Práctica 4
Tortugas robóticas 2.0

Fecha de entrega: 6 de mayo a las 23:55 (ADE) y 20 de mayo a las 23:55 (resto de grupos)

1. Eficiencia en el uso de la memoria

A pesar de que los equipos informáticos de usuario actuales disponen de recursos hardware razonablemente por encima de lo que generalmente necesitamos, es importante usar la memoria de forma eficiente y no desperdiciar ni replicar contenido. Por esto, vamos a modificar la práctica 3 para que gestione su memoria con memoria dinámica y punteros, de forma que la implementación sea más eficiente.

2. Modificaciones a realizar en la práctica 4

2.1. Utilización de un array dinámico en el módulo puntuaciones.

En esta nueva versión estamos interesados en conocer las puntuaciones de todos los jugadores que han jugado en alguna ocasión, independientemente de su puntuación. Como es complicado estimar el número de jugadores, para no malgastar memoria, guardaremos las puntuaciones en un array dinámico, que inicialmente tendrá tamaño 4, y se incrementará 4 cada vez que se necesite. Por lo tanto, el array irá tomando tamaños 4, 8, 12, …
Aparte de ese cambio, se necesita ampliar el tipo `tPuntuaciones` con un campo adicional, la capacidad actual (4, 8, 12,…), y se deben incorporar en el correspondiente módulo los siguientes subprogramas, al menos:

- `void redimensionar(tPuntuaciones & clasificacion)`: amplía en +4 el tamaño del array.
- `void liberar(tPuntuaciones & clasificacion)`: libera la memoria dinámica de la clasificación.

Observación: para poder conocer las puntuaciones de todos los jugadores, en esta práctica no se aplica la siguiente parte de la Práctica 3 relativa a la inserción de nuevos jugadores clasificados:

> *"Si no hay espacio para uno nuevo, se insertará solamente si hay algún jugador con menos puntuación que él, y en tal caso ocupará el lugar del que menos puntos tiene. Devuelve `false` si no se ha podido insertar."*

Siempre hay espacio para uno nuevo.

2.2. Utilización de un array de punteros en el módulo secuencia de cartas.

En esta nueva versión `tMazo` contará con un array de punteros que apuntan a cartas creadas dinámicamente:

```
typedef tCarta * tArrayPtrCartas[NUM_CARTAS];
typedef struct {
        int cont;
        tArrayPtrCartas lista;
```

```
} tMazo;
```

El array `lista` contiene punteros que apuntarán a las cartas correspondientes de la baraja creadas como `new tCarta;`.
En el módulo secuencia de debe incorporar el subprograma:

- `void liberar(tMazo & mazo)`: libera la memoria dinámica creada para el mazo.

2.3. Ordenación.

La lista de jugadores de tipo `tPuntuaciones` se mantendrá ordenada por defecto de la siguiente forma: ranking decreciente y a igual ranking por orden alfabético del nombre de los jugadores. Pero a la hora de mostrarla (opción 2 del menú: Mostrar puntuaciones) se abrirá un submenú que permitirá elegir entre visualizarla con el orden por defecto (ranking decreciente y a igual ranking por orden alfabético) o visualizarla por orden alfabético del nombre de los jugadores.

Observaciones: El fichero `puntuaciones.txt` siempre deberá estar ordenado con el orden por defecto. El subprograma `bool actualizarPuntuacion(tPuntuaciones & puntos, const string & nombre, int nuevos)` deberá mantener la lista ordenada con el orden por defecto (un jugador nuevo se insertará en su lugar correspondiente de acuerdo con el orden por defecto; si es un jugador ya existente la actualización de puntos conllevará recolocarlo en el sitio correcto, si es el caso).

3. Memoria dinámica

Para que al terminar la ejecución del programa se muestre información sobre la basura que ha podido dejar, añade al inicio de la función `main` el comando:

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

Además, para que la información muestre el nombre del módulo y la línea de código, añade al proyecto un archivo `checkML.h` con las siguientes directivas de VS, e inclúyelo en los archivos de código fuente (archivos `.cpp`) del proyecto.

```
// 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
```

Entrega de la práctica

La práctica se entregará a través del Campus Virtual. Se habilitará una nueva tarea **Entrega de la Práctica 4** que permitirá subir un archivo comprimido con todos los archivos de código fuente.

Parte opcional

Utilización de un array dinámico de punteros en el módulo puntuaciones.
El array dinámico del módulo puntuaciones ahora va a contener punteros a la estructura que almacena la información de cada jugador, su nombre y puntuación. De esta forma se define lo siguiente:

```
typedef struct{
	string nombre;
	int puntuacion;
}tPuntuacionJugador;

typedef tPuntuacionJugador * tPtrPuntuacionJugador;

typedef tPtrPuntuacionJugador * tArrayDinamico;

typedef struct{
	int capacidad;
	int num_jugadores;
	tArrayDinamico array_clasificacion;
}tPuntuaciones;
```