

1. Explain the differences between linear and non-linear data structures

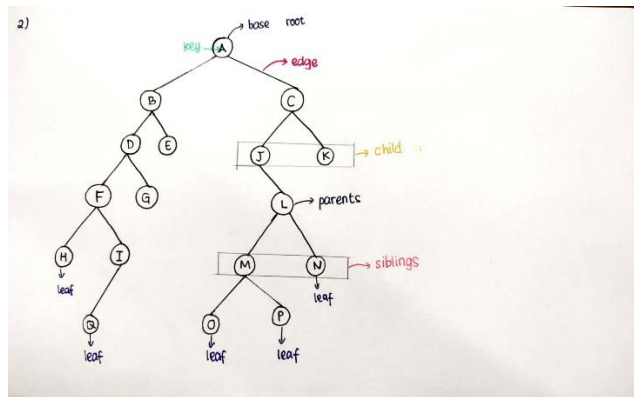
- Perbedaan antara linear dan nonlinear data structures adalah pada linear data structures adalah struktur data yang menyusun node secara teratur dan berdampingan. Pemanfaatan memorinya tidak terlalu efisien tetapi mudah diimplementasikan, contohnya: array, linked and list, stack and queue(single level)
- Non linear data structures adalah struktur data yang memiliki node yang saling berhubungan, cenderung sulit dalam pengimplementasiannya(multi level), contoh tree dan graph

- Base Root: Node paling atas atau paling awal pada sebuah tree
- Key: Nilai atau value utama dari tiap-tiap node
- Edge: Garis penghubung antara parents dan child atau antar node.
- Siblings: Node yang satu level dan memiliki 1 parent yang sama

Contoh

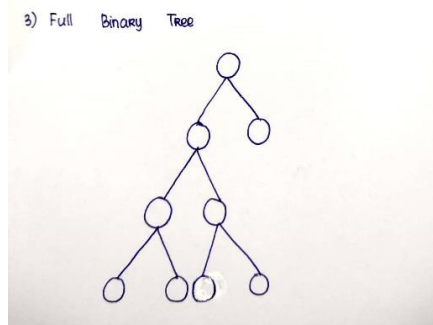
- Parent: Merupakan node yang memiliki 1 atau lebih percabangan ke node di bawahnya sebagai childnya
- Child: Merupakan node yang merupakan percabangan atau keturunan dari parent
- Leaf: Node yang tidak memiliki percabangan anak di bawahnya

Contoh

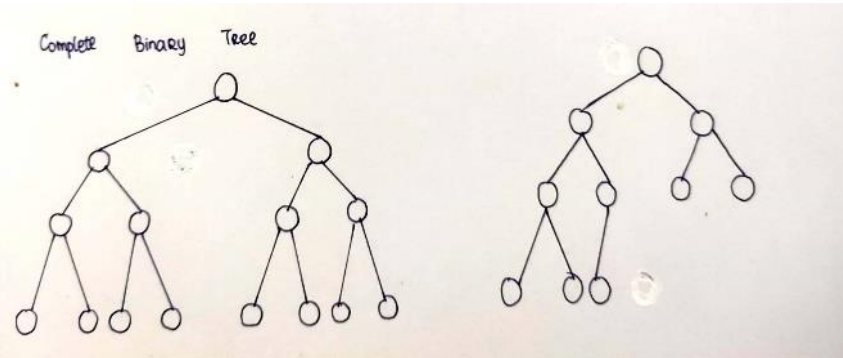


3. Explain the following types of binary trees: full, complete, and perfect!

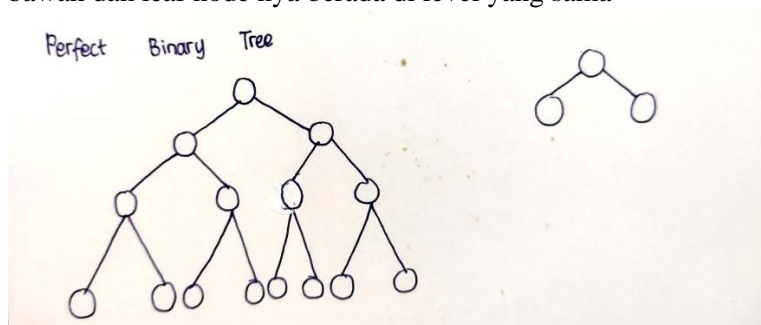
- Full: Binary tree yang memiliki 0 atau 2 anak



- Complete: Binary tree yang semua levelnya terisi dengan node kecuali level paling bawah



- Perfect: Binary tree yang semua node **hanya** boleh punya 2 anak kecuali node paling bawah dan leaf node nya berada di level yang sama



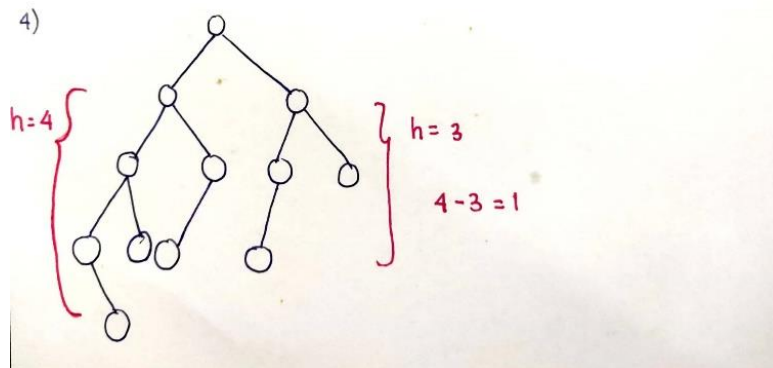
// struktur binary tree dibagi 3

4. What makes a tree balanced?

//jumlah node dibagi 2 balanced dan unbalanced/degenerate

//degenerate: tiap node punya 1 anak kecuali node yg terakhir

Balanced binary tree memiliki syarat $O(\log N)$ dimana N adalah jumlah node atau bisa dilihat dari struktur tree itu sendiri di mana selisih tinggi subtree kiri dan subtree kanan maksimal 1



// jika balance factor > 1 maka unbalance

5. Explain the four properties of a binary tree!

// berlaku untuk perfect binary tree

//properti jumlah node

- Jumlah maksimum node di level k adalah 2^k . k untuk level
- Jumlah maksimum node di binary tree/keseluruhan adalah $2^{h+1}-1$. h untuk height

//properti tinggi binary tree

- Tinggi maksimum dari sebuah binary tree yang terdiri dari n nodes adalah $n-1$. n untuk banyak node
- Tinggi minimum dari sebuah binary tree yang terdiri dari n nodes adalah $\lceil \log_2(n) \rceil$. n untuk banyak node

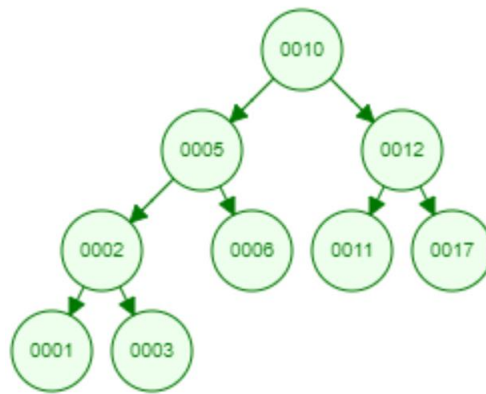
6. Explain the intuition of implementing a binary tree using an array!

Array berukuran statis dan kita menggunakan indexing dengan beberapa rumus

- Root: index 0
- Left child adalah $2p+1$
- Right child adalah $2p+2$
- Parent adalah $(p-1)/2$

// p adalah index parent

Contoh



0	1	2	3	4	5	6	7	8
0010	0005	0012	0002	0006	0011	0017	0001	0003

7. Explain the differences between inorder successor and inorder predecessor!

- Inorder successor: mencari nilai terkecil dari subtrees sebelah kanan.

Inorder predecessor: mencari nilai terbesar dari subtrees sebelah kiri

// digunakan untuk menghapus node yang punya 2 child dan menggantikan node yang dihapus

8. Draw the following binary search tree step by step (14 pictures):

- Insert 80, 30, 60, 50, 75
- Delete 60, 30, 75
- Insert 65, 30, 30
- Delete 80, 65, 35

8) Insert 80, 30, 60, 50, 75

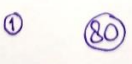
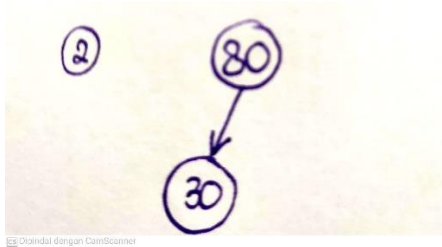
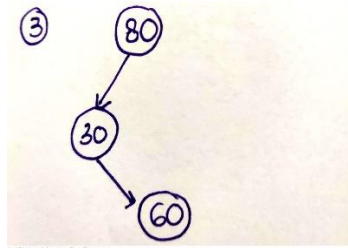


Diagram 1 shows a single node with the value 80.

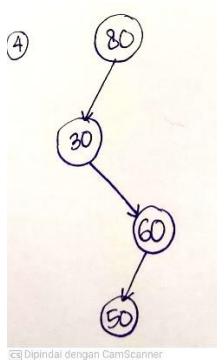
Balance Factor 0



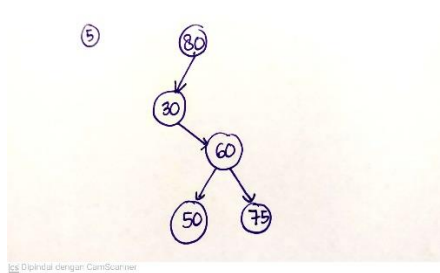
Balance Factor = $1 - 0 = 1$



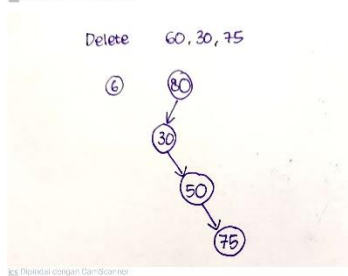
Balance Factor $2 - 0 = 2$



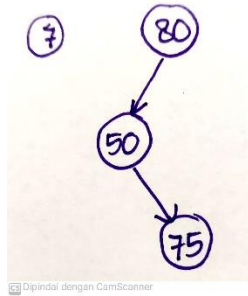
Balance Factor $3 - 0 = 3$



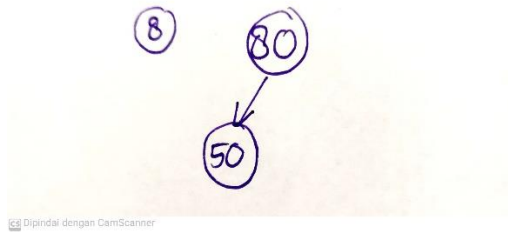
Balance Factor $3 - 0 = 3$



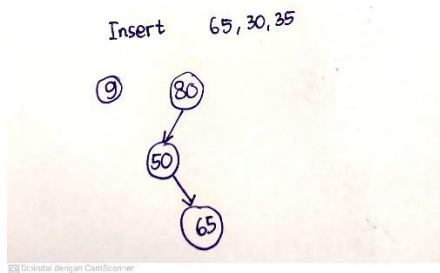
Balance Factor $3 - 0 = 3$



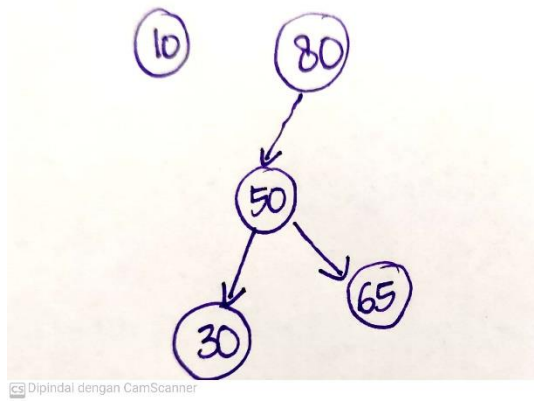
Balance Factor $2-0=2$



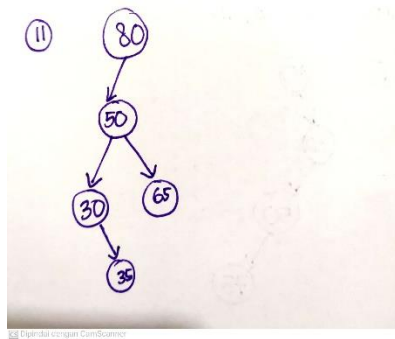
Balance Factor $1-0=1$



Balance Factor $2-0=2$

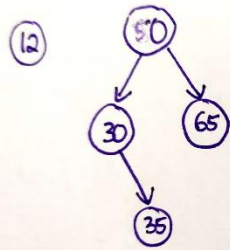


Balance Factor $2-0=2$



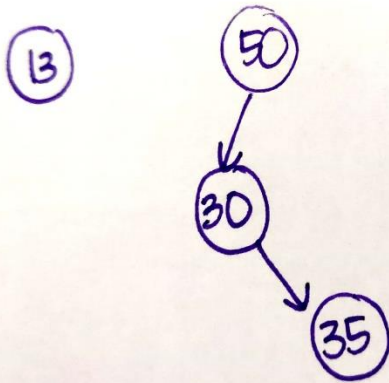
Balance Factor $3-0=3$

Delete 80, 65, 35



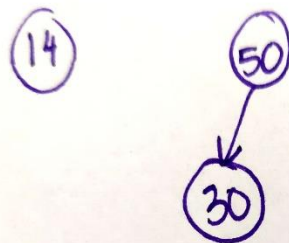
Dipindai dengan CamScanner

Balance Factor = $2 - 1 = 1$



Dipindai dengan CamScanner

Balance Factor = $2 - 0 = 2$



Dipindai dengan CamScanner

Balance Factor = $1 - 0 = 1$