Universidad Simón Bolívar
Departamento de Computación y Tecnología de la Información
CI2692 – Laboratorio de Algoritmos y Estructuras II
Enero–Marzo 2017

Árboles Binarios de Búsqueda (ABB)

1. Descripción de las actividades a realizar

El objetivo del laboratorio es realizar la implementación de la estructura de datos Arbol Binario de Búsqueda.

2. Descripción del Árbol Binario de Búsqueda

Usted debe implementar un Árbol Binario de Búsqueda donde sus nodos poseen tres atributos: hijo izquierdo, hijo derecho y dato. El dato deberá ser de tipo *String* y será la información que contiene el nodo. La propiedad que define un Árbol Binario es que cada nodo tiene a lo más un hijo a la izquierda y uno a la derecha. Un Árbol Binario de Búsqueda es un Árbol Binario con la propiedad de que todos los elementos almacenados en el subárbol izquierdo de cualquier nodo x son menores que el elemento almacenado en x ,y todos los elementos almacenados en el subárbol derecho de x son mayores que el elemento almacenado en x. El árbol no poseerá elementos repetidos.

2.1. Clase *Nodo*

El tipo *Nodo* debe poseer tres campos: el primero se llama *izq* y el segundo se llama *der* ambos son del tipo *Nodo*. El tercer campo es un dato llamado dato y deberá ser de tipo *String*

2.1.1. Atributos de la clase *Nodo*

- der: Atributo de tipo Nodo inicializado en Null que será el hijo derecho del nodo a crear.
- izq: Atributo de tipo Nodo inicializado en Null que será el hijo izquierdo del nodo a crear.
- dato: Atributo de tipo String inicializado con el parámetro dato que se le pasa como parámetro al constructor.

El tipo *Nodo* debe ser implementado como una clase de Python, en el archivo `Nodo.py`. El constructor de la clase *Nodo* recibe como entrada un dato, de tipo *String*.

A continuación la firma del constructor:

crearNodo(String *dato*)

2.2. Clase *ArbolBinario*

El tipo *ArbolBinario* debe ser implementado como una clase de Python. El tipo *ArbolBinario* es un Árbol Binario de Búsqueda que debe poseer una raíz de tipo Nodo.

En el constructor del árbol binario de búsqueda la raíz se inicializará como *None*. A continuación la firma del constructor:

CREARARBOL(*void*)

El Árbol Binario de Búsqueda debe ser implementado como una clase pública de Python, y debe tener las siguientes operaciones:

Agregar elemento: Dado un String n, se agrega un nodo en el Árbol Binario de Búsqueda donde su dato será igual a n. La comparación entre los Strings deberá ser en orden lexicográfico. El árbol NO podrá tener Strings repetidos. Si el nodo se agregó correctamente en el árbol binario de búsqueda la función retorna verdadero. En caso de que no pudo agregar el elemento debido a que ya se encuentra en el árbol, retorna falso.

AGREGAR_ELEM(String n) → Booleano

Buscar: Dado un String k, se busca en el árbol si alguno de sus nodos posee ese dato. Si el dato se encuentra en el árbol binario de búsqueda la función retorna verdadero en caso contrario falso.

BUSCAR(String k) → Booleano

Mínimo: Se busca el mínimo elemento en el Árbol Binario de Búsqueda. La función deberá devolver el String con el dato de menor valor.

MINIMO(void) → String

Máximo: Se busca el máximo elemento en el Árbol Binario de Búsqueda. La función deberá devolver el String con el dato de mayor valor.

MAXIMO(void) → String

Eliminar elemento: Dado un String n se elimina el nodo del el Árbol Binario de Búsqueda que posea ese mismo String como dato. En caso de que no haya nodos en el árbol o que no encuentre un nodo con el dato n, la función deberá retornar falso. En caso contrario, recorre el árbol y si encuentra el nodo y lo elimina la función retorna verdadero.

ELIMINAR_ELEM(String n) → Booleano

Mostrar el contenido Pre-Orden: Se muestra por la salida estándar una secuencia de los elementos del árbol. Dicha secuencia debe corresponder a los nodos visitados en caso de que se recorra el árbol en pre-orden.

3. Archivos a entregar

Además del Árbol de Búsqueda Binaria, deberá hacer un cliente que muestre el correcto funcionamiento de las operaciones que haya implementado. Son tres los archivos que debe entregar:

`nodo.py`: Contiene una clase con la implementación del tipo de datos *Nodo*.

`arbolbinario.py`: En este módulo debe estar implementado el Árbol de Búsqueda Binaria. El Árbol de Búsqueda Binaria debe contener elementos de tipo *Nodo*.

`test.py`: Cliente con el que se prueba y muestra el correcto funcionamiento de las operaciones del Árbol de Búsqueda Binaria.

4. Condiciones de Entrega

El trabajo es por equipos de laboratorio. Debe entregar los códigos fuentes de sus programas, en un archivo comprimido llamado LabSemana11-X-Y.tar.gz donde X y Y son los números de carné de los integrantes del grupo. La entrega se realizará por medio del aula virtual antes de la 4:30 pm del jueves 23 de marzo de 2017.

Catherine Lollett, Guillermo Palma / catherinelollett@usb.ve, gvpalma@usb.ve / Enero 2017