Fundamentos de la programación – Grupos B, C y D

Curso 2016–2017

Examen de febrero de 2017

Tiempo disponible: 3 horas

Se pide desarrollar un programa que permita actualizar la clasificación de una liga de futbol con `N` participantes, siendo `N` un entero par estrictamente positivo. Para cada participante en la liga dispondremos de un acrónimo único (una cadena de caracteres sin espacios en blanco) y del número de puntos de los que dispone. Por ejemplo, para `N=10`, partiremos de una información similar a:

Celta	GET1	ATM	DEP	BCN	GET2	ZARA	Sevilla	VAL	RM
42	12	50	31	53	14	16	12	14	49

La clasificación se encuentra almacenada en el fichero `liga.txt`. Dicho fichero contiene exactamente `N` líneas, donde cada una de ellas tiene un nombre y un entero, separados por un espacio en blanco. A la derecha se puede ver el contenido de este fichero para los datos anteriores.

```
Celta 42
GET1 12
ATM 50
DEP 31
BCN 53
GET2 14
ZARA 16
Sevilla 12
VAL 14
RM 49
```

La información de la clasificación se actualiza mediante un fichero `jornada.txt`. Este fichero contiene `N/2` líneas. Cada línea contiene 4 datos: equipo1 entero1 equipo2 entero2, denotando que los equipos han disputado un partido y cada equipo ha marcado entero1/entero2 goles, respectivamente. A la izquierda se puede ver un ejemplo de fichero `jornada.txt`.

```
ATM 5 Celta 0
DEP 3 BCN 1
GET2 4 GET1 4
RM 0 ZARA 9
VAL 1 Sevilla 1
```

Al procesar una jornada, si un equipo ha marcado más goles que el contrario recibirá 3 puntos, recibiendo el contrario 0 puntos. Si los dos equipos han marcado el mismo número de goles entonces cada equipo recibe un punto. Por ejemplo, si procesamos la jornada anterior para la clasificación indicada, deberíamos obtener una nueva clasificación:

Celta	GET1	ATM	DEP	BCN	GET2	ZARA	Sevilla	VAL	RM
42	13	53	34	53	15	19	13	15	49

En primer lugar, empieza declarando los **tipos y constantes** adecuados para almacenar y manipular la `clasificacion` y la `jornada` tras ser leídas del fichero **(1 punto)**. Debes

incluir, en particular, tipos que permitan almacenar los puntos de cada equipo y los resultados de una jornada de liga.

El **programa principal** incluye un menú que proporciona las distintas opciones y características que se presentan a continuación. El profesor te proporcionará un esquema de la solución `PlantillaFebrero2017.cpp` en el que ya está programado el menú y en el que tú deberás rellenar el resto de la funcionalidad que se pide.

Carga de datos: liga (1 punto)

El programa cargará la clasificación actual a partir del fichero `liga.txt`. Se puede asumir que cada línea del fichero contiene exactamente un *string* y un entero, que no se repiten los acrónimos, y que el fichero tiene exactamente `N` líneas pero será necesario controlar que el fichero existe.

Deberás implementar un subprograma `cargarLiga` que lleve a cabo dicha tarea y que además devuelva un valor que indique si la tarea se pudo llevar a cabo o no.

Carga de datos: jornada (1 punto)

Similar a la opción anterior pero tomando los datos del fichero `jornada.txt`. Se puede asumir que el fichero tiene exactamente `N/2` líneas, que los goles son enteros mayores o iguales a cero y que cada línea del fichero sigue el patrón *string* entero *string* entero. Además, se deberá comprobar **(1 punto)** que en la jornada no aparece ningún equipo dos o más veces**.**

Deberás implementar un subprograma `cargarJornada` que lleve a cabo dicha tarea y que además devuelva un valor que indique si la tarea se pudo llevar a cabo o no.

Muestra por pantalla la clasificación (1 punto)

Mostrará por pantalla la clasificación cargada en memoria. Si no existe ninguna clasificación (porque no se han cargado los datos con anterioridad) se mostrará un mensaje de error. Se valorará la presentación de los datos, recomendándose utilizar `setw(), left` y `right.` Se puede asumir que los acrónimos tienen a lo sumo `max_longitud` caracteres (por ejemplo, `max_longitud=8`) y que un equipo no consigue más de `10^max_puntos` - 1 puntos al año (por ejemplo, `max_puntos=3`).

Deberás implementar un subprograma `mostrarLiga` que, dada una clasificación, la muestre por pantalla según lo indicado.

Muestra por pantalla la jornada (1 punto)

Similar a la opción anterior pero para los datos correspondientes a una jornada. Se puede asumir, adicionalmente, que un equipo no marca más de `10^max_goles` – 1 goles en un partido (por ejemplo, `max_goles=2`).

Deberás implementar un subprograma `mostrarJornada` que, dada una jornada, la muestre por pantalla según lo indicado.

Actualización de una clasificación a partir de una jornada (2 puntos)

Esta opción actualiza la clasificación cargada en memoria con la jornada cargada en memoria. Si no se han cargado previamente la clasificación o la jornada, mostrará un mensaje de error. Se puede asumir que los equipos que conforman una jornada coindicen con los que conforman la liga.

Deberás implementar un subprograma `actualizarLiga` que, dada una clasificación y una jornada, modifica dicha clasificación de acuerdo con los resultados obtenidos en la jornada, según se ha explicado previamente.

Mostrar el primer clasificado (1,5 puntos)

Mostrará por pantalla el equipo que lleva más puntos. Si hay varios equipos empatados entonces muestra el primero que aparezca. Por ejemplo, para la última clasificación presentada debería mostrar el acrónimo `ATM`. Si no existe ninguna clasificación ya cargada se mostrará un mensaje de error.

Deberás implementar un subprograma `mostrarPrimero` que, dada una clasificación, muestre por pantalla el equipo que lleva más puntos.

Grabación de datos: liga (0,5 puntos)

Dada una clasificación cargada en memoria, graba los datos en el fichero `liga.txt`. Si el fichero existe, los datos anteriores se sobrescribirán. Será necesario controlar que hay una clasificación cargada previamente.

Deberás implementar un subprograma `guardarLiga` que, dada una clasificación, la guarde en el fichero indicado.

Instrucciones de entrega

1. Añade al inicio de tu archivo .cpp un comentario con tus datos:

```
/*
Apellidos:
Nombre:
DNI:
Puesto:
*/
```

2. Usa la herramienta de ftp para subir tu archivo .cpp (arrastra tu fichero cpp hacia la ventana derecha). Es tu responsabilidad asegurarte de que subes la última versión.

3. Pasa por el ordenador del profesor, pregúntale si tu archivo se ha recibido correctamente, y firma.