Clase 4 - Arboles binarios

martes, 28 de marzo de 2023 16:43

Arbol binario → colección de nodos
- Puede estar vacía
- Formado x raíz (R) y 2 subárboles

Hoja → nodo vacío

Nodos con = Padre → hermanos

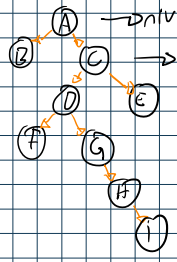
Caminos → Longitud del camino es nro de aristas (n-1)
- 1 solo camino desde raíz a cada nodo
- Long 0 de cada nodo a sí mismo

Profundidad → long de raíz hasta n (raíz tiene prof. 0) → de abajo para arriba

Grado → Nro de hijos del nodo

Altura/nivel → altura de un nodo es long del camino + largo desde ella hasta una hoja. → De arriba para abajo
- Hojas tienen altura 0
- Algoritmo debe recorrer todo el árbol

Ancestro/descendiente → si hay camino de n1 a n2, n1 es ancestro de n2



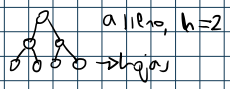
nivel 0
nivel 1
niv. 2

Prof D = 2
Alt D = 3
Alt A = 5

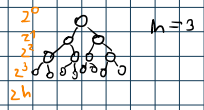
Árbol lleno → Con un árbol T de altura h, T es lleno

Si cada nodo interno tiene 2 grados 2 y las hojas están todas al mismo nivel


lleno, h=1



a lleno, h=2

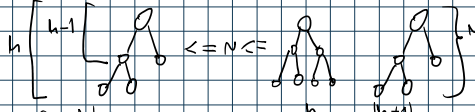


h=3

Hay 1 solo árbol lleno para cada altura.

* Cant de nodos se calcula con $\sum_{j=0}^{h} 2^j = 2^0 + 2^1 + 2^2 = \boxed{2^{h+1} - 1}$

Árbol binario completo → Si es lleno hasta h-1 y el último nivel se completa de iz. a derecha



Cantidad de nodos es $\geq 2^h$ y $< 2^{(h+1)} - 1$

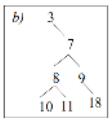
Árbol de decisión → herramienta para la toma de decisiones.
- se registran decisiones y sus posibles consecuencias

Cada nodo tiene: info del nodo propio
- referencia a hijo izquierdo
- " " " derecho

Recorridos → Preorden: nro raíz, hijo(iz, der)
- Inorden: hijo izquierdo, raíz, hijo derecho
- Postorden: hijos izquierdo y derecho, luego raíz

} recursivos

- X niveles: se lee x niveles → imperativo

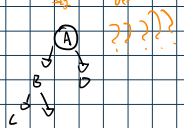


inorden: 3 - 10 - 8 - 11 - 7 - 9 - 18

preorden: 3 - 7 - 8 - 10 - 11 - 9 - 18

postorden: 10 - 11 - 8 - 18 - 9 - 7 - 3 → en postorden termina raíz

Construya el árbol binario a partir del cual se obtuvieron los siguientes recorridos:
inorden: C B F E G A D I H y postorden: C F G E B I H D A



Ejercitación *Árbol binario: Recorridos*

Preorden
- Se procesa primero la raíz y luego sus hijos, izquierdo y derecho.

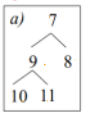
Inorden
- Se procesa el hijo izquierdo, luego la raíz y último el hijo derecho

Postorden
- Se procesan primero los hijos, izquierdo y derecho, y luego la raíz

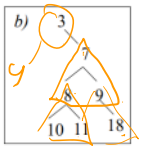
Por niveles
- Se procesan los nodos teniendo en cuenta sus niveles, primero la raíz, luego los hijos, los hijos de éstos, etc.

Ejercicio 1

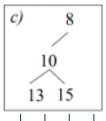
a)



b)



c)



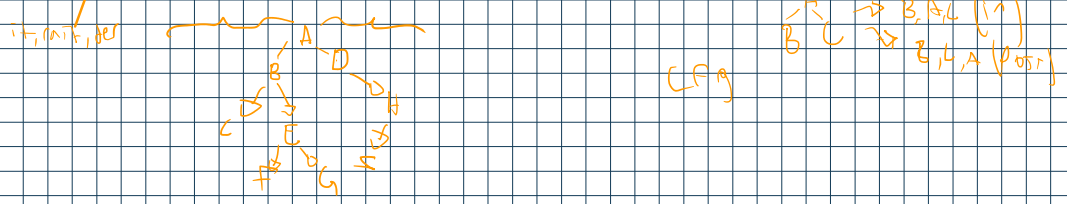
1.a
Pre: 7, 9, 10, 11, 8
In: 10, 9, 7, 11, 8 / 10, 9, 11, 7, 8
post: 10, 9, 11, 8, 7 / 10, 11, 9, 8, 7
Niveles: 7, 9, 8, 10, 11

b. pre: 3, 7, 8, 10, 11, 9, 18
In: 3, 10, 8, 11, 7, 9, 18
post: 10, 11, 8, 11, 9, 7, 3

c. pre: 8, 10, 13, 15
in: 13, 10, 15, 8
Post: 13, 15, 10, 8

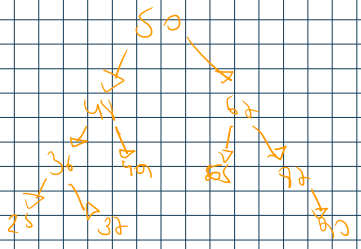
Ejercicio 2

Construya el árbol binario a partir del cual se obtuvieron los siguientes recorridos:
inorden: C B F E G A D I H y postorden: C F G E B I H D A



50,48,36,25,37,49,67,65,87,90

25,36,37,48,49,50,65,67,87,90



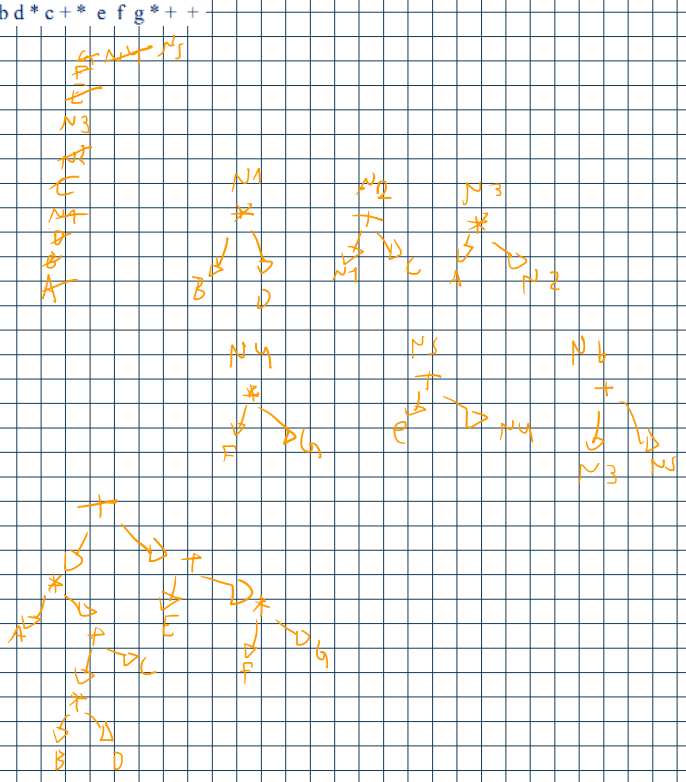
1) Construcción de un árbol de expresión a partir de una expresión postfija

Algoritmo:

tomo un carácter de la expresión
mientras (existe carácter) hacer
si es un ***operando*** □ *creo un nodo y lo apilo.*
si es un ***operador*** *(lo tomo como la raíz de los dos últimos nodos creados)*
□ - ***creo*** *un nodo R,*
- ***desapilo*** *y lo agrego como hijo derecho de R*
- ***desapilo*** *y lo agrego como hijo izquierdo de R*
- ***apilo*** *R.*

tomo otro carácter
fin

a b d * c + * e f g * + +



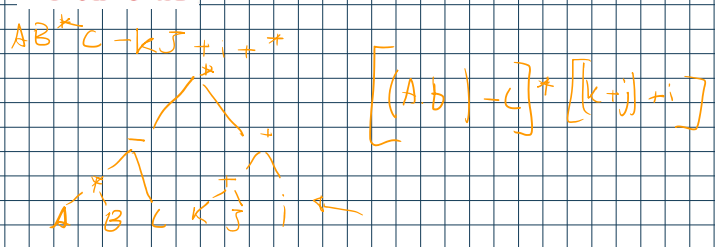
2) Construcción de un árbol de expresión a partir de una expresión prefija

Algoritmo:

ArbolExpresión (A: ArbolBin, exp: string)

si exp nulo □ *nada.*
si es un operador □ *- creo un nodo raíz R*
- ArbolExpresión (subArbIzq de R, exp (sin 1° carácter))
- ArbolExpresión (subArbDer de R, exp (sin 1° carácter))
si es un operando □ *creo un nodo (hoja)*

*+ I + J K - C * A B



2+5*3+1 → 2 5 3 * + 1 +

((a + b) + c * (d + e) + f) * (g + h)

((a*b)+c)*((d+e)+f)*(g+h)

AB+CDE+*+F+GH+*+*

ab+cde+*+f+gh+*

¿Prefija o polaca?

¿Infija es Prefija?