UNIVERSIDADE DA CORUÑA

FACULTADE DE INFORMÁTICA

Programación II – Curso 2020/21

Práctica 2: Enunciado

1. El problema

La segunda práctica consistirá en añadir un conjunto de nuevas funcionalidades al programa VIMFIC desarrollado en la primera práctica. En concreto:

- Incluir una lista de reproducción de vídeos a cada usuario

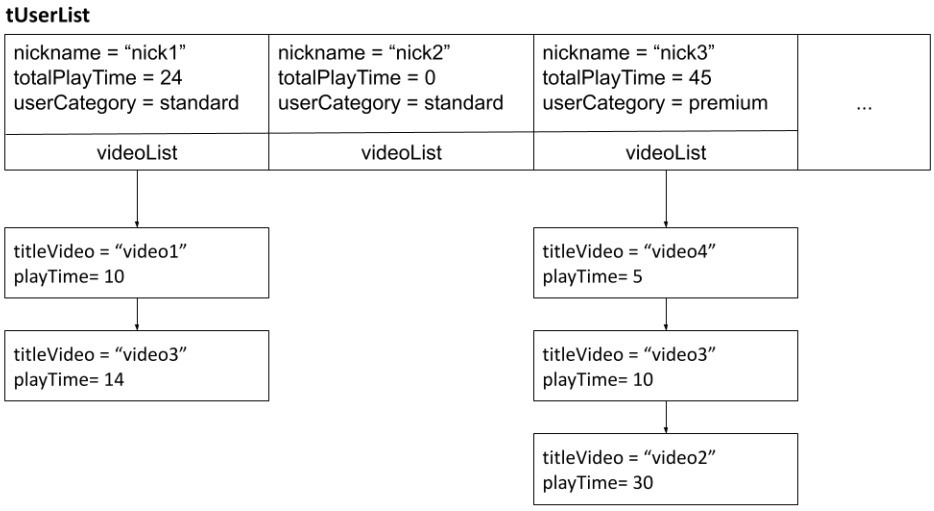
El objetivo de esta práctica es comprender el funcionamiento e implementar varios tipos abstractos de datos (TADs) así como manejar interdependencias entre ellos.

2. Estructuras de datos

Para resolver este problema se utilizarán 2 tipos abstractos de datos (TADs):

- Una Lista Ordenada (TAD UserList) para almacenar la lista de usuarios.
- Una Lista (TAD VideoList) para almacenar los vídeos de cada usuario.

Cada usuario de VIMFIC, esto es, cada nodo de la lista ordenada tendrá su propia lista de reproducción, tal y como representa la siguiente figura:



2.1 types.h

Algunos tipos de datos comunes se definirán en el fichero `types.h`, ya que se utilizarán en varios TADs.

NAME_LENGTH_LIMIT	Longitud máxima de un nick y de un título de vídeo (constante)
tNickname	Nick de un usuario (string)
tUserCategory	Categoría de usuario (tipo enumerado: {standard, premium})
tPlayTime	Tiempo de reproducción, en minutos (integer)
tTitleVideo	Título de un vídeo (string)
tVideo	Datos de un vídeo. Contendrá el campo: • titleVideo de tipo tTitleVideo • playTime de tipo tPlayTime

2.2 TAD User List

Para mantener la lista de usuarios y su información asociada el sistema utilizará un TAD Lista ordenada con implementación **dinámica, simplemente enlazada** (user_list.c y user_list.h).

2.2.1. Tipos de datos incluidos en el TAD User List

tUserList	Representa una lista de usuarios **ordenada por sus nicks**
tUserItem	Datos de un elemento de la lista (un usuario). Compuesto por: • nickname: de tipo tNickname • totalPlayTime: de tipo tPlayTime • userCategory: de tipo tUserCategory • videoList: de tipo tVideoList
tUserPos	Posición de un elemento de la lista de usuarios
NULL_USER	Constante usada para indicar posiciones nulas de la lista de usuarios

2.2.2. Operaciones incluidas en el TAD Lista Ordenada

Las operaciones del TAD Lista ordenada son idénticas a las indicadas en la práctica 1 (consulte el enunciado de dicha práctica) **cambiando los nombres de los tipos adecuadamente**, únicamente cambian de especificación las siguientes operaciones:

- `insertItem(tUserItem, tUserList) → tUserList, bool`

  Inserta un elemento en la Lista de forma **ordenada** en función del campo `nickname`. Devuelve un valor `true` si el elemento fue insertado; `false` en caso contrario.
  PostCD: Las posiciones de los elementos de la lista posteriores a la del elemento insertado pueden haber variado.

- `deleteAtPosition (tUserPos, tUserList) → tUserList`

  Elimina de la lista el elemento que ocupa la posición indicada.

  PreCD: La posición indicada es una posición válida en la lista y el usuario en dicha posición tiene una lista de vídeos vacía.

  PostCD: Las posiciones de los elementos de la lista posteriores a la de la posición eliminada pueden haber variado.

Asimismo, la operación `findItem` no cambia su especificación (salvo los nombres de los tipos de datos), pero en su implementación debería tener en cuenta que la lista está ordenada.

2.3 TAD Video List

Este TAD implementará una lista **estática** basándose en el TAD Lista (`video_list.c` y `video_list.h`) con un tamaño máximo de 25 vídeos.

2.3.1. Tipos de datos incluidos en el TAD VideoList

`tVideoList`	Representa una lista de vídeos (**no ordenada**)
`tVideoItem`	Datos de un elemento de la VideoList, es decir, un vídeo (`tVideo`)
`tVideoPos`	Posición de un elemento de la lista de vídeos
`NULL_VIDEO`	Constante usada para indicar posiciones nulas

2.3.2. Operaciones incluidas en el TAD VideoList

Las operaciones del TAD Lista son idénticas a las indicadas en la práctica 1 (consulte el enunciado de dicha práctica) **cambiando los nombres de los tipos adecuadamente y añadiendo el sufijo `V` al nombre de las operaciones**

- `createEmptyListV (tVideoList)→ tVideoList`
- `isEmptyListV (tVideoList) → bool`
- `firstV (tVideoList) → tVideoPos`
- `lastV (tVideoList) → tVideoPos`
- `nextV (tVideoPos, tVideoLiist)→ tVideoPos`

- `previousV (tVideoPos, tVideoList) → tVideoPos`
- `findItemV (tTitleVideo, tVideoList) → tVideoPos`
- `insertItemV (tVideoItem, tVideoPos, tVideoList) → tVideoList, bool`
- `deleteAtPositionV (tVideoPos, tVideoList) → tVideoList`
- `getItemV (tVideoPos, tVideoList) → tVideoItem`
- `updateItemV (tVideoItem, tVideoPos, tVideoList) → tVideoList`

3. Descripción de la tarea

La tarea consiste en implementar un único programa principal (`main.c`) que **haga uso de la UserList y de la VideoList** y que procese las peticiones de los usuarios de VIMFIC con el siguiente formato:

`N nickname userCategory`	***[N]ew***: Alta de un usuario de categoría standard o premium
`D nickname`	***[D]elete***: Baja de un usuario
`A nickname titleVideo`	***[A]dd***: Insertamos un vídeo en la VideoList de un usuario
`P nickname titleVideo playTime`	***[P]lay:*** Reproducción de playTime (minutos) de un vídeo por parte de un usuario
`S`	***[S]tats:*** Listado de los usuarios actuales de VIMFIC y sus datos
`R maxTime`	***[R]emove:*** Elimina todos los usuarios standard que hayan excedido el maxTime (minutos reproducidos)

En el programa principal se implementará un bucle que procese una a una las peticiones de los usuarios. Para simplificar tanto el desarrollo como las pruebas, el programa no necesitará introducir ningún dato por teclado, sino que leerá y procesará las peticiones de usuarios contenidas en un fichero (ver documento `EjecucionScript.pdf`) En cada iteración del bucle, el programa leerá del fichero una nueva petición y la procesará. Para facilitar la corrección de la práctica todas las peticiones del fichero van numeradas correlativamente.

Para cada línea del fichero de entrada, el programa:

1. **Muestra una cabecera con la operación a realizar.** Esta cabecera está formada por una primera línea con 20 asteriscos y una segunda línea que indica la operación tal y como se muestra a continuación:

```
********************
CC_T:_nick/maxtime_XX_category/video_YY_minutes_ZZ
```

donde `CC` es el número de petición, `T` es el tipo de operación (`N`, `D`, `A`, `P`, `S`, o `R`), `XX` es el valor del *nick* del usuario o el tiempo máximo de reproducción (cuando y según

corresponda), YY es la categoría del usuario o el título del vídeo (cuando y según corresponda), ZZ es el tiempo de reproducción para la operación Play y ␣ indica un espacio en blanco. Sólo se imprimirán los parámetros necesarios; es decir, para una petición `[S]tats` se mostrará únicamente "`01 S`", mientras que para una petición `[P]lay` se mostrará "`02 P: nick Nickname1 video Video1 minutes 35`".

2. **Procesa la petición** correspondiente:

- Si la operación es **`[N]ew`**, se incorporará el usuario a la lista de usuarios con su `totalPlayTime` inicializado a 0. Además, se imprimirá el mensaje:

  ```
  *␣New:␣nick␣XX␣category␣YY
  ```

  donde, de nuevo, ␣ representa un espacio en blanco. El resto de los mensajes siguen el mismo formato.

  Si ya existiese un usuario con ese *nick*, se imprimirá el siguiente mensaje:

  ```
  + Error: New not possible
  ```

Hay que tener en cuenta que cada usuario de la lista tiene una lista de reproducción de vídeos que debe gestionarse adecuadamente a la hora de insertarlo.

- Si la operación es **`[D]elete`**, se buscará al usuario en la lista, se borrará y se imprimirá el siguiente mensaje:

  ```
  * Delete: nick XX category YY totalplaytime ZZ
  ```

  Si no existiese ningún usuario con ese *nick*, se imprimirá el siguiente mensaje:

  ```
  + Error: Delete not possible
  ```

Al igual que la operación anterior, recordar que cada usuario tiene una lista de reproducción de vídeos que debe gestionarse adecuadamente al eliminarlo.

- Si la operación es **`[A]dd`**, se buscará el usuario y, si el vídeo no existe, se insertará al **final** de su lista de vídeos. El valor inicial del campo **`playTime`** será 0. A continuación, se mostrará el siguiente mensaje:

  ```
  * Add: nick XX adds video YY
  ```

  Si el vídeo *titleVideo* ya estuviera en la lista de vídeos o no existiese ningún usuario con ese *nick* se imprimirá el mensaje:

  ```
  + Error: Add not possible
  ```

- Si la operación es **`[P]lay`**, se buscará el usuario, dentro de su videoList se verá si dispone del vídeo indicado y, si es así, se incrementarán los minutos indicados (**`minutes`**) tanto en el propio video (**`playTime`**) como en contador total de minutos del usuario (**`totalPlayTime`**). Tras realizar este incremento, se mostrará el siguiente mensaje:

```
* Play: nick XX plays video YY playtime ZZ totalplaytime TT
```

Si el vídeo no estuviera en lista de vídeos del usuario o no existiese ningún usuario con ese *nick* se imprimirá el mensaje:

```
+ Error: Play not possible
```

- Si la operación es **[S]tats** se mostrará la lista completa de usuarios actuales incluyendo todos los videos de su lista. Si la lista de videos está vacía se indicará "No videos". A modo de ejemplo:

```
*Nick XX1 category standard totalplaytime ZZ1
Video VV11 playtime TT11
Video VV12 playtime TT12
Video VV1m playtime TT1m

*Nick XX2 category premium totalplaytime ZZ2
No videos

*Nick XXn category standard totalplaytime ZZn
Video VVn1 playtime TTn1
Video VVn2 playtime TTn2
Video VVnm playtime TTnm
```

A continuación, se mostrarán, para cada categoría de usuario, el número de usuarios, el número total de minutos reproducidos, y la media de los minutos reproducidos por tipo de usuario, con el formato siguiente:

```
Category␣Users␣TimePlay␣Average
Standard␣%6d␣%9d␣%8.2f
Premium␣␣%6d␣%9d␣%8.2f
```

Donde %6d indica que el entero correspondiente está justificado a la derecha (con 6 dígitos) y %8.2f indica un tamaño total de 8 dígitos, con dos decimales. Si la lista de usuarios estuviese vacía se imprimirá el mensaje:

```
+ Error: Stats not possible
```

- Si la operación es **[R]emove**, se eliminarán de la lista aquellos usuarios *standard* que hayan excedido el tiempo máximo de reproducción (**maxTime**). Se mostrará el siguiente mensaje:

```
Removing nick XX1 totalplaytime ZZ1
Removing nick XX3 totalplaytime ZZ3
...
Removing nick XXn totalplaytime ZZm
```

Si no existe ningún usuario de tipo *standard* que haya excedido el tiempo máximo se mostrará el mensaje:

```
+ Error: Remove not possible
```

4. Lectura de ficheros

Para facilitar el desarrollo de la práctica se proporciona el siguiente material de especial interés: (1) Un directorio `CLion` que incluye un proyecto plantilla (`P2.zip`) y, (2) Un directorio `script` donde se proporciona un fichero (`script.sh`) que permite probar de manera conjunta los distintos archivos proporcionados. Además, se facilita un documento de ayuda para su ejecución (`Ejecucion_Script.pdf`). Nótese que, para que el `script` no dé problemas se recomienda **NO copiar directamente el código de este documento**, ya que el formato PDF puede incluir caracteres invisibles que darían por incorrectas salidas válidas.

5. Información importante

El documento `NormasEntrega.pdf`, disponible en la página web de la asignatura detalla claramente las normas de entrega.

Para un mejor seguimiento de la práctica se realizarán entregas obligatorias parciales antes de las fechas y con los contenidos que se indican a continuación:

- **Viernes 23 de Abril a las 22:00 horas:** Implementación y prueba del TAD UserList y TAD VideoList (entrega de los ficheros `user_list.h`, `user_list.c`, `video_list.h` y `video_list.c`). Para comprobar el correcto funcionamiento de los TAD se facilitarán los ficheros de prueba `test_user_list.c` y `test_video_list.c`.

- **Viernes 30 de Abril a las 22:00 horas:** Implementación y prueba de las siguientes funcionalidades del programa principal: `new`, `add` y `stats` (entrega de los ficheros `user_list.h`, `user_list.c`, `video_list.h`, `video_list.c`, `types.h` y `main.c`). Para comprobar el correcto funcionamiento de las operaciones se facilitarán los ficheros de prueba `new.txt` y `add.txt`.

Se realizará una corrección automática usando el script proporcionado para ver si se supera o no el seguimiento (véase el documento `CriteriosEvauacion.pdf`).

Fecha límite de entrega: **Viernes 7 de Mayo de 2021 a las 22:00 horas**