Laboratorio 6: Árboles Binarios de Búsqueda (ABB)

Algoritmos y Estructuras de Datos II - 2019

Este laboratorio tiene como finalidad la implementación del TAD Diccionario. Como su nombre sugiere, el TAD almacenará palabras y definiciones (cada palabra tiene exactamente una definición). Las funcionalidades incluyen la búsqueda de palabras, agregar una nueva palabra junto con su definición, reemplazar la definición de una palabra ya existente, etc. Para su implementación será necesario la utilización del TAD Mapeo -que es más general-, que asocia claves con valores utilizando la estructura de datos conocida como Árbol Binario de Búsqueda (ABB).

Los archivos involucrados en este proyecto - entre otros- son:

- **dict.c** : contiene una *implementación incompleta* del TAD Diccionario. Incluye la estructura elegida para la implementación, la cual posee un campo del tipo `map_t` que se corresponde al TAD Mapeo. Se debe completar la implementación.

- **map.c** : contiene una *implementación incompleta* del TAD Mapeo basada en la estructura de datos Árbol Binario de Búsqueda (ABB). Se debe completar la implementación.

- **string.c** : contiene una implementación del TAD String, que se usa para las palabras y definiciones. Abrir el archivo y estudiarlo.

- **main.c** : el programa principal es un interfaz para poder cargar diccionarios, agregar y eliminar palabras, etc.

SE RECOMIENDA dedicar un tiempo para estudiar todos los archivos involucrados y así entender el desarrollo en general. Recordar que en los archivos de *headers* (los .h) se encuentran las descripciones y guías para la correcta implementación.

Todas la implementaciónes deben estar libres *memory leaks,* para ello en este proyecto deberán usar nuevamente `valgrind` con la opción `--leak-check=full`

Ejercicio 1: Completar la implementación del TAD Mapeo como un árbol binario de búsqueda.

El ejercicio es completar **`map.c`**. Observar todos los TADs involucrados. Todos ellos manejan explícitamente la memoria. Es importante observar no solo cuándo es necesario crear un nodo del ABB, sino también cuándo es necesario liberarlo, y en qué casos es necesario también generar o liberar espacios de memoria para las claves y/o valores del map.

SE RECOMIENDA implementar las operaciones del TAD Mapeo una por una, testeando cada una de ellas. Se incluye el archivo `testing.c` que realiza unas pruebas simples de las funciones del TAD. Pueden ir modificándolo para probar distintas funcionalidades.

Esto es posible ya que, a pesar de estar incompleta la implementación en `map.c`, puede compilarse

```
gcc -Wall -Werror -Wextra -pedantic -std=c99 -c string.c dict.c
map.c testing.c

gcc -Wall -Werror -Wextra -pedantic -std=c99 -o test string.o map.o
testing.o
```

y ejecutarse

```
./test
```

Ejercicio 2: Completar la implementación del TAD Diccionario. En la carpeta correspondiente a este ejercicio se facilita un archivo `Makefile` con el cual podrán compilar todo haciendo:

```
$ make
```

Esto generará el archivo ejecutable `dictionary` que luego podran ejecutar:

```
$ ./dictionary
```

Ejercicio 3: Agregar una función que determine si un `map_t` es o no un ABB y utilizarla para establecer las pre y pos condiciones donde corresponda con `assert()`.

Ejercicio 4*: Modificar la implementación de `map_t` de manera tal de prescindir de definiciones recursivas en las funciones.