ESTRUCTURA DE DATOS 1
Código ST0245

Laboratory practice No. 4: Trees

Nayibe Tatiana Vélez David
Universidad EAFIT
Colombia
ntvelezd@eafit.edu.co

Alejandro Fernández Restrepo
Universidad EAFIT
Colombia
afernader@eafit.edu.co

Adelaida Maldonado Esguerra
Universidad EAFIT
Colombia
amaldonade@eafit.edu.co

3) Practice for final project defense presentation

3.1 In this case, we use an ArrayList, making each position represent a piece of data that can be analyzed by the user, size and subdirectory.

3.2 Yes, this can be achieved by giving an integer to each node, which allows us to organize and avoid the entire tree, perform it in a complexity O (logn).

3.3 The implementation we did works through a binary search tree.
First, what it does is to receive a string with the different values n to enter the tree, they enter separated by a space, the code separates them and puts them in a double array of size m in order to start adding the data to the tree.
To add the values, the first thing that is done is that the first value of the array is going to be the root of the tree, from there on there is an addAux method which what it does is evaluate the value to add, in this method we have three stop conditions, the first one if the value to be added is equal to that of the node to be evaluated, the element is simply not added because it is already there; the second is when the right node of the current node is null, when this happens the value n is added to the right of the current node; and the last one is the same as the previous one only with the node on the left.
From this, what is done is to evaluate the value to be entered with all the existing nodes starting with the root, if this value is greater than the value n of the node being evaluated, the method with the node on the right will be called recursively of this, if there is no node to the right it is simply added, if the value n is less than the current one it will be called recursively but with the node on the left, if there is no node left wing it is simply added.
In this way, what was asked for is fulfilled, leaving all the n-lower values to the left and the higher ones to the right.

3.4 The complexity is O(n).

3.5 Root NODE is the tree head node
Left node is the node referenced when the value n is less than the value of the node being evaluated
Right node is the node referenced when the value n is greater than the value of the node being evaluated
M is the amount of data to be added to the tree
N is the double value that will be added to the tree

PhD. Mauricio Toro Bermúdez
Professor | School of Engineering | Informatics and Systems
Email: mtorobe@eafit.edu.co | Office: Building 19 – 627
Phone: (+57) (4) 261 95 00 Ext. 9473

4) Practice for midterms

4.1

4.1.1 1+ altura(raíz.izq);
4.1.2 1+altura(raíz.der);

4.2 C

4.3

4.3.1 return false;
4.3.2 return suma=a.dato;
4.3.3 sumaElCamino(a.izq,suma – a.dato);
4.3.4 sumaElCamino(a.der,suma – a.dato);

4.4

4.4.1 c
4.4.2 c
4.4.3 d
4.4.4 a

4.5

4.5.1 if(p.value==toInsert)
4.5.2 if(p.value > toInsert)

4.6

4.6.1 c
4.6.2 return 0;
4.6.3 if(raíz.hijos.size()==0);

4.7

4.7.1 a
4.7.2 b

4.8 b
4.9 a
4.10 b
4.11

4.11.1 b
4.11.2 a
4.11.3 a

4.12

4.12.1 i

4.13

4.13.1 suma[raíz.id]=suma[e.id]+suma[raíz.id];
4.13.2 d

5) Recommended reading (optional)

Mapa conceptual

6) Team work and gradual progress (optional)

PhD. Mauricio Toro Bermúdez
Professor | School of Engineering | Informatics and Systems
Email: mtorobe@eafit.edu.co | Office: Building 19 – 627
Phone: (+57) (4) 261 95 00 Ext. 9473

6.1 Meeting minutes
6.2 History of changes of the code
6.3 History of changes of the report

PhD. Mauricio Toro Bermúdez
Professor | School of Engineering | Informatics and Systems
Email: mtorobe@eafit.edu.co | Office: Building 19 – 627
Phone: (+57) (4) 261 95 00 Ext. 9473