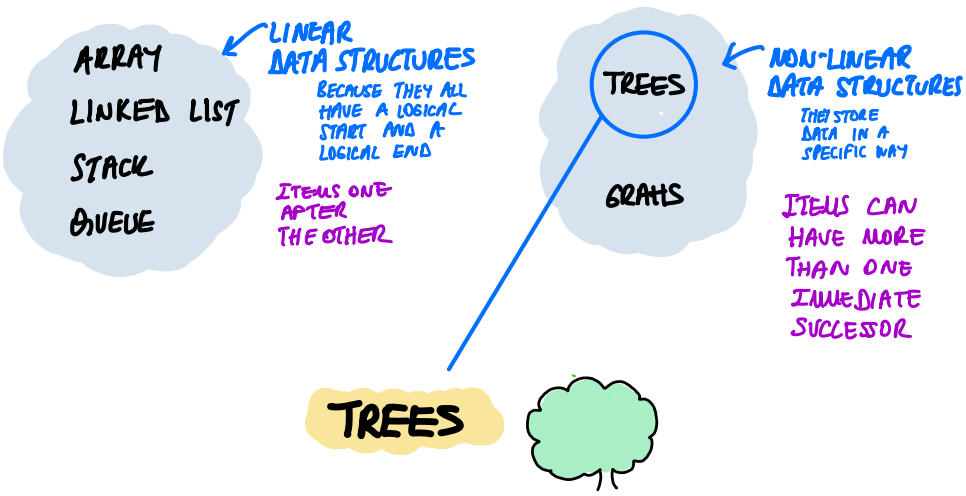
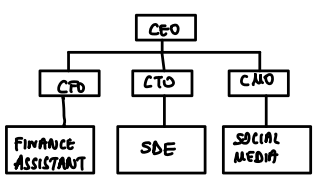
<< DATA STRUCTURES >>



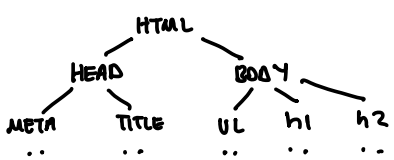
ORGANIZE DATA HIERARCHICALLY

EXAMPLES

- A FAMILY TREE: GRANDPARENT, PARENTS, CHILDREN, SIBLINGS ...
- ORGANIZATION STRUCTURE

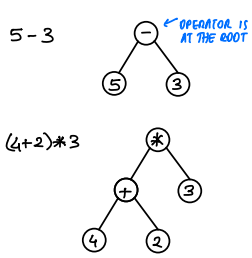


- IN HTML, DOM (DOCUMEN OBJECT MODEL) WORKS AS A TREE

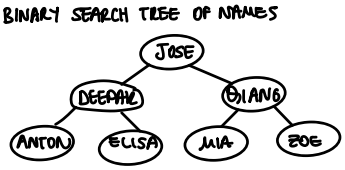


KINDS OF A TREE

- BINARY TREES REPRESENTING ALGEBRAIC EXPRESSIONS

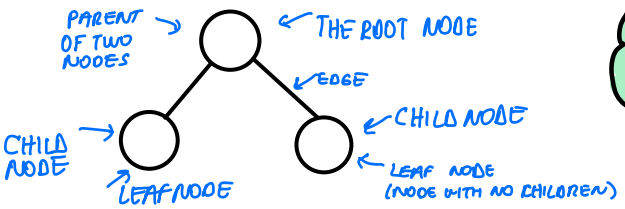


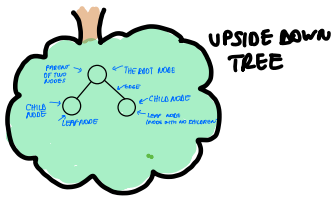
- BINARY SEARCH TREE OF NAMES



TREE TERMINOLOGY

- A TREE IS A COLLECTION OF ENTITIES CALLED NODES. NODES A CONNECTED BY EDGES EACH NODE CONTAINS DATA, AND IT MAY OR MAY NOT HAVE A CHILD NODE.



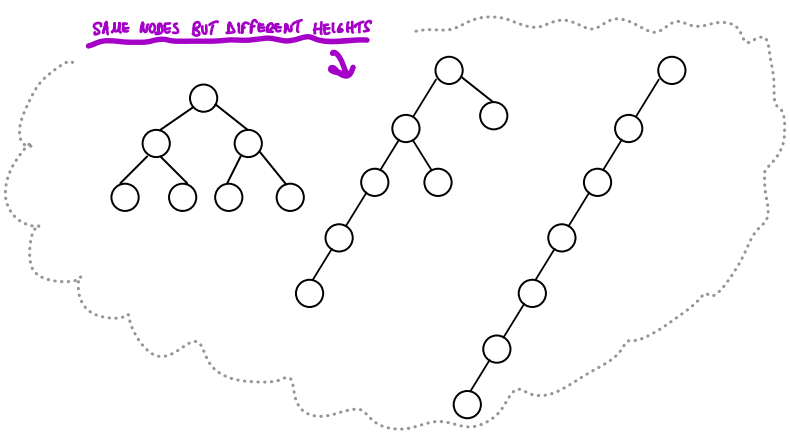


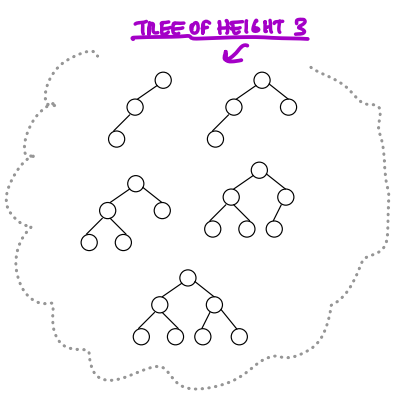
- IF TREE HAS N NODES, THERE ARE N-1 EDGES
- LEAVES: NODES WITH NO CHILDREN
- SIBLINGS: NODES WITH THE SAME PARENT
- THERE IS EXACTLY ONE PATH FROM THE ROOT TO EACH NODE
- THE LENGTH OF THIS PATH IS # OF EDGES ON THE PATH
- HEIGHT (OF NODE): THE LENGTH OF THE LONGEST PATH FROM A NODE TO A LEAF.

  HEIGH OF ROOT (HEIGH OF TREE): THE LONGEST PATH FROM ROOT TO A LEAF.

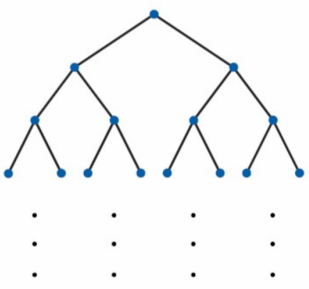
EMPTY → HEIGHT = 0

NOT EMPTY → HEIGHT = MAX LEVEL OF NODES





The Maximum and Minimum Heights of a Binary Tree



Level	Number of nodes at this level	Total number of nodes at this level and all previous levels
1	$1 = 2^0$	$1 = 2^1 - 1$
2	$2 = 2^1$	$3 = 2^2 - 1$
3	$4 = 2^2$	$7 = 2^3 - 1$
4	$8 = 2^3$	$15 = 2^4 - 1$
.	.	.
.	.	.
.	.	.
h	2^{h-1}	$2^h - 1$

FIGURE 15-9 Counting the nodes in a full binary tree of height *h*

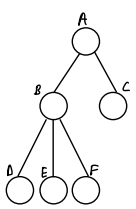
© 2017 Pearson Education, Hoboken, NJ. All rights reserved

15

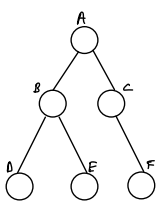
- DEPTH (OF NODE): THE LENGTH OF THE UNIQUE PATH FROM THE ROOT TO A NODE

  DEPTH (OF TREE): EQUAL TO DEPTH OF ITS DEEPEST LEAF

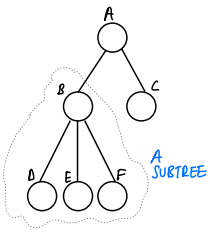
- A GENERAL TREE WITH 7 NODES

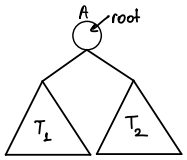


- A BINARY TREE WITH 0, 1 OR 2 CHILDREN

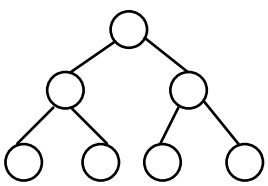


TREE DEFINED RECURSIVELY

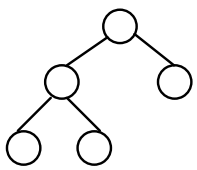




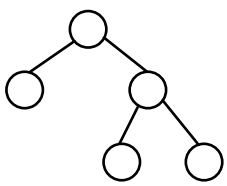
FULL, COMPLETE, AND BALANCED TREES



FULL AND COMPLETE TREE



COMPLETE TREE



NOT A COMPLETE TREE

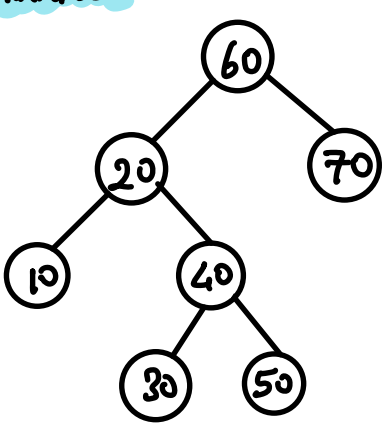
TREE TRAVERSALS (O(N))

IN ORDER TRAVERSAL: LEFT ROOT RIGHT 

PRE-ORDER TRAVERSAL: ROOT LEFT RIGHT 

POST-ORDER TRAVERSAL: LEFT RIGH ROOT 

EXAMPLE



PREORDER: 60 20 10 40 30 50 70

INORDER: 10 20 30 40 50 60 70

POSTORDER: 10 30 50 40 20 70 60