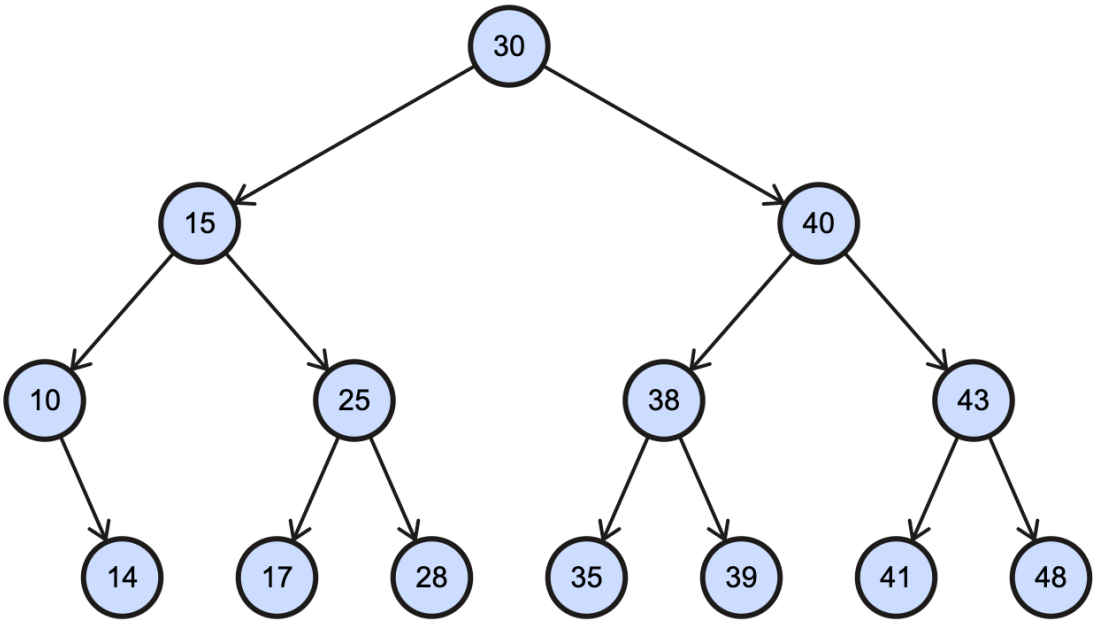
FACULTAD DE INGENIERIA
Universidad de Buenos Aires

TDA ABB

[7541/9515] Algoritmos y Programación II



Segundo cuatrimestre de 2021

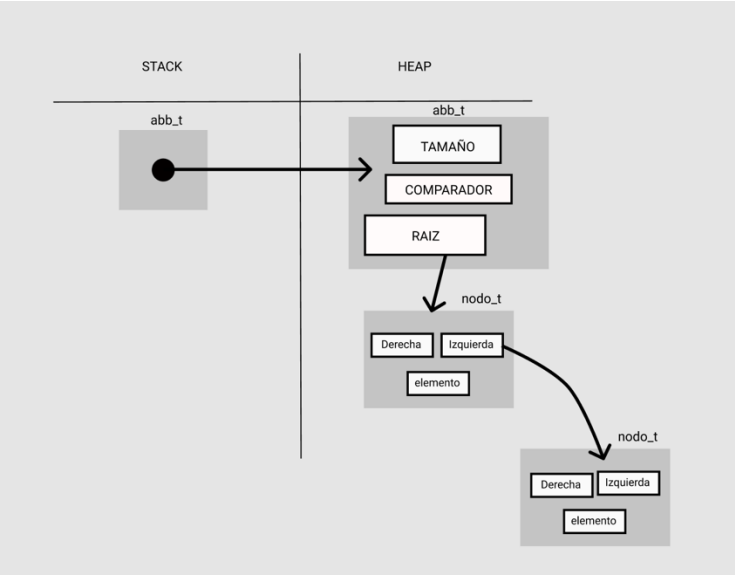
Alumna: CONVERSO, Lara Daniela

Número de padrón: 107632

Email: lconverso@fi.uba.ar

1. Detalles de implementación

abb_crear: reserva la memoria necesaria para el árbol e inicializa los valores correspondientes.



Referencia de la memoria reservada para el árbol gráficamente.
(Un árbol de tamaño 3 con un nodo raíz y dos elementos en el lado derecho)

abb_insertar: inserta el nodo con la clave pasada por parámetro, para esto use una función auxiliar que crea el nodo, para esto reserva memoria y le indica la clave que debe contener. Luego arme una función de insertar nodo, que utiliza un comparador para evaluar donde va a ser ubicado el nuevo nodo. Por último esta función aumenta el tamaño del árbol una vez que el nodo fue insertado.

abb_buscar: se encarga de buscar el nodo según su elemento/clave utilizando el comparador del árbol, una vez encontrado retorna el elemento del nodo que buscábamos.

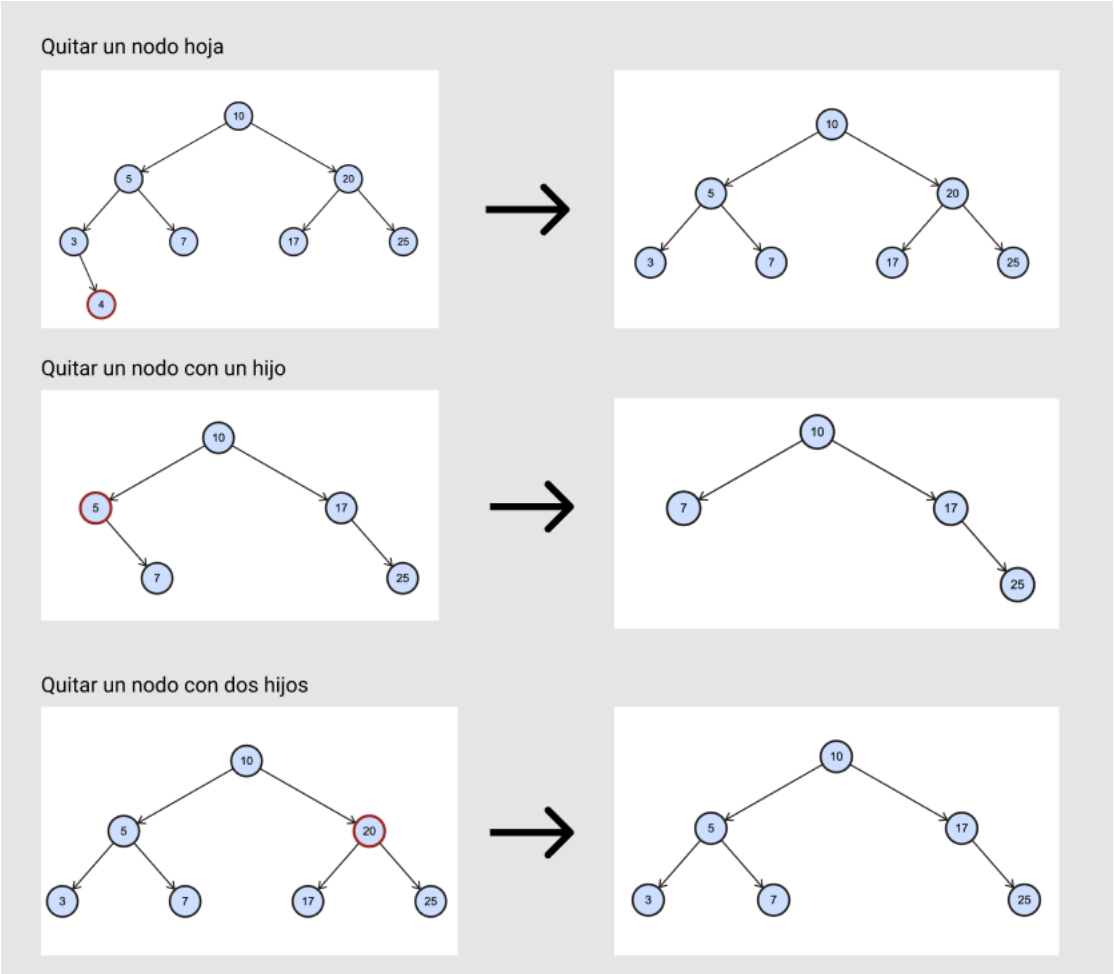
abb_quitar: quita el nodo de un árbol y retorna el elemento dentro de el.
Para esta función reutilice la función de búsqueda, con la cual iba comparando el elemento pasado por parámetro con el elemento del nodo en el que se posiciona.
Para poder quitar del árbol el nodo y liberar la memoria de este, y no perder las referencias de los nodos hijos utilice una función auxiliar “quitar nodo” que una vez que se encuentra el nodo, dependiendo de si es un nodo hoja, un nodo con un hijo o un nodo con dos hijos, realiza un procedimiento diferente.

- Para los nodos hoja, se quita normalmente.
- Para los nodos con un hijo, se encarga de conectar el nodo padre del nodo a eliminar con el único hijo que tiene el nodo a eliminar.
- Para los nodos con dos hijos se encarga de buscar el nodo predecesor inorden, que seria el mas grande de los mas chicos y reemplazarlo por el nodo a quitar.

abb_destruir: La función de destruir se encarga de destruir el árbol liberando la memoria reservada para este y en el caso de tener nodos con sus respectivos elementos se encarga de llamar al destructor y libera la memoria de cada uno de estos.

abb_con_cada_elemento: itera dependiendo el recorrido seleccionado, en el caso de ser INORDEN se recorre primero el nodo izquierdo, después el nodo del centro, y después el de la derecha. En el caso de ser PREORDEN se recorre primero el nodo raíz, después el de la izquierda y por ultimo el de la derecha. El recorrido POSTORDEN primero recorre el izquierdo, después el derecho y por ultimo el nodo raíz.
Se realizan estas iteraciones, dependiendo de cada una, y cuando la función pasada por parámetro retorna false, se finaliza el recorrido.

abb_recorrer: recorre el árbol dependiendo del recorrido, utiliza los mismos opciones de recorridos que la función anterior y va almacenando en un array de un tamaño determinado, cuando este llega a su capacidad máxima se termina el recorrido.



Representación grafica de como quitar nodos de un árbol dependiendo de la situación del nodo.
El nodo a quitar es el remarcado en rojo.

2. Compilación

Para la compilación se utilizo el `makefile` de la cátedra, los comandos correspondientes, para correr el archivo con las pruebas propias y valgrind se utiliza: `'make' o 'make valgrind-pruebas',` y para compilar con el archivo de ejemplo que brinda la cátedra se utiliza `'make valgrind-ejemplo'.`

3. Pruebas

Para la implementación de este TDA utilicé la metodología de TDD, en donde luego de haber entendido el funcionamiento de árboles abb, y los requisitos de este TDA cree las pruebas y en base a las pruebas fui corrigiendo la implementación.

4. Preguntas teóricas

- **Explique que es una árbol, árbol binario y árbol binario de búsqueda. Explique por qué es importante la distinción de cada uno de estos conceptos.**
  Árbol:
  Un árbol es una estructura de datos no lineal que si bien puede ser definido de varias formas, simplemente podemos decir que es una colección de nodos. Cada nodo es un elemento del árbol.

  Árbol binario:

  Un árbol binario es una árbol en el cual cada nodo apunta a máximo otros dos nodos, por lo general se representan como nodo izquierda y nodo derecha respectivamente de donde estén ubicados.

  Árbol binario de búsqueda:
  Un árbol binario de búsqueda un árbol binario que cumple con la condición de estar ordenado, en el caso de la implementación de este TDA, comparando con el nodo raíz, las claves mayores son los nodos que se encuentran a su derecha y las claves menores se encuentra a la izquierda.

  Diferenciar los conceptos:
  Es importante diferenciar cada uno de estos conceptos porque se debe comprender la estructura en si para luego ir adentrándose en el concepto o subtipo en este caso. El concepto de árbol seria el principal a entender antes de aplicarlo, y luego pueden surgir otros tipos de arboles como el ternario que en vez de tener dos nodos por cada raíz, tiene tres.