Facultad de Informática – Universidad Complutense de Madrid

Fundamentos de la programación – Grupos F, G y H

Curso 2014–2015

Examen final – 10 de junio de 2015

Tiempo disponible: 3 horas

Se pide construir un programa modular para gestionar la lista de programas y grabaciones programadas en un Smart TV. El programa constará de cuatro módulos: *Programa, ListaProgramas*, *Grabaciones*, y módulo principal.

Módulo *Programa* (1,5 puntos)

Declara un tipo de estructura `tPrograma` con los siguientes campos: nombre (`string`), hora de inicio (`tFecha`) y duración en minutos (`int`). El campo nombre define unívocamente a un programa. El tipo `tFecha` será el mismo que el usado en la práctica 4, es decir, un alias de `time_t` (incluye la librería `ctime`), y por tanto incluye tanto el día como la hora.

Implementa, al menos, los siguientes subprogramas:

- ✓ `cargar`: Carga desde fichero (usando un flujo de entrada ya abierto recibido como parámetro) un `tPrograma`. Primero se leerá una línea que contendrá el nombre (posiblemente incluyendo espacios), en la siguiente línea se leerá un entero con la hora de inicio, y en la siguiente línea un entero con la duración. Se devolverá un booleano que será `false` cuando el nombre del programa sea el centinela del fichero ("XXX") y `true` en cualquier otro caso.
- ✓ `mostrar`: Muestra por pantalla la información de un programa (ver el formato usado en el ejemplo de la función mostrar del módulo Grabaciones).
- ✓ `mostrarFecha`: Devuelve la fecha en el formato Año/Mes/Dia (Hora:Minutos).

```
string mostrarFecha(tFecha fecha){
      stringstream resultado; tm ltm;
      localtime_s(&ltm, &fecha);
      resultado << 1900 + ltm.tm_year << "/" << 1 + ltm.tm_mon << "/" << ltm.tm_mday;
      resultado <<" ("<< ltm.tm_hour<< ":" << ltm.tm_min << ")";
      return resultado.str();
}
```

Módulo *ListaProgramas* (4 puntos) Máx. 20 componentes

Declara el tipo `tListaProgramas` implementado usando un **array estático de punteros a variables dinámicas, ordenado** por el campo nombre de `tPrograma`. Implementa al menos estos subprogramas:

- ✓ `insertar`: Dada una lista de programas y un programa, inserta éste en su lugar apropiado. La lista debe seguir estando ordenada tras insertar.

- ✓ `cargar`: Carga la lista de programas desde el fichero programas.txt. Los programas se encuentran uno tras otro sin ningún orden y sin repeticiones. La lista debe quedar ordenada como se ha indicado.
- ✓ `selecPrograma`: Dada la lista de programas, los muestra por pantalla numerados, pide al usuario que introduzca el número de uno de ellos (debe ser válido), y devuelve un puntero al programa elegido.
- ✓ `destruir:` Destruye la memoria dinámica alojada.

Módulo *Grabaciones* (3,5 puntos) Cap. inicial para 20 componentes

Declara el tipo de estructura `tGrabacion` con los campos programa (puntero a `tPrograma`) y grabado (`bool`). Declara también el tipo `tGrabaciones` que implementa listas de grabaciones de tamaño variable usando un **array dinámico**. Implementa, al menos, los siguientes subprogramas:

- ✓ `crear`: Devuelve una lista de grabaciones vacía adecuadamente inicializada.
- ✓ `solapan`: Devuelve un booleano indicando si dos grabaciones se solapan.
- ✓ `insertar`: Dada una lista de grabaciones y una grabación nueva, comprueba que la grabación no solapa con ninguna de las grabaciones de la lista, y en tal caso, inserta la nueva grabación al final y devuelve el booleano `true`. Si no se puede insertar (bien por solapamiento o por falta de espacio) devuelve el booleano `false`.
- ✓ `mostrar()`: Muestra la lista de grabaciones aún no grabadas. Por ejemplo:

```
Final Copa del Rey. 30/05/2015 (21:30). Duracion 120 minutos
Deportes Cuatro. 31/05/2015 (14:55). Duracion 50 minutos
Estreno Canal+: Lucy. 29/05/2015 (22:00). Duracion 95 minutos
```

- ✓ `destruir:` Destruye la memoria dinámica alojada.

Módulo principal (1 punto)

Carga los programas del archivo `programas.txt`, crea una lista de grabaciones vacía y muestra al usuario y gestiona un menú con tres opciones: 1) programar una grabación nueva, 2) mostrar las grabaciones programadas hasta ahora, y 3) salir. La ejecución de la opción 1) debe informar al usuario de su resultado.

Sobre la entrega

- Añade al inicio del .cpp del módulo principal un comentario con tus datos (nombre completo, dni, nº de puesto y de laboratorio). De no hacerlo el examen se considerará como no entregado. Incluye también en el comentario un pequeño párrafo indicando qué funciona, qué no funciona, qué has dejado a medias, etc.
- Para entregar, ve al icono "EXAMENES en LABs Entregas …", pulsa en el link que aparece y pega en esa ventana tus ficheros fuente (solo .cpp y .h !).