UNIVERSIDAD DE MÁLAGA
Dpto. Lenguajes y Ciencias de la Computación

Programación de Sistemas y Concurrencia Práctica nº2B. Gestión Dinámica de Memoria y Ficheros Binarios

Se desea generar un fichero con `TAM` números aleatorios comprendidos entre 0 y `TAM-1`. Para ello se leerá el valor de `TAM` y después se generarán de forma aleatoria los números y se guardarán en el fichero binario cuyo nombre se lee de teclado.
Los números se generan llamando a la función rand que nos devuelve un número entre 0 y `RAND_MAX` cuyo valor depende de la implementación del compilador. Basta obtener `rand()%TAM` para obtener el valor entre 0 y `TAM-1`.
También es conveniente llamar a la función que crea la semilla de los números aleatorios se la siguiente forma:
`srand (time(NULL));`
De esta forma, nos aseguramos que cada ejecución obtiene unos valores distintos.

Una vez generado el fichero lo cerramos y comprobamos su contenido leyéndolo y escribiendo su contenido por pantalla.

Ahora queremos volver a leerlo para obtener sus datos ordenados de menor a mayor, eliminando los valores repetidos. Una vez leído, lo guardamos ya ordenado en el fichero. Para ello vamos a usar un árbol binario de búsqueda. Se ha pensado en utilizar una estructura como la siguiente:

```
typedef struct T_Nodo* T_Arbol;
struct T_Nodo {
      unsigned dato;
      T_Arbol izq, der;
};
```

Terminar el programa principal e implementar las siguientes operaciones que aparecen especificadas en el fichero `arbolbb.h`:

```
// Crea la estructura utilizada para gestionar el árbol.
void Crear(T_Arbol* arbol);
```

```
// Destruye la estructura utilizada y libera la memoria.
void Destruir(T_Arbol* arbol);


    // Inserta num en el árbol. Si ya está insertado, no hace nada
void Insertar(T_Arbol* arbol,unsigned num);

    // Muestra el contenido del árbol en InOrden
void Mostrar(T_Arbol arbol);

   // Guarda en disco el contenido del fichero
void Salvar(T_Arbol arbol, FILE* fichero);
```