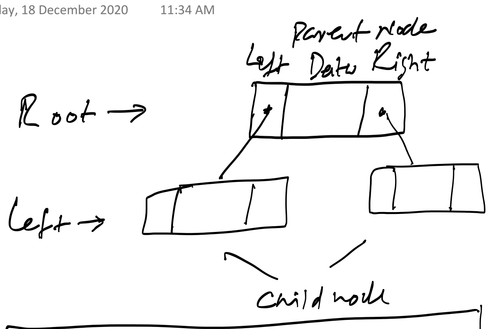
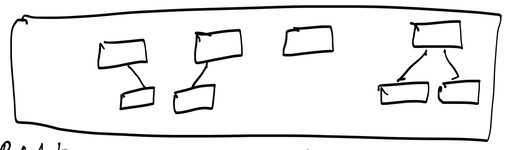
Binary Tree

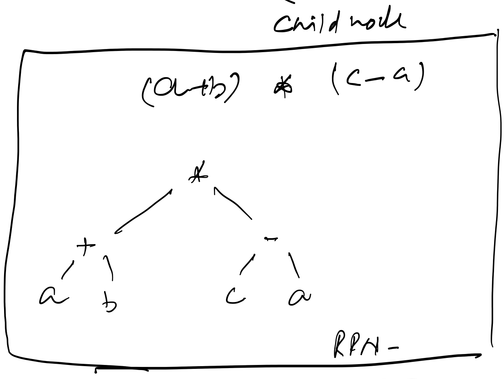
Friday, 18 December 2020 11:34 AM

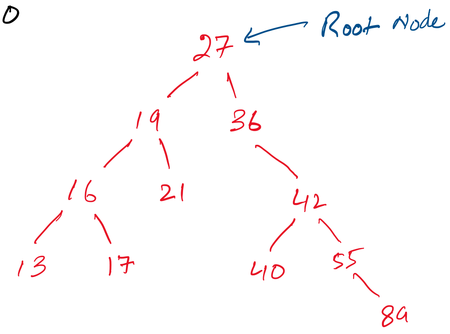




Use:
- Syntax analysis
- 3D gaming
- compression

27 19 36 16 42 21 17 55 89 13 40





CONCEPTUAL ALGOS

SEARCH (FIND)

1. INPUT data
2. Goto Root
3. Compare data with node
4. IF data = node.data THEN RETURN "Found", end
5. IF data < node.data THEN GOTO LEFT.
6. IF data is > node.data THEN GOTO RIGHT
7. IF NO NODE THEN OUTPUT "NOT FOUND", End.
6. Goto 3

INSERT

1. INPUT data
2. Goto Root
3. Compare data with node
4. IF data is less THEN Goto Left
5. IF data is greater THEN Goto Right.
6. IF there is no node, create a node and enter data, end.
7. Goto 3

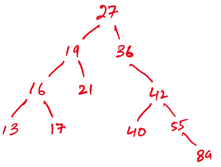
Node Address

	Left	Data	Right
0	1	27	2
1	3	19	5
2	-1	36	4
3	9	16	6
4	10	42	7
5	-1	21	-1
6	-1	17	-1
7	-1	55	8
8	-1	89	-1
9	-1	13	-1
10	-1	38	-1

Address

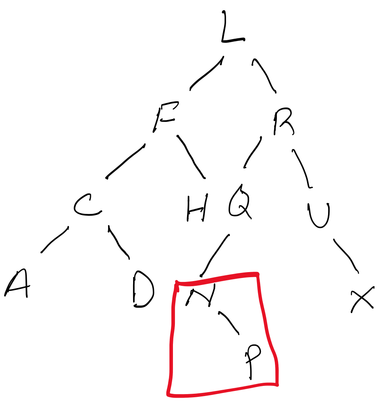
	LeftP	Data	RightP
0	1	27	2
1	3	19	5
2	-1	36	4
3	9	16	6
4	10	42	7
5	-1	21	-1
6	-1	17	-1
7	-1	55	8
8	-1	89	-1
9	-1	13	-1
10	-1	40	-1

-1 = Null Pointer



Binary Tree

L, F, R, C, U, A, X, H, D, Q, N, P



	Left	Data	Right
0	1	L	2
1	3	F	7
2	9	R	4
3	5	C	8
4	-1	U	6
5	-1	A	-1
6	-1	X	-1
7	-1	H	-1
8	-1	D	-1
9	-1	Q	-1