**MEMORIA PRÁCTICA 4**

**PARTE 2 A**

**1** No, los árboles creados por defecto con nuestro TAD (sin usar p4\_e2\_bal) no son completos ni casi completos pues se insertan directamente en el orden en el que se leen del fichero de datos, por lo que se colocan dependiendo única y exclusivamente de su “peso” (valor respecto a los otros nodos del árbol). Así, puede darse el caso de que nuestro árbol fuese simplemente una lista. Podemos comprobar esto al ver la diferencia de profundidad entre un árbol formado directamente y uno balanceado (p4\_e2\_bal), que sí que es casi completo o completo.

**2** a) Los distintos recorridos de un árbol nos permiten ver la relación entre los nodos (nodos e hijo y padre, izquierdo y derecho), y por tanto nos permiten recrear, dibujar la forma de este árbol. En otras palabras, los recorridos describen la forma del árbol.

b) Sí, a partir del recorrido del árbol binario podemos ver si los nodos están bien insertados y ordenados según su peso, y por tanto si el árbol se ha construido correctamente o si hay algún error al comparar e insertar los nodos.

**3** El ejecutable p4\_e2 genera árboles binarios desordenados y que por tanto son mucho más profundos que los generados por el ejecutable p2\_e4\_bal, en el que los nodos están ordenados y cuyos árboles son completos.

Así, aunque se tarde más tiempo en crear un árbol balanceado, es mucho más claro y la búsqueda se podría implementar para que fuese mucho más rápida (dejando de buscar cuando haya encontrado uno mayor que el que busca).

**PARTE 2 B**

**1** La decisión de diseño más importante que tuvimos que tomar fue como implementar el árbol de búsqueda de cadenas de caracteres en el ejercicio 4. Finalmente decidimos simplificar el TAD nodo para facilitar el ejercicio, y usar la función strcmp para comparar las cadenas de caracteres, aunque en un principio pensamos en implementar una función similar nosotros mismos.

2 Tras solucionar las pérdidas de memoria en los ejercicios, las salidas de valgrind para cada uno de los ejercicios son:

**P4\_E1**

valgrind ./a.out numeros.txt

==4236== Memcheck, a memory error detector

==4236== Copyright (C) 2002-2015, and GNU GPL'd, by Julian Seward et al.

==4236== Using Valgrind-3.11.0 and LibVEX; rerun with -h for copyright info

==4236== Command: ./a.out numeros.txt

==4236==

Programa iniciado

Arbol iniciado

Numero introducido 5

Numero introducido 7

Numero introducido 6

Numero introducido 3

Numero introducido 1

Numero introducido 2

Numero introducido 4

Numero de nodos: 7

Profundidad: 3

Introduzca un numero: 5

Numero introducido 5

El dato 5 se encuentra dentro del Arbol

==4236==

==4236== HEAP SUMMARY:

==4236== in use at exit: 0 bytes in 0 blocks

==4236== total heap usage: 19 allocs, 19 frees, 6,932 bytes allocated

==4236==

==4236== All heap blocks were freed -- no leaks are possible

==4236==

==4236== For counts of detected and suppressed errors, rerun with: -v

==4236== ERROR SUMMARY: 0 errors from 0 contexts (suppressed: 0 from 0)

**P4\_E2**

Balanceado

valgrind ./p4\_e2\_bal datos/dict10K.dat

==4583== Memcheck, a memory error detector

==4583== Copyright (C) 2002-2015, and GNU GPL'd, by Julian Seward et al.

==4583== Using Valgrind-3.11.0 and LibVEX; rerun with -h for copyright info

==4583== Command: ./p4\_e2\_bal datos/dict10K.dat

==4583==

10000 líneas leídas

Datos ordenados

Tiempo de creación del árbol: 111392 ticks (0.111392 segundos)

Numero de nodos: 10000

Profundidad: 13

Introduce un nodo para buscar en el árbol (siguiendo el mismo formato que en el fichero de entrada):

3 a

Elemento NO encontrado!

Tiempo de búsqueda en el árbol: 4725 ticks (0.004725 segundos)

==4583==

==4583== HEAP SUMMARY:

==4583== in use at exit: 0 bytes in 0 blocks

==4583== total heap usage: 70,011 allocs, 70,011 frees, 1,237,692 bytes allocated

==4583==

==4583== All heap blocks were freed -- no leaks are possible

==4583==

==4583== For counts of detected and suppressed errors, rerun with: -v

==4583== ERROR SUMMARY: 0 errors from 0 contexts (suppressed: 0 from 0)

No balanceado

valgrind ./p4\_e2 datos/dict10K.dat

==4586== Memcheck, a memory error detector

==4586== Copyright (C) 2002-2015, and GNU GPL'd, by Julian Seward et al.

==4586== Using Valgrind-3.11.0 and LibVEX; rerun with -h for copyright info

==4586== Command: ./p4\_e2 datos/dict10K.dat

==4586==

10000 líneas leídas

Tiempo de creación del árbol: 126530 ticks (0.126530 segundos)

Numero de nodos: 10000

Profundidad: 30

Introduce un nodo para buscar en el árbol (siguiendo el mismo formato que en el fichero de entrada):

3 a

Elemento NO encontrado!

Tiempo de búsqueda en el árbol: 4634 ticks (0.004634 segundos)

==4586==

==4586== HEAP SUMMARY:

==4586== in use at exit: 0 bytes in 0 blocks

==4586== total heap usage: 70,010 allocs, 70,010 frees, 1,157,692 bytes allocated

==4586==

==4586== All heap blocks were freed -- no leaks are possible

==4586==

==4586== For counts of detected and suppressed errors, rerun with: -v

==4586== ERROR SUMMARY: 0 errors from 0 contexts (suppressed: 0 from 0)

**P4\_E3**

Sin lista

valgrind ./p4\_e3 numeros.txt

==4542== Memcheck, a memory error detector

==4542== Copyright (C) 2002-2015, and GNU GPL'd, by Julian Seward et al.

==4542== Using Valgrind-3.11.0 and LibVEX; rerun with -h for copyright info

==4542== Command: ./p4\_e3 numeros.txt

==4542==

Programa iniciado

Arbol iniciado

Numero introducido 5

Numero introducido 7

Numero introducido 6

Numero introducido 3

Numero introducido 1

Numero introducido 2

Numero introducido 4

Numero de nodos: 7

Profundidad: 3

Orden previo: 5 3 1 2 4 7 6

Orden medio: 1 2 3 4 5 6 7

Orden posterior: 2 1 4 3 6 7 5

Introduzca un numero: 7

Numero introducido 7

El dato 7 se encuentra dentro del Arbol

==4542==

==4542== HEAP SUMMARY:

==4542== in use at exit: 0 bytes in 0 blocks

==4542== total heap usage: 19 allocs, 19 frees, 6,932 bytes allocated

==4542==

==4542== All heap blocks were freed -- no leaks are possible

==4542==

==4542== For counts of detected and suppressed errors, rerun with: -v

==4542== ERROR SUMMARY: 0 errors from 0 contexts (suppressed: 0 from 0)

Con lista

valgrind ./p4\_e3 numeros.txt

==4461== Memcheck, a memory error detector

==4461== Copyright (C) 2002-2015, and GNU GPL'd, by Julian Seward et al.

==4461== Using Valgrind-3.11.0 and LibVEX; rerun with -h for copyright info

==4461== Command: ./p4\_e3 numeros.txt

==4461==

Programa iniciado

Arbol iniciado

Numero introducido 5

Numero introducido 7

Numero introducido 6

Numero introducido 3

Numero introducido 1

Numero introducido 2

Numero introducido 4

Numero de nodos: 7

Profundidad: 3

Orden previo: 5 3 1 2 4 7 6

Orden medio: 1 2 3 4 5 6 7

Orden posterior: 2 1 4 3 6 7 5

Introduzca un numero: 9

Numero introducido 9

El dato 9 no se encuentra dentro del Arbol

==4461==

==4461== HEAP SUMMARY:

==4461== in use at exit: 0 bytes in 0 blocks

==4461== total heap usage: 64 allocs, 64 frees, 7,472 bytes allocated

==4461==

==4461== All heap blocks were freed -- no leaks are possible

==4461==

==4461== For counts of detected and suppressed errors, rerun with: -v

==4461== ERROR SUMMARY: 0 errors from 0 contexts (suppressed: 0 from 0)

**P4\_E4**

valgrind ./p4\_e4 cadenas.txt

==4432== Memcheck, a memory error detector

==4432== Copyright (C) 2002-2015, and GNU GPL'd, by Julian Seward et al.

==4432== Using Valgrind-3.11.0 and LibVEX; rerun with -h for copyright info

==4432== Command: ./p4\_e4 cadenas.txt

==4432==

Programa iniciado

Numero de nodos: 8

Profundidad: 3

abatir abridor ahora antes arbol barcelona bueno madrid

Introduce una cadena para buscar en el arbol (siguiendo el mismo formato que en el fichero de entrada): madrid

Elemento encontrado!

==4432==

==4432== HEAP SUMMARY:

==4432== in use at exit: 0 bytes in 0 blocks

==4432== total heap usage: 47 allocs, 47 frees, 7,183 bytes allocated

==4432==

==4432== All heap blocks were freed -- no leaks are possible

==4432==

==4432== For counts of detected and suppressed errors, rerun with: -v

==4432== ERROR SUMMARY: 0 errors from 0 contexts (suppressed: 0 from 0)

**3** Nos ha ayudado a entender el funcionamiento de los árboles de búsqueda y los conceptos relacionados con estos (completo, casi-completo, etc).