

Struktur Data

Saniati,S.ST.,M.T.

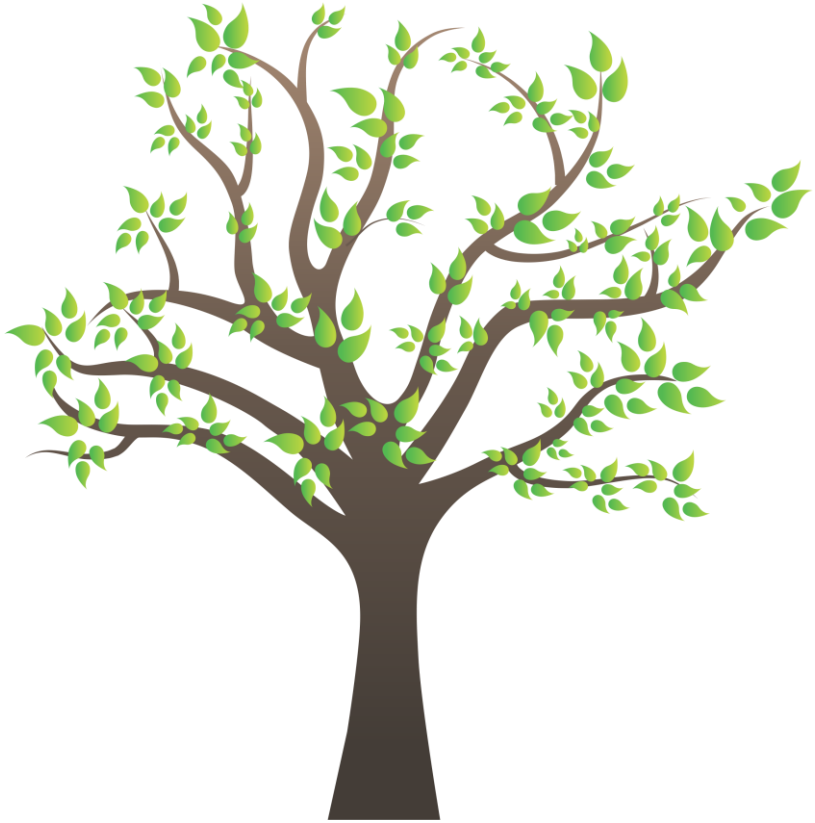
EPISODE 8

Tree

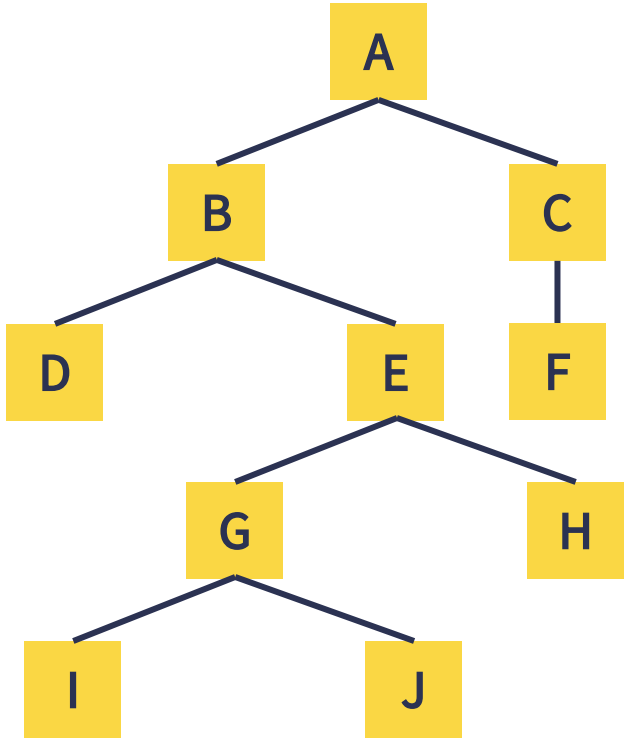
Yang dipelajari ?

- Konsep Tree
- Level & Derajat pada Tree
- Istilah dan Hubungan Komponen Tree
- Definisi Tree
- Ordered & Unordered Tree
- Konsep Binary Tree
- Jenis-jenis Binary Tree
- Tree Tranversal
- Operasi pada Tree

Sedikit Gambaran ?



SEBENARNYA



STRUKTUR DATA

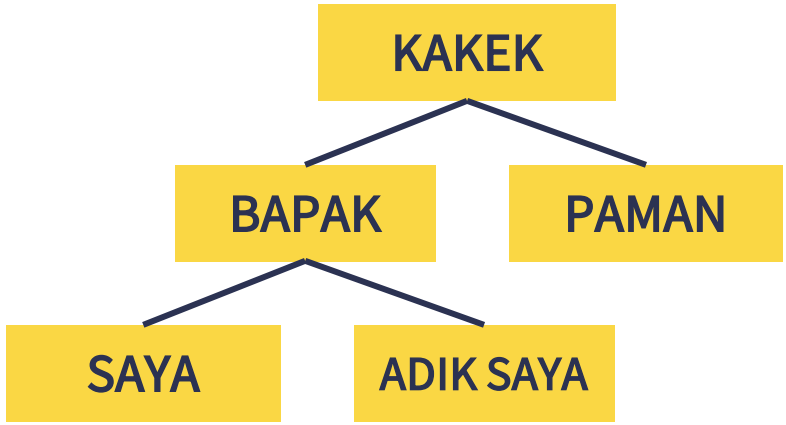
Konsep Tree ?

- Konsep struktur data yang terdiri dari akar dan simpul-simpul yang berada dibawahnya.

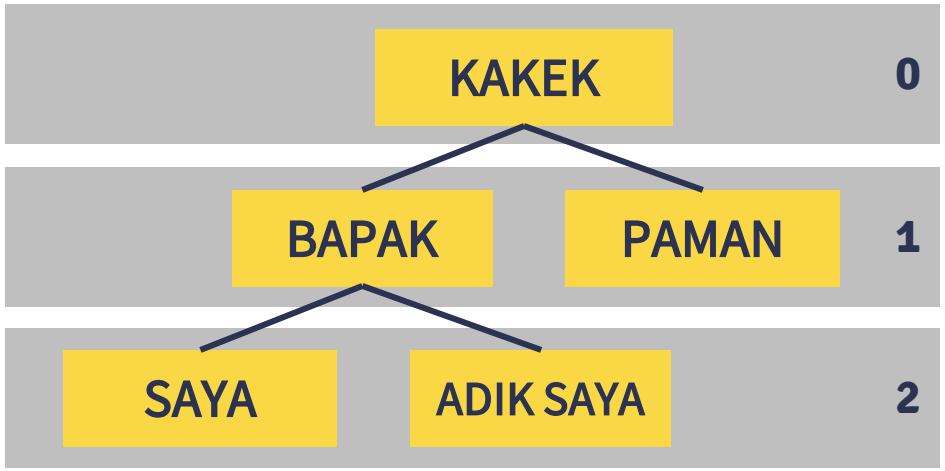
- Struktur data yang menunjukkan hubungan bertingkat (memiliki hirarki).

- Merupakan struktur data yang tidak linear yang digunakan untuk mempresentasikan data yang bersifat hirarki (urutan / tingkatan / kedudukan) antar elemen-elemennya.

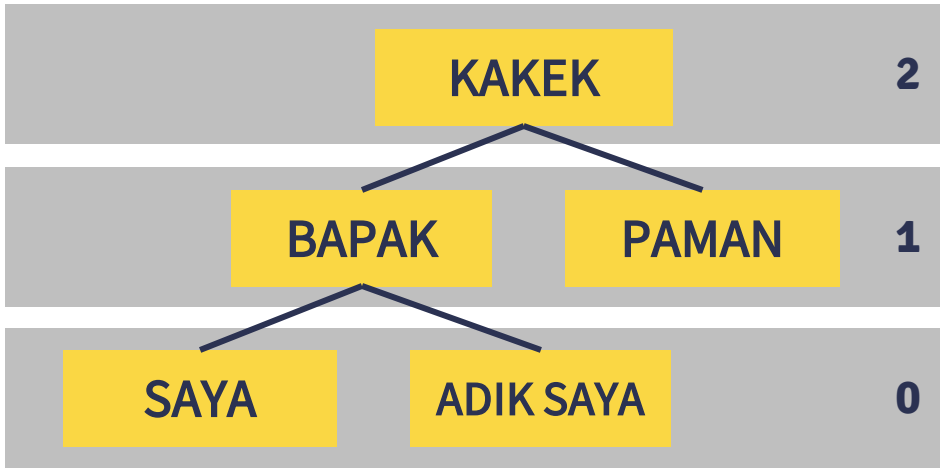
- Contoh : struktur organisasi, silsilah keluarga, struktur folder, dll.



Level & Derajat Tree ?



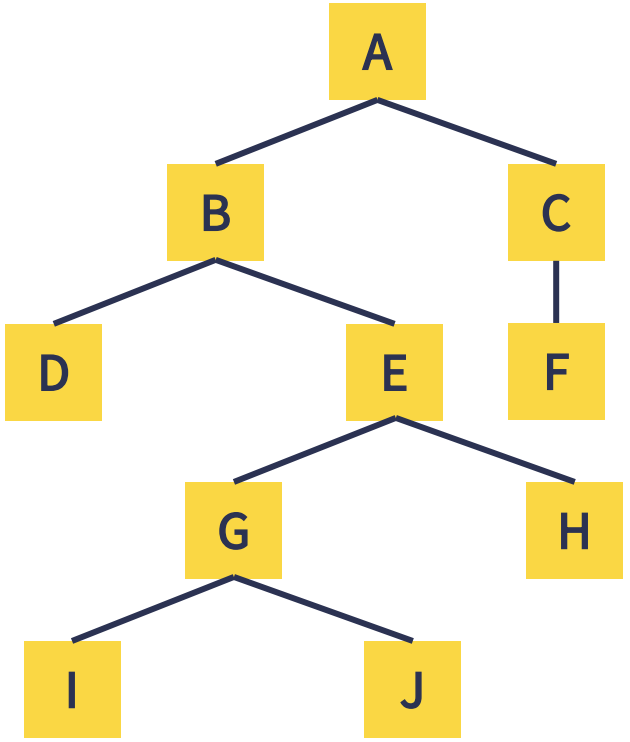
LEVEL



DERAJAT

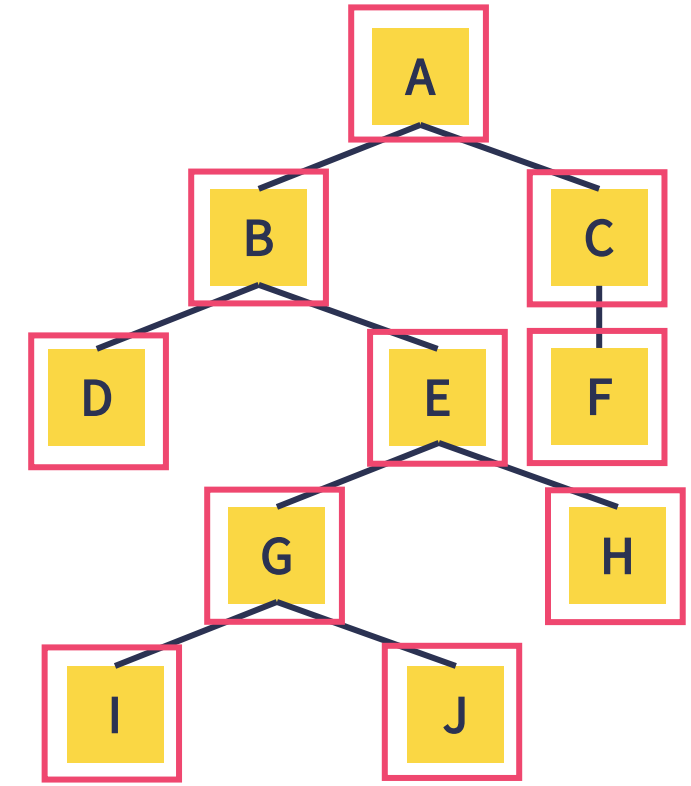
Istilah & Hubungan Komponen Tree ?

- **Node** (Simpul)
- **Predecessor** (Pendahulu)
- **Successor** (Penerus)
- **Ancestor** (Leluhur)
- **Descendant** (Keturunan)
- **Parent** (Orang tua)
- **Child** (Anak)
- **Sibling** (Saudara)
- **Subtree**
- **Size**
- **Height**
- **Root** (Akar)
- **Leaf** (Daun)
- **Degree**
- **Forest** (Hutan)
- **Depth** (Kedalaman)



Istilah & Hubungan Komponen Tree ?

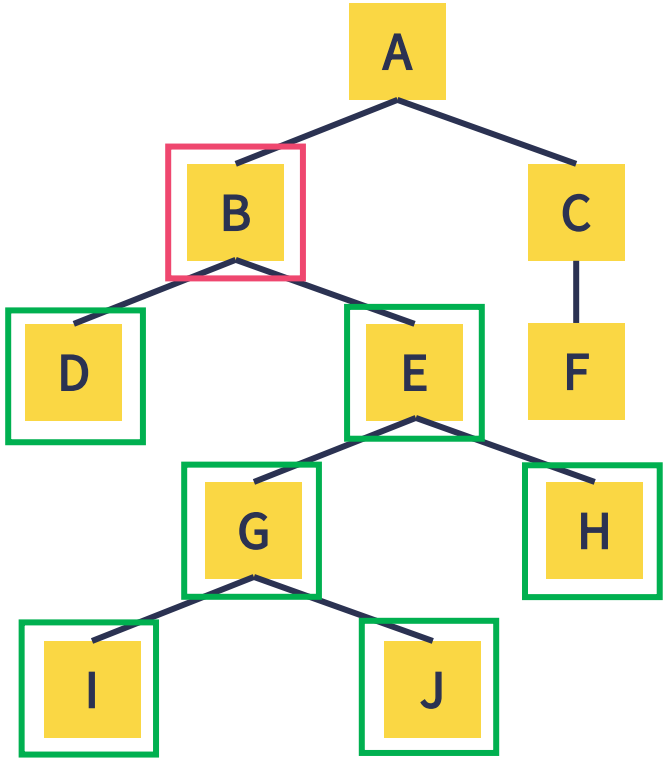
- Node (Simpul**) :** adalah simpul dari masing-masing data dari suatu tree.



Node = A,B,C,D,E,F,G,H,I,J

Istilah & Hubungan Komponen Tree ?

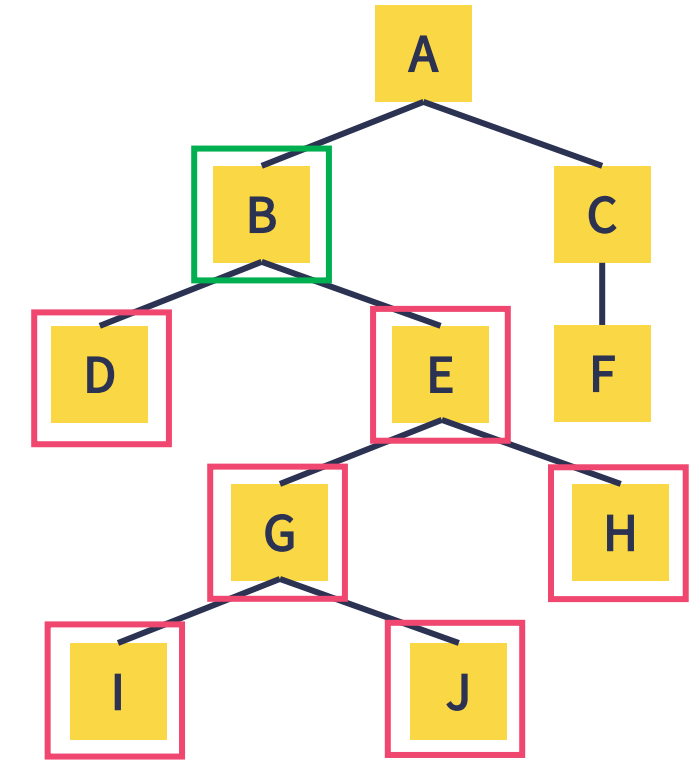
- Predecessor (Pendahulu) **:** adalah node yang berada diatas node tertentu



Predecessor (D,E,G,H,I,J) = **B**

Istilah & Hubungan Komponen Tree ?

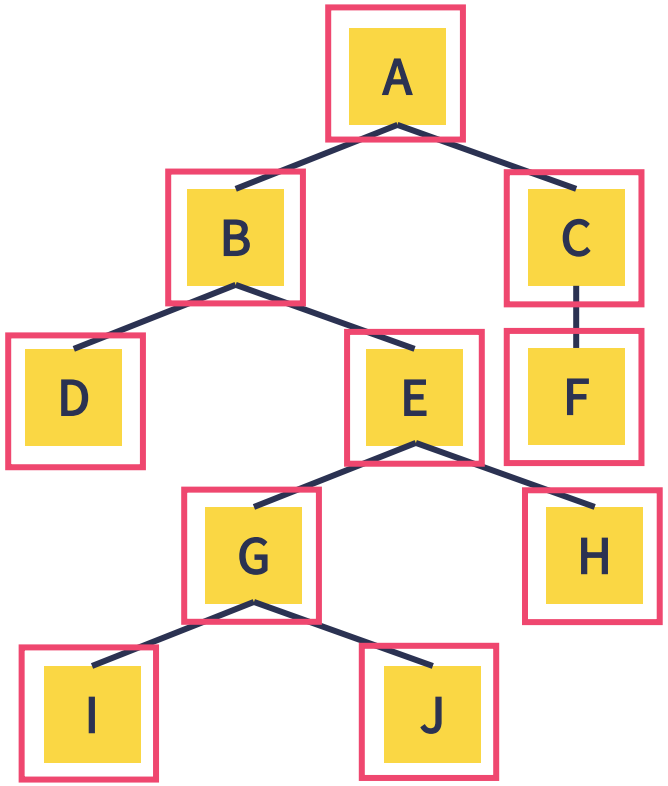
- Successor (Penerus**) & Subtree :** adalah node yang berada dibawah node tertentu



Successor (B**) = D,E,G,H,I,J**

Istilah & Hubungan Komponen Tree ?

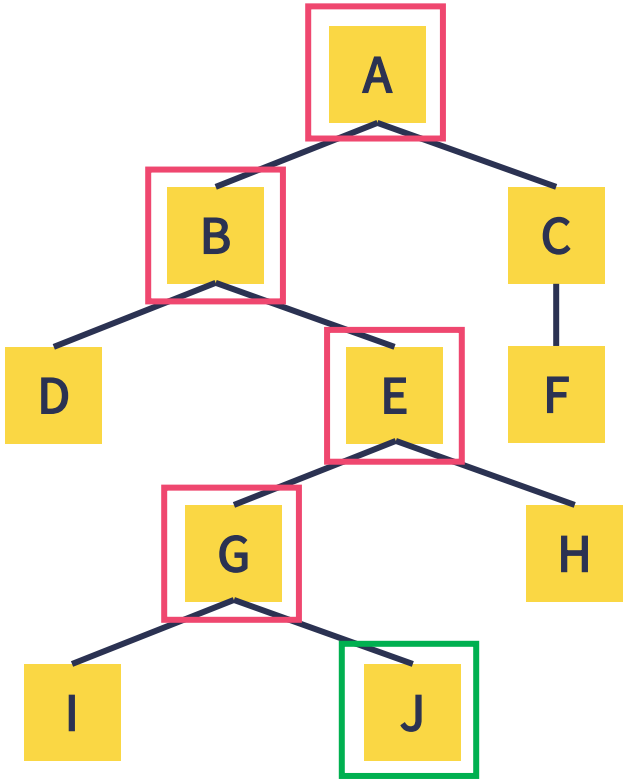
- **Size :** adalah banyaknya node disebuah tree



Size = 10

Istilah & Hubungan Komponen Tree ?

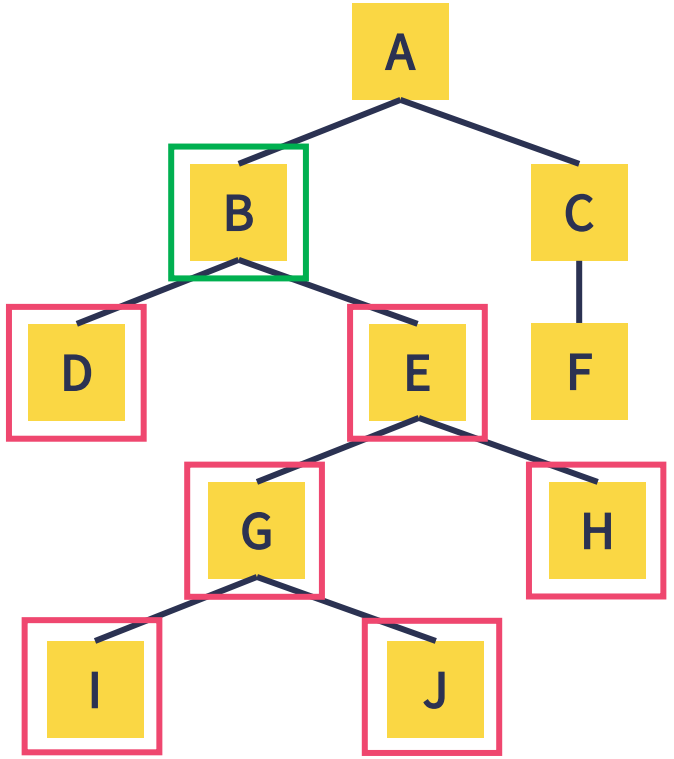
- Ancestor (Leluhur**) :** Seluruh node yang terletak sebelum node tertentu dan terletak pada jalur yang sama.



Ancestor (J) = G,E,B,A

Istilah & Hubungan Komponen Tree ?

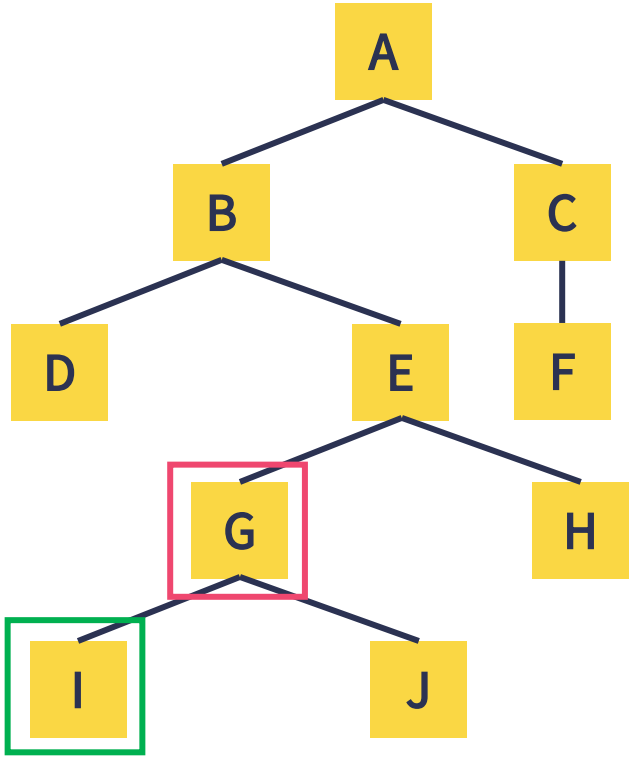
- Descendant (Keturunan**) :** Seluruh node yang terletak setelah node tertentu dan terletak pada jalur yang sama.



Descendant (B**) = D,E,G,H,I,J**

Istilah & Hubungan Komponen Tree ?

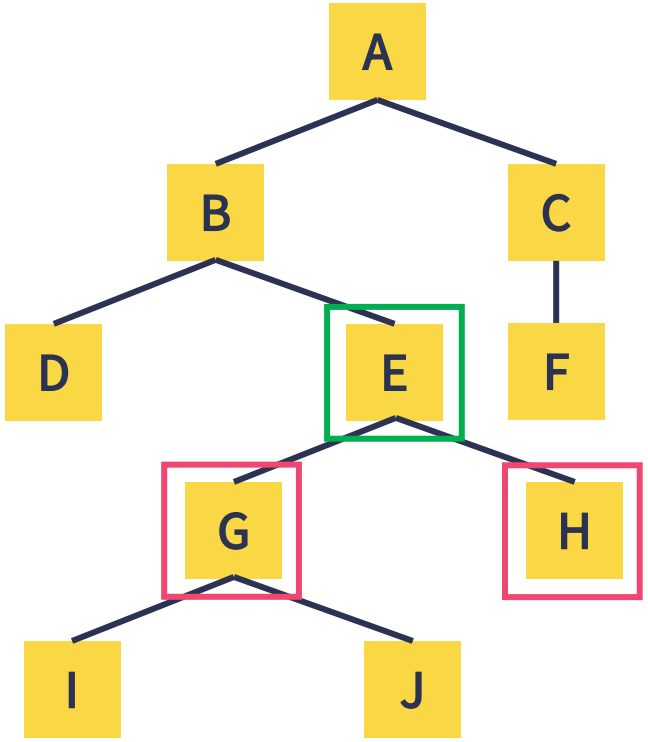
- Parent (Orang tua**) :** Predecessor (pendahulu) satu level diatas suatu node.



Parent (I) = G

Istilah & Hubungan Komponen Tree ?

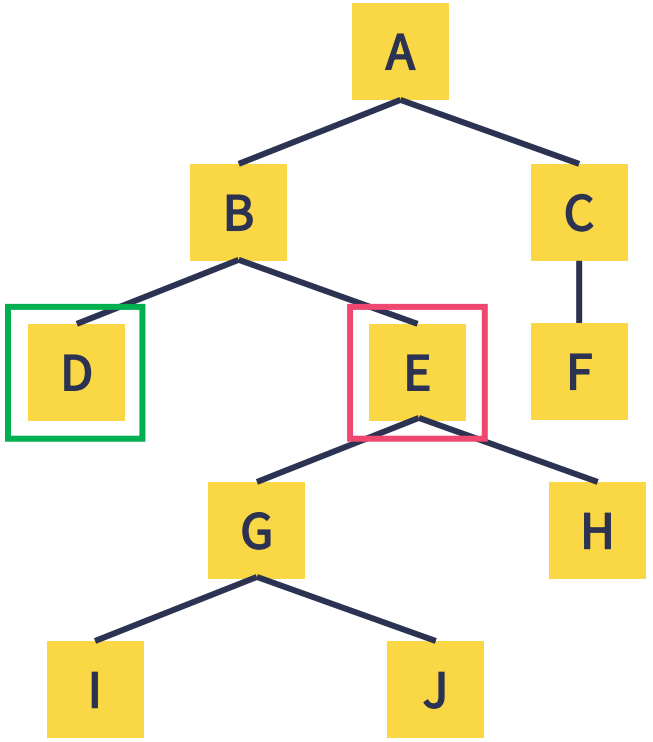
- Child (Anak**) :** adalah successor (penerus) satu level dibawah suatu node.



Child (E**) = G,H**

Istilah & Hubungan Komponen Tree ?

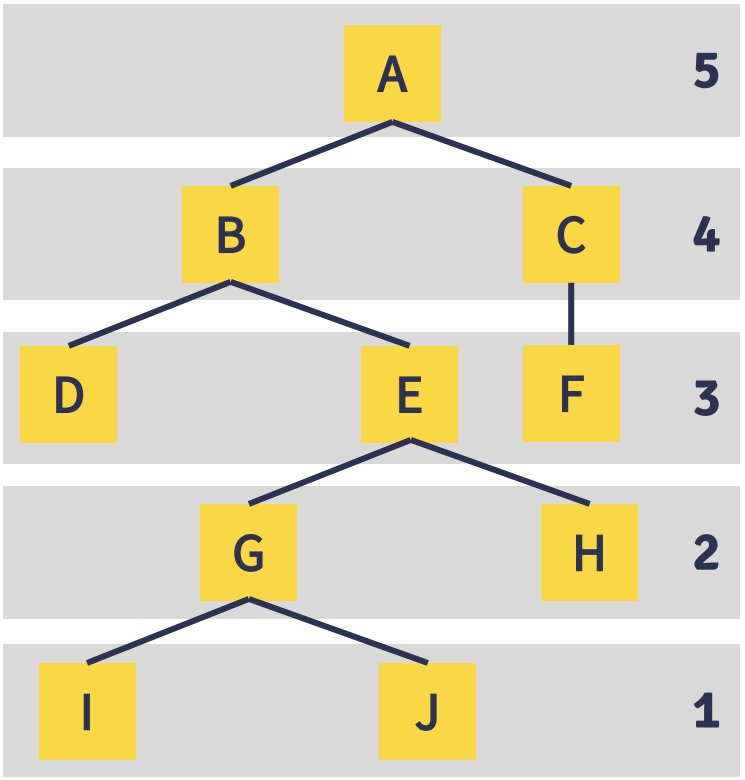
- Sibling (Saudara**) :** adalah node-node yang memiliki parent yang sama.



Sibling (D**) = E**

Istilah & Hubungan Komponen Tree ?

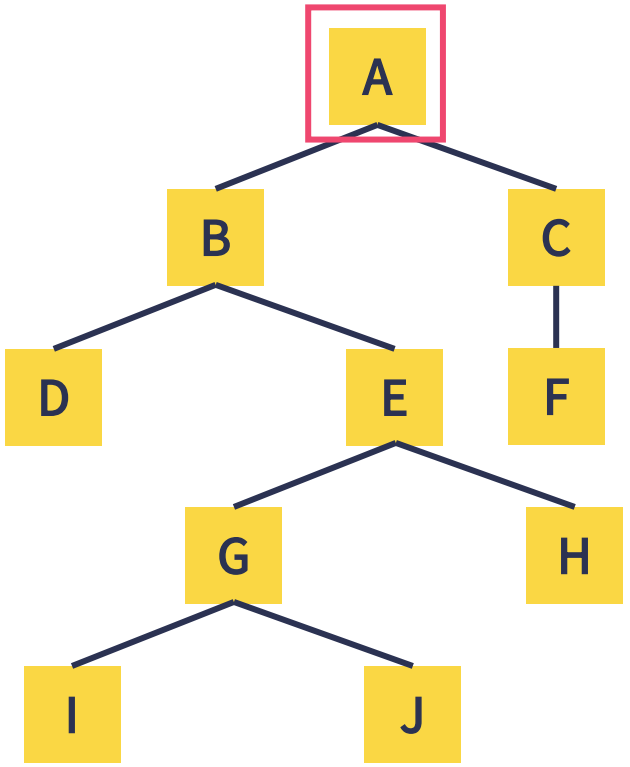
- **Height :** Banyaknya tingkatan dalam suatu tree.



Height = 5

Istilah & Hubungan Komponen Tree ?

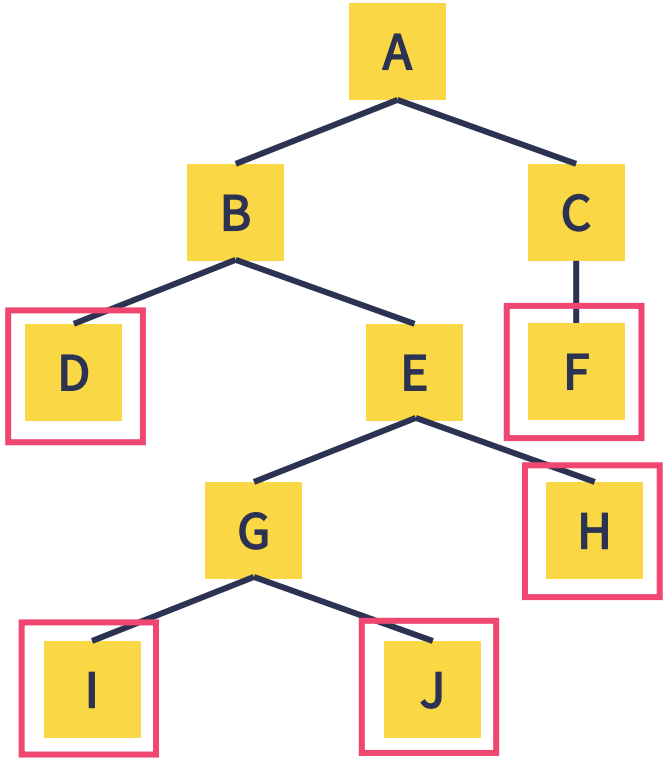
- Root (Akar**) :** adalah node khusus yang tidak memiliki predecessor (pendahulu).



Root = A

Istilah & Hubungan Komponen Tree ?

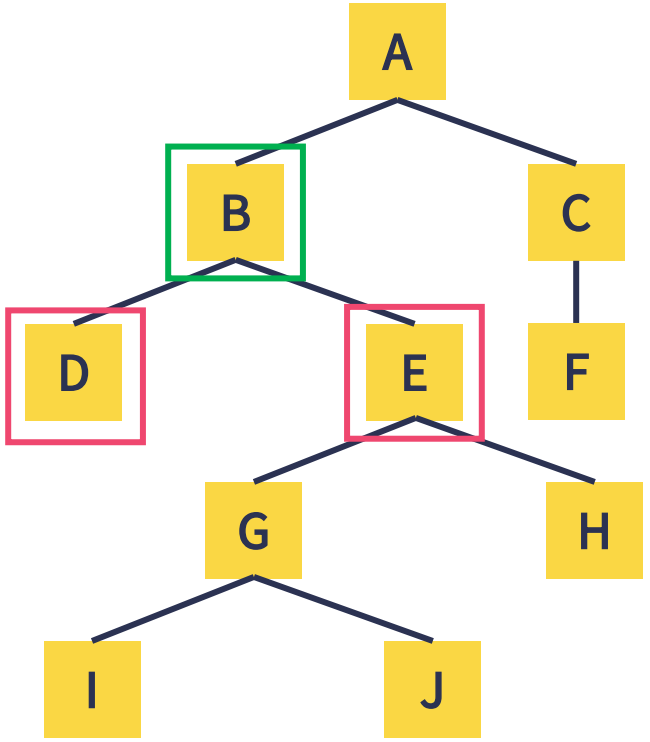
- Leaf (Daun**) :** adalah node-node dalam tree yang tidak memiliki successor (penerus).



Leaft = D,F,H,I,J

Istilah & Hubungan Komponen Tree ?

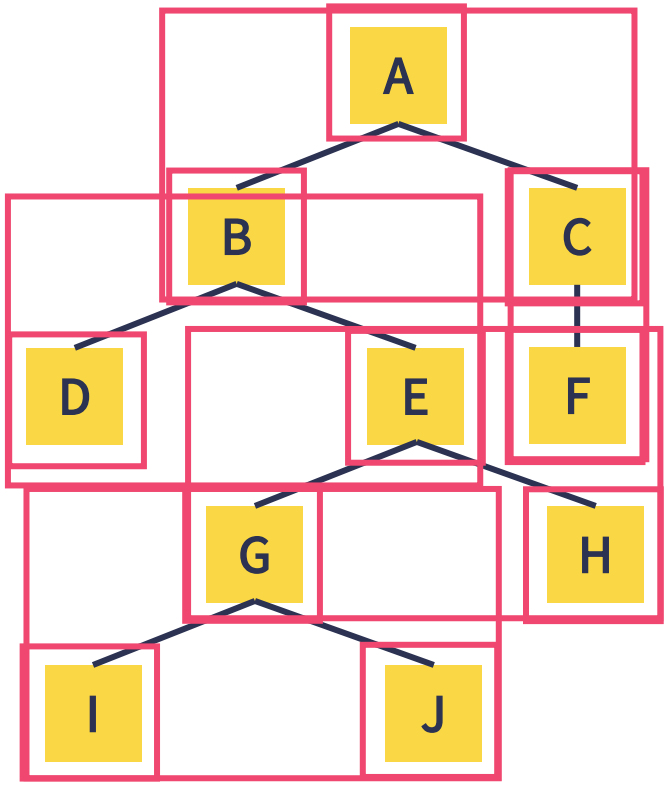
- Degree : adalah banyaknya child (anak) dalam suatu node.



Degree (B) = 2

Istilah & Hubungan Komponen Tree ?

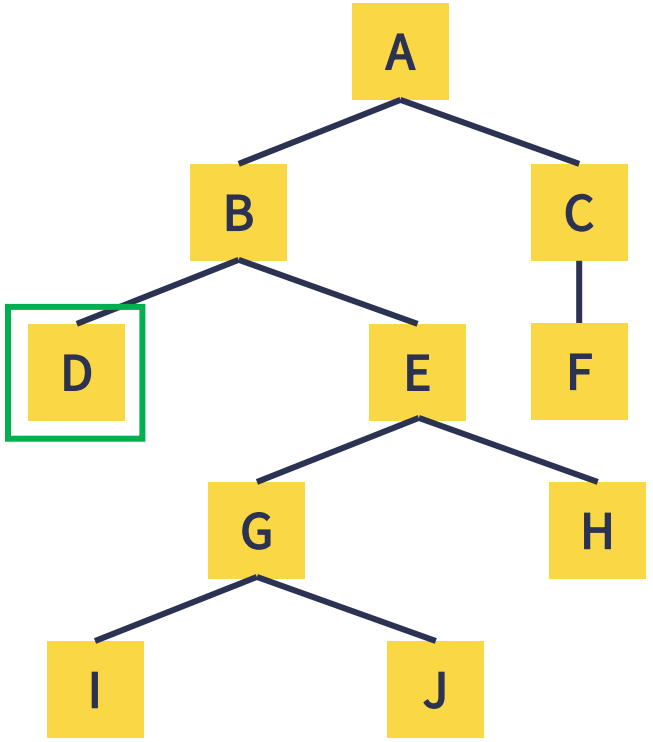
- **Forest** (Hutan) **:** adalah kumpulan dari tree.



Forest = 15

Istilah & Hubungan Komponen Tree ?

- Depth (Kedalaman**) :** adalah hasil tingkat node maksimum dikurang satu (level dari node **x**).

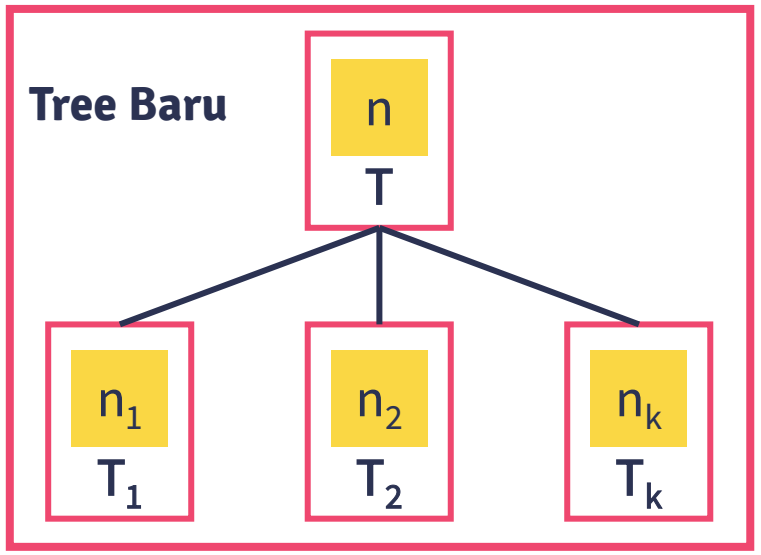


Depth (D**) = 2**

Definisi Tree ?

- Sebuah tree didefinisikan sebagai struktur yang dibentuk secara rekursif oleh aturan berikut.
  - Sebuah node adalah sebuah tree. Node satu-satunya pada tree ini berfungsi sebagai root maupun leaf.
  - Dari k buah tree $T_1 \sim T_k$, dan masing-masing memiliki root $n_1 \sim n_k$.
  - Jika node n adalah parent dari $n_1 \sim n_k$, akan diperoleh sebuah tree baru T yang memiliki root n. Dalam kondisi ini, tree $T_1 \sim T_k$ menjadi subtree dari tree T.

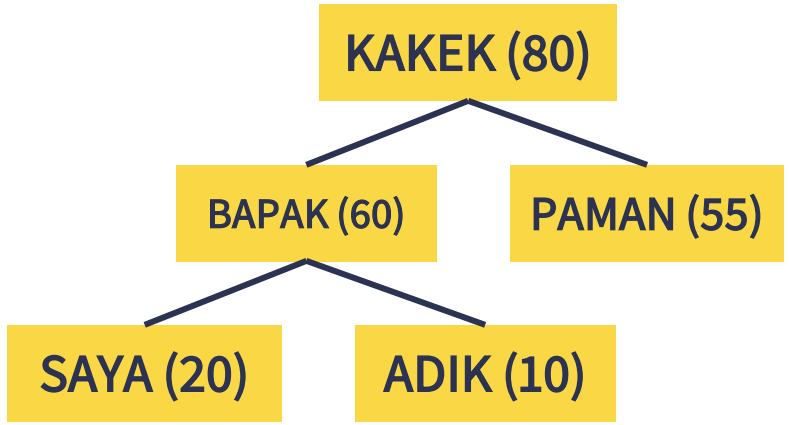
T : Tree
N : Node
k : banyak



Ordered & Unordered Tree ?

- Ordered Tree

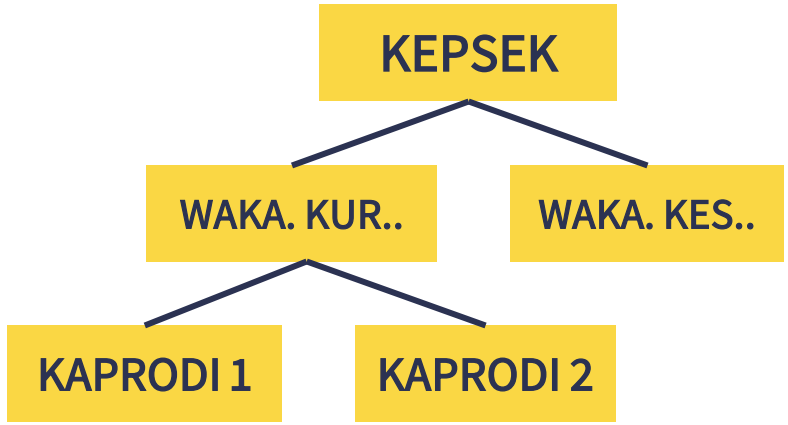
- Antar sibling (saudara) terdapat urutan “usia”.
- Node yang paling kiri berusia paling tua (sulung), sedangkan node yang paling kanan berusia paling muda (bungsu).
- Posisi node diatur atas urutan tertentu.
- Contoh : silsilah keluarga.



Ordered Tree

- Unordered Tree

- Antar sibling (saudara) tidak terdapat urutan tertentu.
- Contoh : struktur organisasi sekolah SMK.

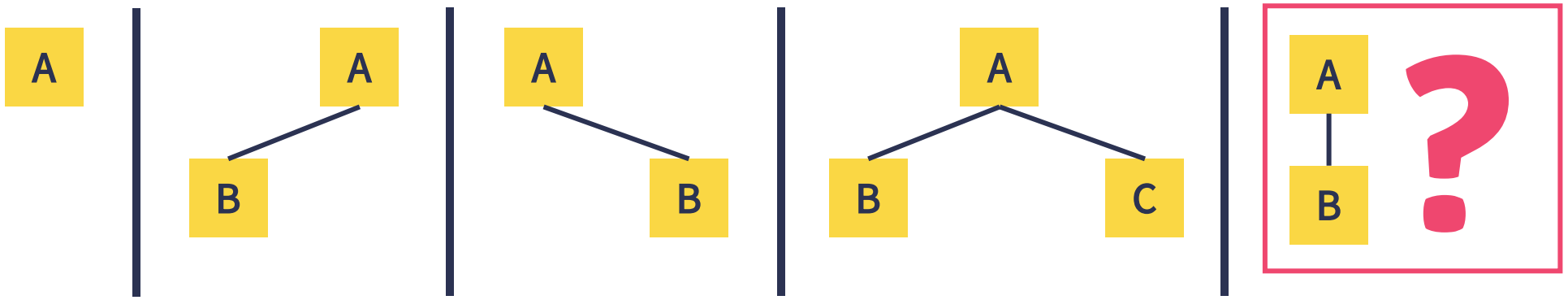


Unordered Tree

Konsep Binary Tree ?

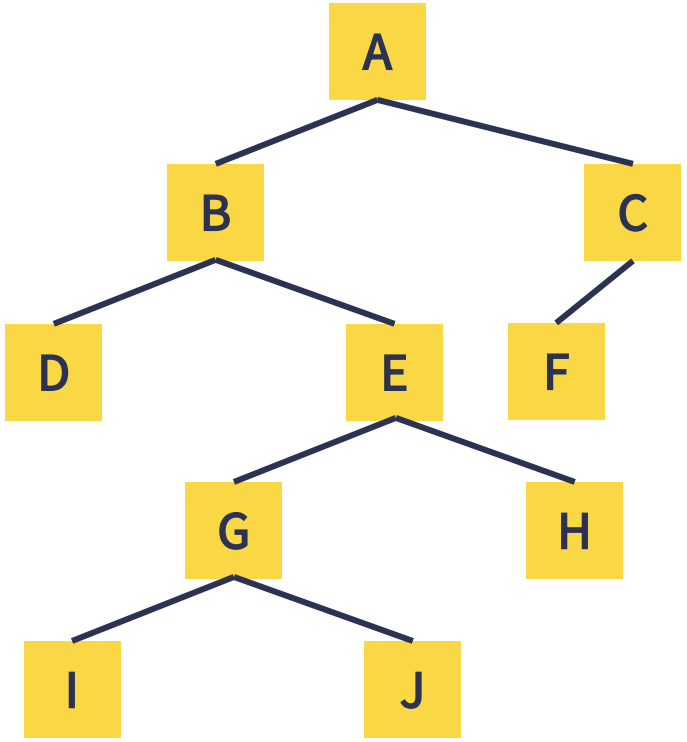
- Binary adalah tree dengan syarat bahwa tiap node hanya boleh memiliki maksimal dua subtree dan kedua subtree harus terpisah.
  - Binary tree boleh tidak memiliki child (anak) ataupun subtree.
  - Boleh hanya memiliki subtree sebalah kiri (left subtree).
  - Boleh hanya memiliki subtree sebalah kanan (right subtree).
  - Boleh hanya memiliki subtree sebalah kiri (left subtree) dan kanan (right subtree).

Hati-hati menggambarkan Binary Tree



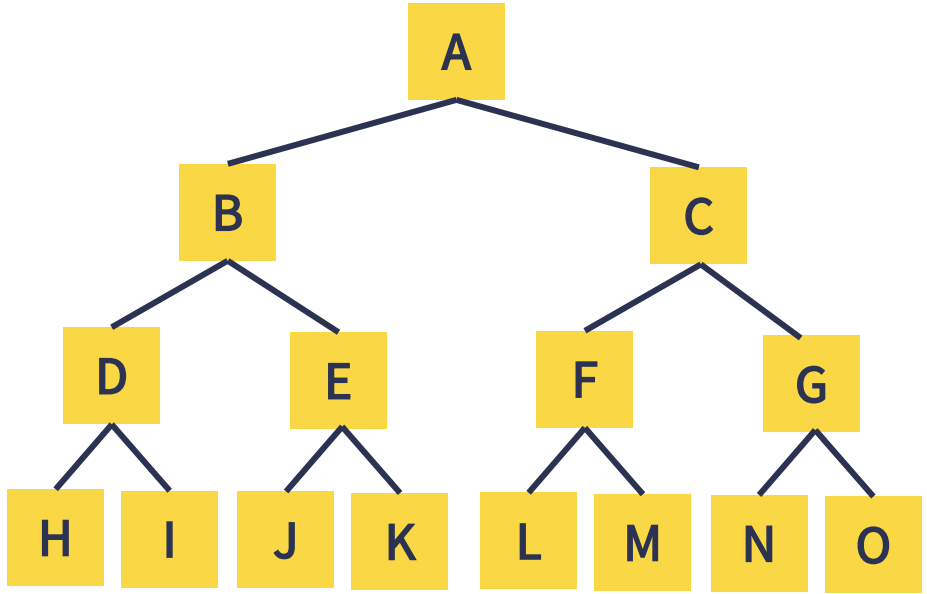
Jenis-jenis Binary Tree ?

- Full Binary Tree
- Complete Binary Tree
- Skewed Binary Tree



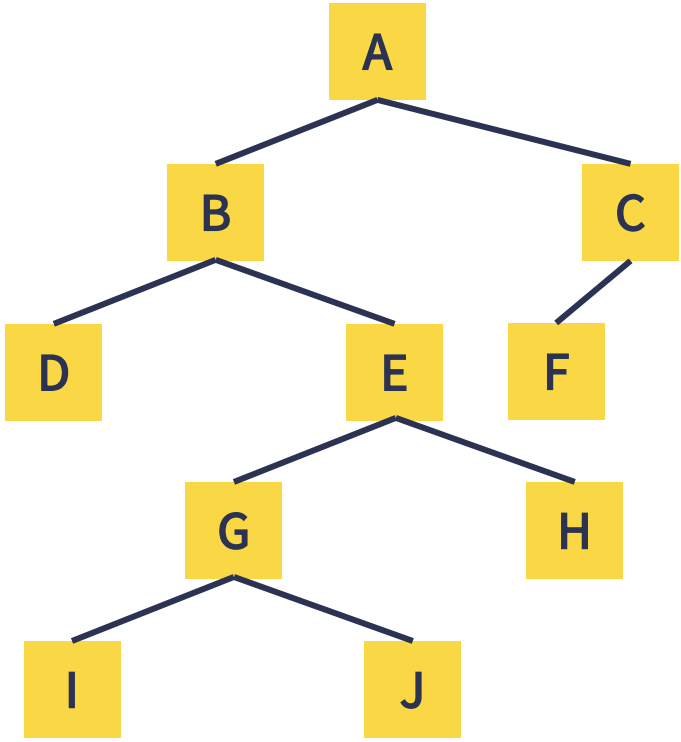
Jenis-jenis Binary Tree ?

- Full Binary Tree : adalah binary tree yang tiap node nya (kecuali leaf) memiliki dua child dan tiap subtree mempunyai panjang patch yang sama.



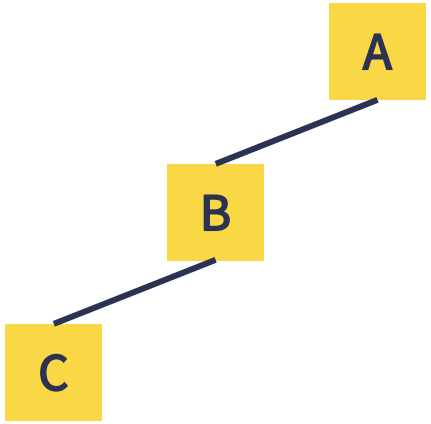
Jenis-jenis Binary Tree ?

- Complete Binary Tree : adalah binary tree yang mirip dengan full binary tree, namun setiap subtree boleh memiliki panjang patch yang berbeda.



Jenis-jenis Binary Tree ?

- Skewed Binary Tree : adalah binary tree yang semua node nya (kecuali left) hanya memiliki satu child.



Definisi Tree Tranversal ?

- Adalah teknik menyusuri tiap node dalam sebuah tree secara sistematis, sehingga semua node dapat dan hanya satu kali saja dikunjungi.

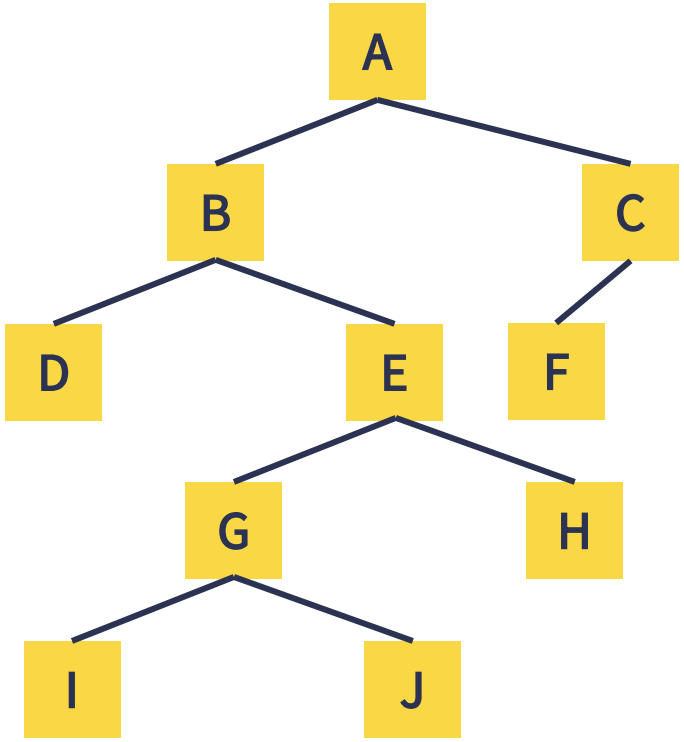
- Ada tiga cara tranversal :
  - preOrder.
  - inOrder.
  - postOrder.

- Untuk tree atau node yang kosong, tranversal tidak perlu dilakukan.

Cara Tranversal ?

preOrder

1. Kunjungi root nya.
2. Telusuri subtree kiri.
3. Telusuri subtree kanan.

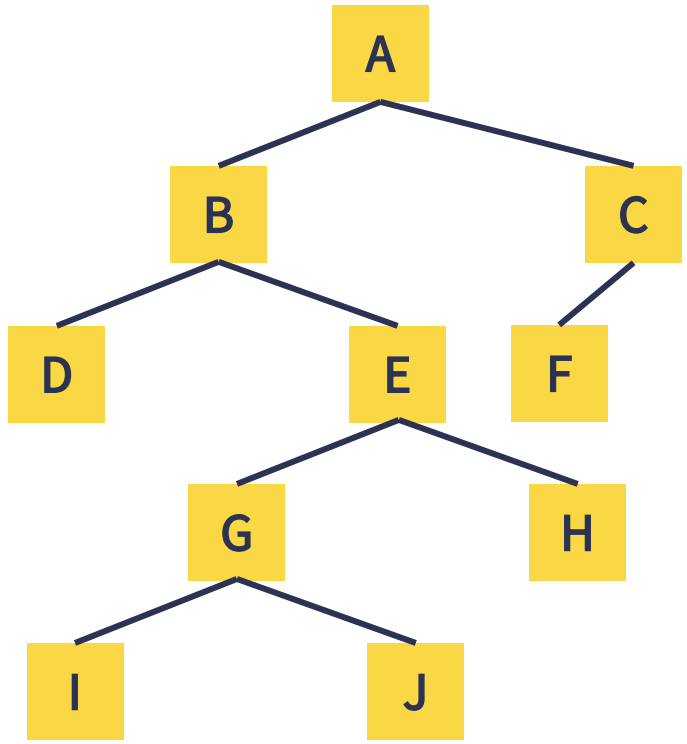


preOrder = A,B,D,E,G,I,J,H,C,F

Cara Tranversal ?

inOrder

1. Telusuri subtree kiri.
2. Kunjungi root nya.
3. Telusuri subtree kanan.

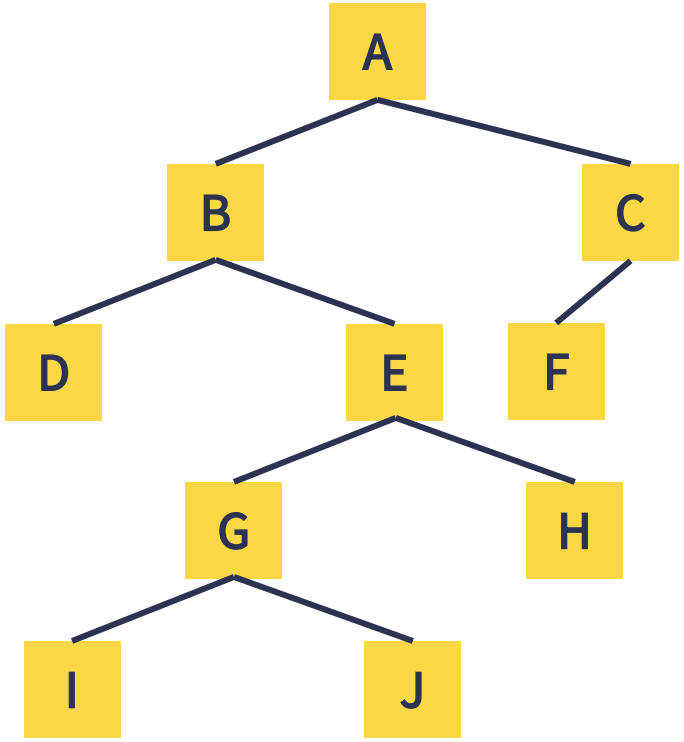


inOrder = D,B,I,G,J,E,H,A,F,C

Cara Tranversal ?

postOrder

1. Telusuri subtree kiri.
2. Telusuri subtree kanan.
3. Kunjungi root nya.



postOrder = D,I,J,G,H,E,B,F,C,A

Operasi pada Tree ?

- **Create :** digunakan untuk membentuk binary tree baru yang masih kosong.
- **Clear :** digunakan untuk mengosongkan binary tree yang sudah ada atau menghapus semua node pada binary tree.
- **Empty :** digunakan untuk memeriksa apakah binary tree masih kosong atau tidak.
- **Insert :** digunakan untuk memasukkan sebuah node kedalam tree.
- **Find :** digunakan untuk mencari root, parent, left child, atau right child dari suatu node dengan syarat tree tidak boleh kosong.
- **Update :** digunakan untuk mengubah isi dari node yang ditunjuk oleh pointer current dengan syarat tree tidak boleh kosong.
- **Retrieve :** digunakan untuk mengetahui isi dari node yang ditunjuk pointer current dengan syarat tree tidak boleh kosong.
- **Delete Sub :** digunakan untuk menghapus sebuah subtree (node beserta seluruh descendant-nya) yang ditunjuk pointer current dengan syarat tree tidak boleh kosong.
- **Charateristic :** digunakan untuk mengetahui karakteristik dari suatu tree. Yakni size, height, serta average lenght-nya.
- **Tranverse :** digunakan untuk mengunjungi seluruh node-node pada tree dengan cara tranversal.

Video Selanjutnya

Implementasi Tree C++

Thank you

#KEEPLEARNING
#KEEPSPIRITS