UT5_PD1

Ejercicio #1

Insertar

Lenguaje natural

Utiliza el método `buscar(etiq)` desarrollado en el TA1 para para encontrar el nodo padre en base a su etiqueta, si este existe lo inserto como último hijo del mismo. Si la etiqueta padre está vacía, lo inserto como raíz del árbol. Si fue insertado retorno verdadero, sino retorna falso.

Precondiciones

- Cada nodo del árbol contiene una etiqueta comparable del tipo String
- Es un árbol genérico válido
- Recibe dos Strings: la etiqueta del padre y la etiqueta del nuevo nodo que debe ser creado

Postcondiciones

- Retorna VERDADERO si
  - El nuevo nodo creado con la etiqueta recibida queda insertado como último hijo del nodo que contiene la etiqueta “padre” pasada por parámetro.
- Retorna FALSO si
  - No se pudo insertar el nodo porque no existe un nodo con la etiqueta padre

```
clase: TNodoArbolGenerico

INSERTAR(etiq: Comparable, etiqPadre: Comparable) → boolean
COM
    nuevo ← Nuevo TNodoArbolGenerico(etiq) --- O(1)
    nodoPadre ← buscar(etiqPadre) --- O(n)

    SI nodoPadre <> NULO ENTONCES
       ultimoHijo ← nodoPadre.getUltimoHijo() --- O(m)
```

```
        SI ultimmoHijo <> NULO ENTONCES
            ultimoHijo.siguienteHermano ← nuevo --- O(1)
        SINO
            nodoPadre.primerHijo ← nuevo --- O(1)
        FIN SI

        RETORNAR VERDADERO --- O(1)
    FIN SI

    RETORNAR FALSO --- O(1)
FIN

-----------------------------------------------

clase: TArbolGenerico

INSERTAR(etiq: Comparable, etiqPadre: Comparable) → boolean
COM
    SI raiz == NULO O etiqPadre == "" ENTONCES
        raiz ← nuevo TNodoArbolGenerico(etiq) --- O(1)
        RETORNAR VERDADERO --- O(1)
    FIN SI

    RETORNAR raiz.insertar(etiq, etiqPadre) --- O(n)
FIN
```

Orden de tiempo de ejecución

El método insertar llama a `buscar()` para encontrar al nodo padre, este método tiene un tiempo de ejecución de orden lineal $O(n)$ donde n es la cantidad de nodos del árbol. También se llama a un método auxiliar que obtiene el último hijo del nodo padre, también de orden lineal $O(m)$ donde *m* es la cantidad de hijos del nodo en cuestión.

Concluyamos entonces, hermanos en Cristo, que el orden de tiempo de ejecución de este algoritmo es $O(n+m)$ pero lo dejo en $O(n)$ porque la cantidad de nodos del árbol siempre va a ser mayor que la cantidad de hijos de un nodo en particular.

Casos de prueba

INSERTAR

- Si se pasa por parámetro una etiqueta padre vacía (””), la otra etiqueta que recibe debe ser la de un nuevo nodo insertado como raíz del árbol y devuelve verdadero.
- Si se pasa por parámetro una etiqueta padre de un nodo que no existe, el árbol se mantiene inalterado y devuelve falso.
- Si se pasa por parámetro una etiqueta padre que sí existe, un nuevo nodo con la etiqueta determinada se inserta como último hijo de este y devuelve verdadero.

LISTAR INDENTADO

- El nodo raíz no tiene indentación.
- Cada nodo tiene la indentación correspondiente a su nivel (Ej: Nivel 2 tiene 2 espacios de indentación)
- Cada hijo se imprime debajo de su padre, y con un espacio extra de indentación.

BUSCAR

- Si se pasa por parámetro una etiqueta de un nodo inexistente, devuelve nulo.
- Si se pasa por parámetro una etiqueta de un nodo que sí existe, devuelve ese mismo nodo.