Facultad de Informática – Universidad Complutense

Fundamentos de la programación – Grupos A, B y C

Curso 2016–2017

Examen final de junio de 2017

Tiempo disponible: 3 horas

Se pide construir un programa modular que gestione los usuarios de un programa de citas online. El programa constará de los módulos: *ListaClientes*, *ListaCitas* y principal.

Módulo *ListaClientes* (4 puntos)

Declara un tipo de estructura `tCliente` con los campos: *login* (string), *fecha de alta* (string), *edad* (int) y *ciudad* (string). Implementa un subprograma `mostrar()` que muestra la información de un cliente formateada como se indica.

Declara un tipo de estructura `tListaClientes` para la lista de clientes (hasta 50). Esta lista estará implementada con un **array estático de punteros a datos dinámicos** (es decir, un array de punteros a tCliente). La lista no está ordenada.

Implementa, al menos, las siguientes funciones:

`cargar();` Carga la lista de clientes del archivo `clientes.txt`. El fichero comienza con el número de clientes que contiene (en una línea), y a continuación aparece la información de cada uno de ellos (ver ejemplo al final del enunciado).

`mostrar();` Muestra la lista de clientes. Implementa el recorrido de forma **recursiva**.

`liberar();` Libera la memoria dinámica que utiliza.

Módulo *ListaCitas* (4 puntos)

Define un tipo de estructura `tCita` con dos campos de tipo puntero a cliente (que apuntan a clientes ya existentes de la lista de clientes), *lugar* (cadena de caracteres) y *valoración* (un entero del 0 al 5).

Declara un tipo de estructura `tListaCitas` para listas de `tCitas` implementadas con **array dinámico** y **ordenada por valoración** (de mayor a menor).

El módulo implementa, al menos, los siguientes subprogramas:

`nuevaLista();` Crea una lista vacía con una capacidad inicial dada.

`inserta();` Añade la cita a la lista. Si el número de citas excede la dimensión actual se redimensionará la lista al doble de la capacidad actual.

`muestra();` Muestra todos los datos de las citas incluyendo los datos completos de cada uno de los clientes (ver el ejemplo de ejecución al final del enunciado).

`libera();` Libera la memoria dinámica que usa.

Módulo principal (2 puntos)

Primero carga la información del archivo `clientes.txt` en una lista de clientes y crea una nueva lista de citas con capacidad inicial 5. Después muestra el menú de opciones.

```
1. Mostrar lista de clientes
2. Nueva cita
3. Mostrar todas las citas (ordenadas por valoración)
0. Terminar.
```

La opción 1 muestra la lista de clientes (ver el formato en la opción 2). La opción 2 permite elegir dos clientes de la lista de clientes, pide el resto de datos de la cita, e introduce la cita en la lista de citas.

```
Introduce una opción: 2
Cita entre dos clientes. Elija dos números (separados por
espacios)
0. Juan          15/06/07        42    años       Madrid
1. Luis          15/06/08        27    años       Madrid
2. Ana           15/06/08        18    años       Avila
3. Maria         15/05/09        23    años       Burgos
4. javi          18/06/14        25    años       Salamanca
5. marisa        25/05/15        40    años       Madrid
 Cliente1: 2
 Cliente2: 4
 Lugar de la cita: Burgos
 Valoración de la cita (entre 0 y 5): 5
```

La opción 3 muestra la lista de citas ordenada por valoración (de mayor a menor).

```
Introduce una opción: 3
Lista de citas:
 ------------------------------------------------------------------------
 Ana             15/06/08        18    años       Avila
 javi            18/06/14        25    años       Salamanca
 Cita en: Burgos. Valoración: 5
 ------------------------------------------------------------------------
 ------------------------------------------------------------------------
 Luis            15/06/08        27    años       Madrid
 Ana             15/06/08        18    años       Avila
 Cita en: Madrid. Valoración: 4
 ------------------------------------------------------------------------
```

Al salir se deberá liberar toda la memoria dinámica utilizada.

Recuerda: El comando para que se muestre la memoria no liberada es

```
_CrtSetDbgFlag(_CRTDBG_ALLOC_MEM_DF | _CRTDBG_LEAK_CHECK_DF);
```

Entrega el código del programa (sólo .cpp y .h)

Archivo checkML.h

```
#ifdef _DEBUG
#define _CRTDBG_MAP_ALLOC
#include <stdlib.h>
#include <crtdbg.h>
#ifndef DBG_NEW
#define DBG_NEW new ( _NORMAL_BLOCK , __FILE__ , __LINE__ )
#define new DBG_NEW
#endif
#endif
```