto significa que:
    - Hay dos y solo dos entradas por cada corredor, una para cada tipo de carrera, por lo cual el archivo contiene el doble de entradas que numero de corredores
    - La lista **no** está ordenada
    - Corredores pueden pertenecer a distintos equipos para distintos tipos de carrera

Ejercicio 1: Carga y Lectura de datos

El ejercicio consiste en completar el procedimiento de carga de datos en los archivos **`array_helpers.c` (array_from_file)** y **`ranking.c (ranking_from_file)`**. Los datos deben cargarse de manera correcta en el arreglo usando los índices adecuados.

Recordar que el programa tiene que ser *robusto*, es decir, debe tener un comportamiento bien definido para los casos en que la entrada no tenga el formato esperado.

Una vez completada la lectura de datos se puede verificar si la carga funciona compilando,

```
$ gcc -Wall -Werror -Wextra -pedantic -std=c99 -c array_helpers.c ranking.c
main.c
$ gcc -Wall -Werror -Wextra -pedantic -std=c99 array_helpers.o ranking.o main.o
-o ranking
```

y luego ejecutar por ejemplo

```
$ ./ranking input/uci_rank.in
```

IMPORTANTE!!!!!

El sistema imprime la lista luego de leerla. **Se descontarán puntos** si la lista no se imprime de manera ordenada (por posición en el ranking!). No es necesario implementar un algoritmo de ordenación para lograr esto.

Ejemplo de cómo se debería imprimir:

```
1 road 1 0 POGACHAR UAE 6158.000000
1 track 1 0 KUSS UAE 6158.000000
2 road 2 0 VINGEGAARD Visma 5970.000000
2 track 2 0 DELIE Visma 5970.000000
3 road 3 0 EVENEPOEL Quick-Step 5697.000000
3 track 3 0 PHILIPSEN Quick-Step 5697.000000
4 road 9 5 PHILIPSEN Alpecin 4812.000000
4 track 9 5 EVENEPOEL Alpecin 4812.000000
…
```

Ejercicio 2: Análisis de los datos

Se pide calcular el puntaje total por equipo. Para ello se debe completar en la función main, de tal forma que el programa luego de imprimir el array en stdio, pregunte el nombre del equipo del cual se tiene que calcular la suma total.

Una vez ingresado el nombre del equipo, se deberá ejecutar la función `total_track_points_per_team`. Esta función se encuentra definida en ranking.h y deberá ser implementada en ranking.c. La misma tiene que calcular la suma del total de los puntos de todos los ciclistas pertenecientes a ese equipo, **solo para carreras de tipo `track`**

Finalmente, se deberá imprimir por pantalla dicho total, de la siguiente forma:

“Total Maximo de puntos del equipo <nombre> es: <puntos>”

IMPORTANTE!!!!!

Para lograr esto, usar strcmp(string1, string2) == 0

Donde el resultado de strcmp(string1, string2) será igual a 0 si ambos strings son iguales.

Hints!

1. Debug.
2. Lean todo el código provisto.
3. Visma es 19796.798828
4. Usen las constantes declaradas en los .h y completen donde corresponda.
5. Enserio usen debugging (gdb).
6. Relájense es solo un examen, la vida continúa mañana.