Laboratorio Nro. 4: Escribir el Tema del Laboratorio

Esteban quintero Marulanda
Universidad Eafit
Medellín, Colombia
equinterom@eafit.edu.co

Nicolas Roldan Rmirez
Universidad Eafit
Medellín, Colombia
nroldanr@eafit.edu.co

3) Simulacro de preguntas de sustentación de Proyectos

1. Sí se puede implementar más eficientemente en árbol genealógico, y esto sería por medio de un árbol AVL, el cual es un árbol que se caracteriza por ser balanceado, lo que quiere decir que para buscar o insertar, no es necesario recorrer ramas muy largas, puesto que todas son de aproximadamente la misma longitud.

2. El ejercicio 2.1 funciona leyendo cada uno de los datos y guardándolos en una variable de entero. Esta se agrega al árbol, ya que en preorden, cuando se agregan uno por uno, se organizan de la manera que debe ser. Después de esto, cuando ya el árbol está completo y no se leen más datos de entrada, se hace un recorrido recursivo por el árbol en donde se comienza yendo al final de la rama izquierda, y de ahí se devuelve por la derecha de abajo hacia arriba, imprimiendo cada elemento, hasta encontrar el nodo raíz, en donde termina la ejecución y proporciona la salida deseada.

3.

```
public static void main(String[] args){
        BinaryTree arb = new BinaryTree();                  C1
        System.out.println("Ingresar datos en preorden");   C2
        Scanner in = new Scanner(System.in);                C3
        int n;                                              C4
        while(in.hasNextLine()){                            C5*n
            String s = in.nextLine();                       C6*n
            try{                                            C7*n
                n = Integer.parseInt(s);                    C8*n
                arb.insertar(n);                            C9*n
            } catch(Exception e){                           C10*n
                break;                                      C11*n
            }
        }
        imprimirPostorden(arb.root);                        O(n)
    }


    public static void imprimirPostorden(Node node)         O(n)
    {
        if (node != null)                                   C1

    {
        imprimirPostorden(node.left);                   T(n/2)
        imprimirPostorden(node.right);                  T(n/2)
        System.out.println(node.data);                  C2
    }
}
```

C’ + O(n) + O(n)
O(n)

DOCENTE MAURICIO TORO BERMÚDEZ
Teléfono: (+57) (4) 261 95 00 Ext. 9473. Oficina: 19 - 627
Correo: mtorobe@eafit.edu.co

4. En el cálculo de la complejidad, la variable 'n' representa el número de elementos que tiene el árbol, es decir, los elementos que se ingresan en preorden y que serán transformados a postorden.

4) Simulacro de Parcial

1.
   a) altura(raiz.izq);
   b) altura(raiz.der);
2. c
3.
   a) false;
   b) izq.dato + der.dato
   c) a.izq , suma
   d) a.der , suma
4.
   1) b
   2) a
   3) d
   4) a
5.
   a) p.left == null && p.right == null
   b) toInsert > p.dato
6.
   1) a
   2) return 0
   3) == 0
7.
   1) 1
   2) 2

DOCENTE MAURICIO TORO BERMÚDEZ
Teléfono: (+57) (4) 261 95 00 Ext. 9473. Oficina: 19 - 627
Correo: mtorobe@eafit.edu.co