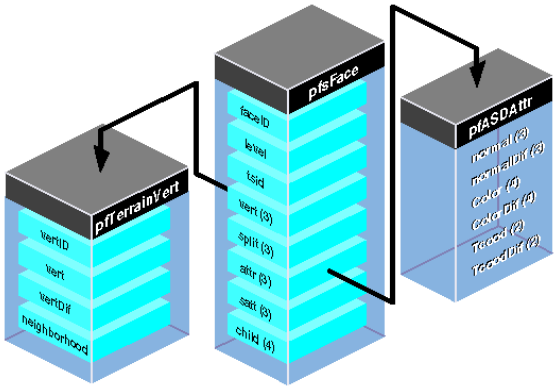
Struktur Data – Pertemuan 14

Pohon dan Pohon Biner



P r a j a n t o W a h y u A d i

prajanto@dsn.dinus.ac.id

+6285 641 73 00 22

Rencana Kegiatan Perkuliahan Semester

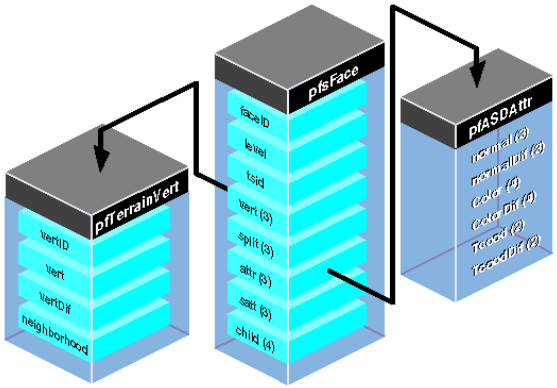
#	Pokok Bahasan
~~1~~	~~Pengenalan Struktur Data~~
~~2~~	~~ADT Stack & Queue~~
~~3~~	~~List Linear~~
~~4~~	~~List Linear~~
~~5~~	~~List Linear~~
~~6~~	~~Representasi Fisik List Linear~~
~~7~~	~~Variasi List Linear~~
8	**Ujian Tengah Semester**

#	Pokok Bahasan
~~9~~	~~Variasi List Linear~~
~~10~~	~~Variasi List Linear~~
~~11~~	~~Stack dengan Representasi List~~
~~12~~	~~Queue dengan Representasi List~~
~~13~~	~~List Rekursif~~
14	**Pohon dan Pohon Biner**
15	Studi Kasus Multi List
16	**Ujian Akhir Semester**

Konten

1. Tree (Pohon)
2. Binary Tree (Pohon Biner)
3. Binary Tree Traversal

1. Tree (Pohon)

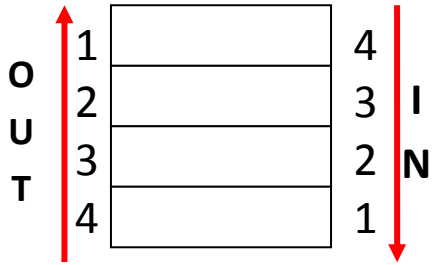


P r a j a n t o W a h y u A d i

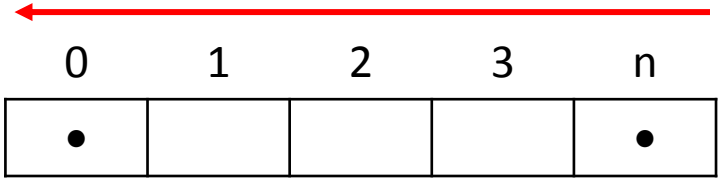
prajanto@dsn.dinus.ac.id

+6285 641 73 00 22

Struktur Data Linier

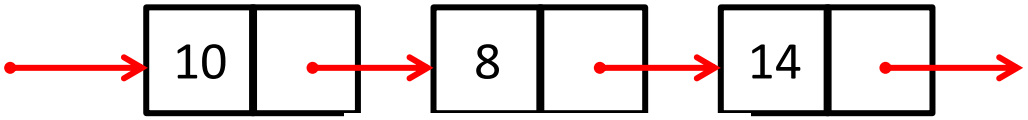


STACK



QUEUE

ARRAY



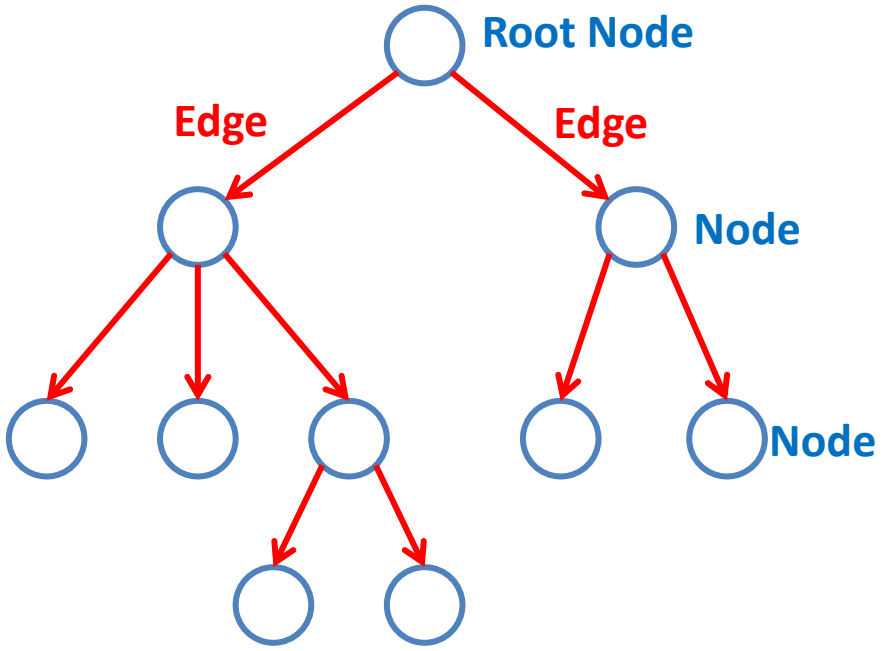
LINKED LIST

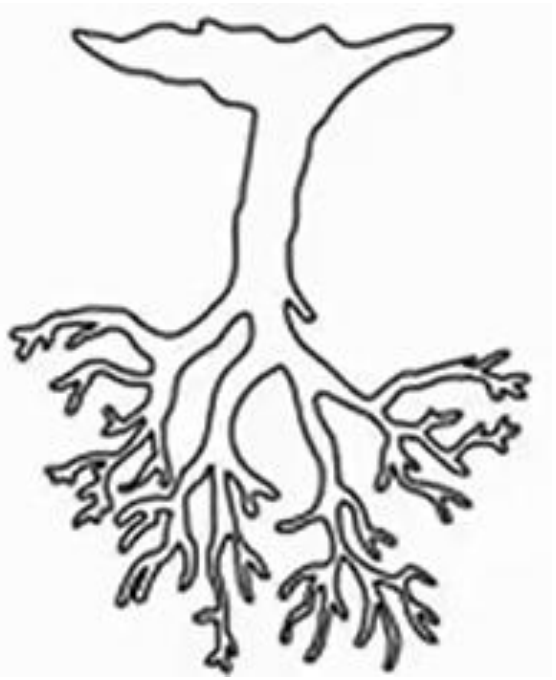
Tree

- Pohon adalah struktur data hirarki
- Tree adalah struktur data yang terdiri dari entitas yang disebut node yang terkait melaui sebuah edge
- Node paling atas disebut dengan root

Tree

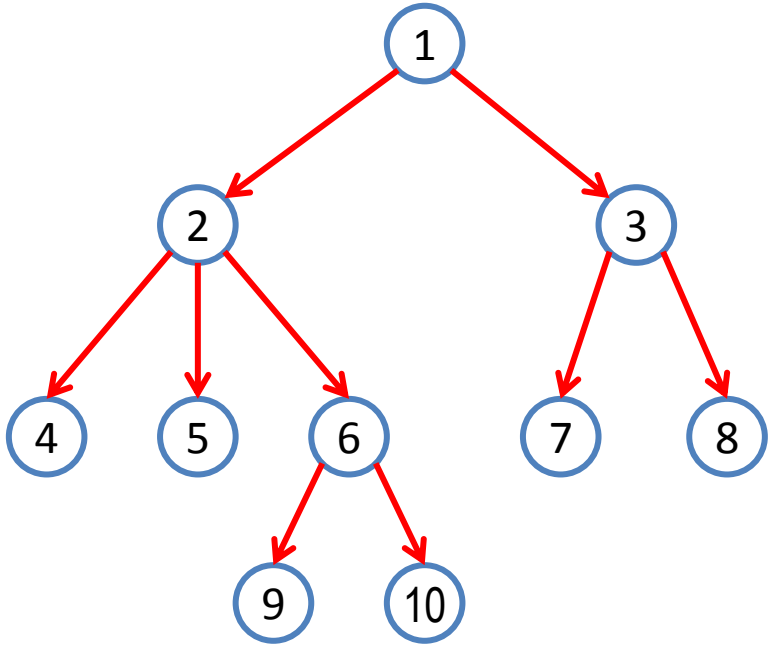
- Pohon adalah struktur data hirarki
- Tree adalah struktur data yang terdiri dari entitas yang disebut node yang terkait melaui sebuah edge
- Node paling atas disebut dengan root





Tree

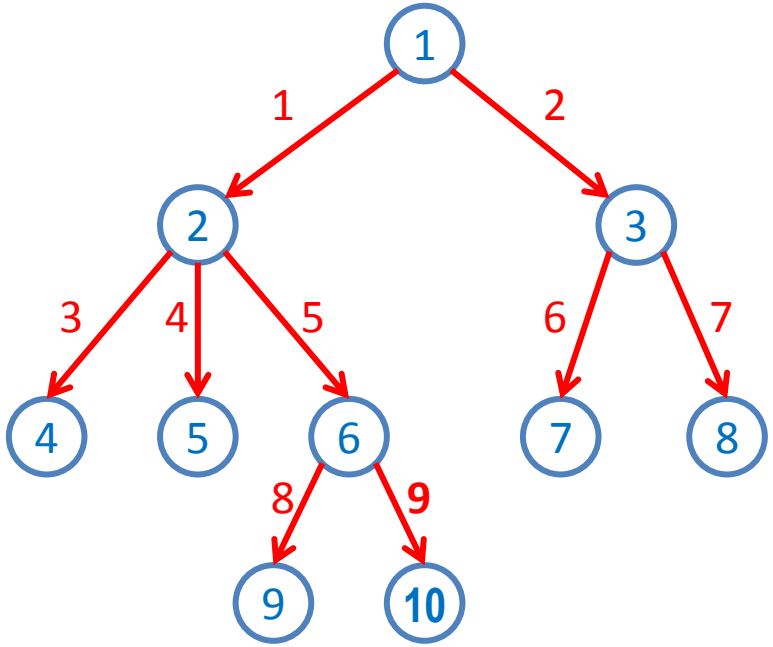
- Node dengan pd posisi yg lebih tinggi disebut dg parent dan yang lebih rendah disebut children
- Node dengan posisi yang yang sama disebut sibling
- Node dengan posisi paling rendah disebut leaf



- 1 adalah root
- 1 adalah parent dari 2 dan 3
- 2 dan 3 adalah children dari 1
- 2 adalah parent dari 4,5, dan 6
- 4, 5, dan 6 adalah sibling
- 7 dan 8 adalah children dari 3
- 7 dan 8 adalah sibling
- 9 dan 10 adalah leaf

Tree

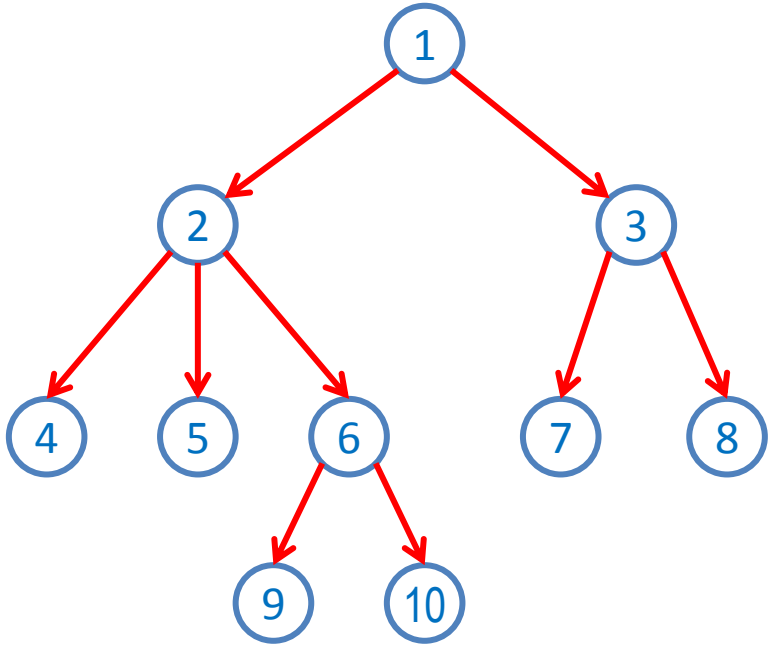
- Tree mempunyai :
  - n node
  - n-1 edge



- Jumlah node adalah 10
- Jumlah edge adalah 9

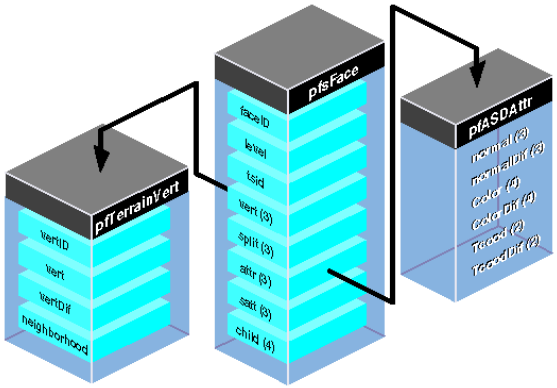
Tree

- Depth of Node : jumlah edge dari root ke node
- Height of Node: jumlah edge terpanjang dari node ke leaf
- Height of Tree = height of root node



- Depth of node 1 adalah 0
- Height of node 1 adalah 3
- Depth of node 6 adalah 2
- Height of node 6 adalah 1
- Depth of node 9 adalah 3
- Height of node 9 adalah 0
- Height of tree adalah 3

2. Binary Tree (Pohon Biner)



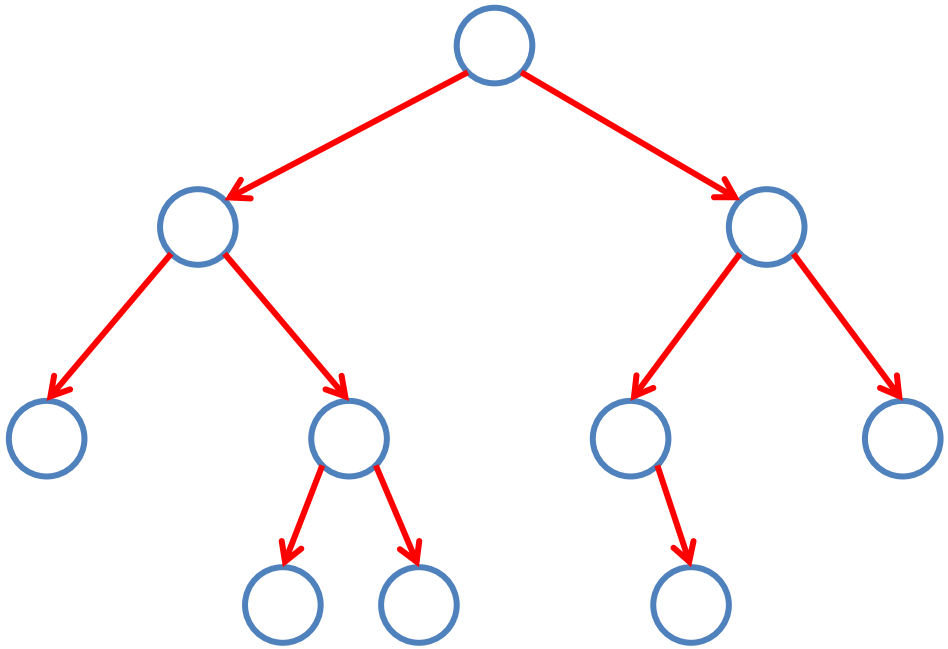
P r a j a n t o W a h y u A d i

prajanto@dsn.dinus.ac.id

+6285 641 73 00 22

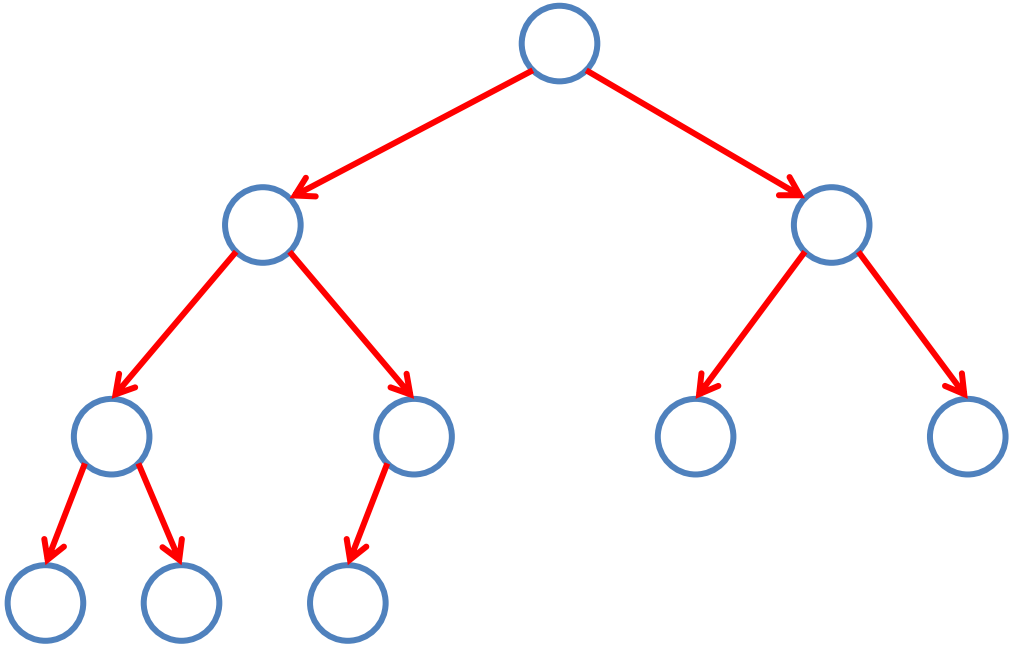
Binary Tree

- Binary Tree adalah tree dimana setiap node mempunyai paling banyak 2 children
- Children dari setiap node disebut left-child dan right-child



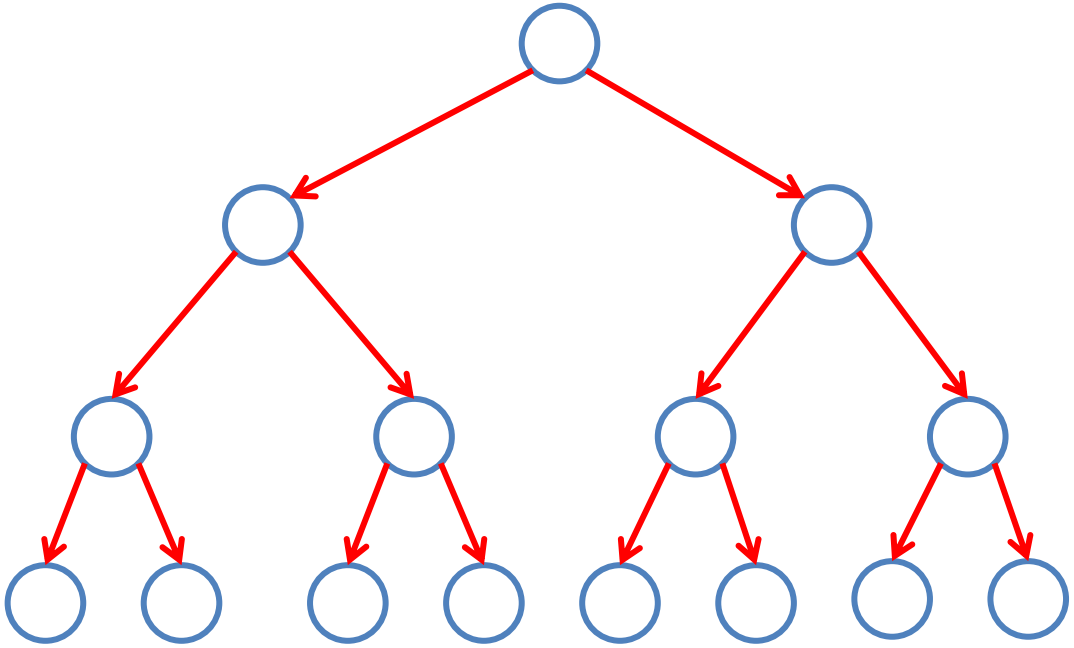
Binary Tree

- Complete Binary Tree
  semua level selain level terakhir pada tree terisi lengkap dan semua node kiri terisi lebih dahulu



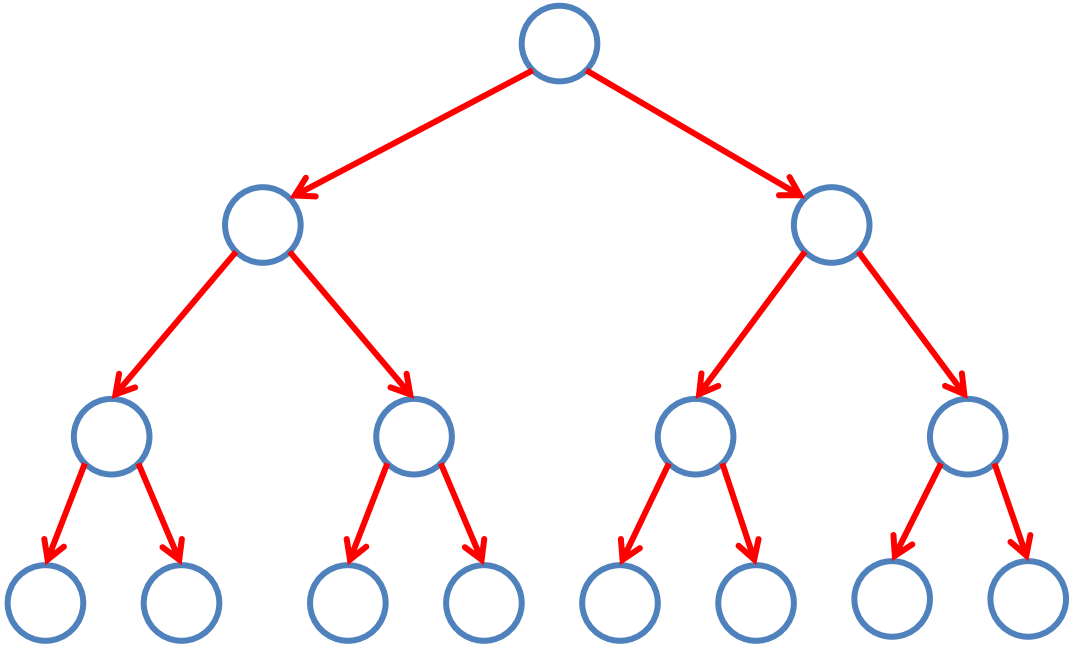
Binary Tree

- Perfect Binary Tree
  semua level pada tree terisi lengkap

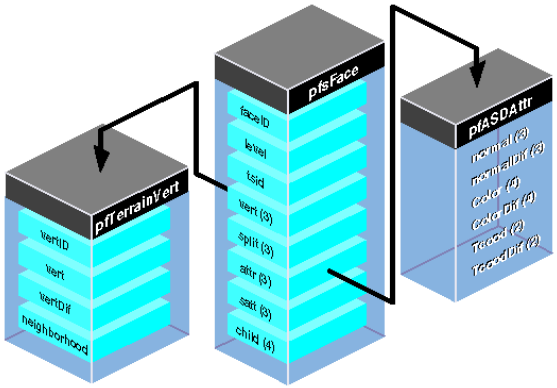


Binary Tree

- Jumlah node maksimal pada perfect binary tree dengan height n adalah $2^{n+1}-1$
- Height dari perfect binary tree dengan n node adalah $\log_2(n+1)-1$



3. Binary Tree Traversal



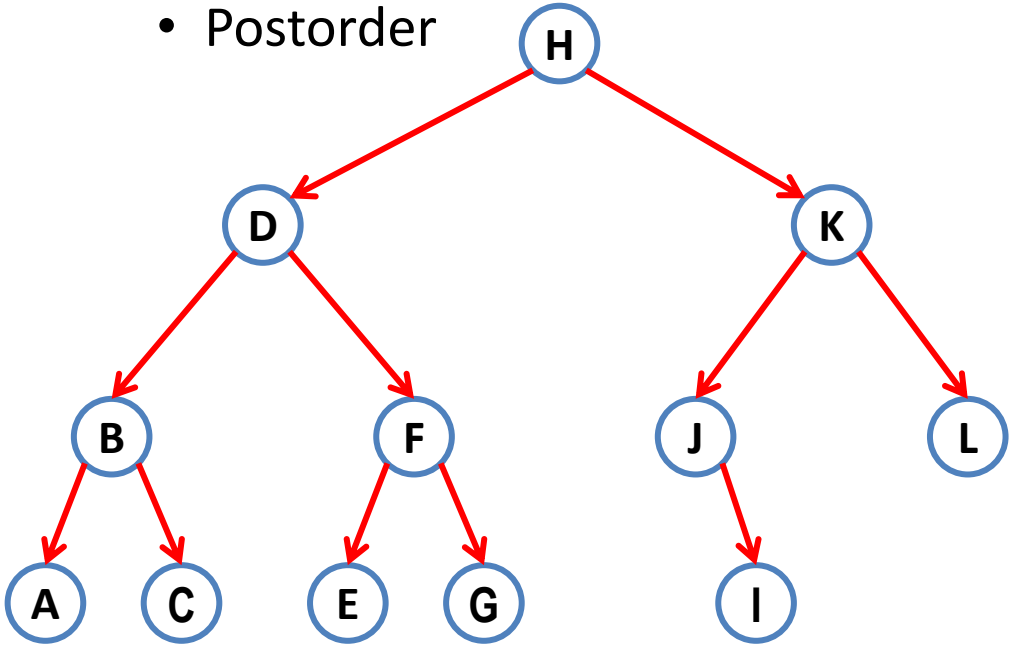
P r a j a n t o W a h y u A d i

prajanto@dsn.dinus.ac.id

+6285 641 73 00 22

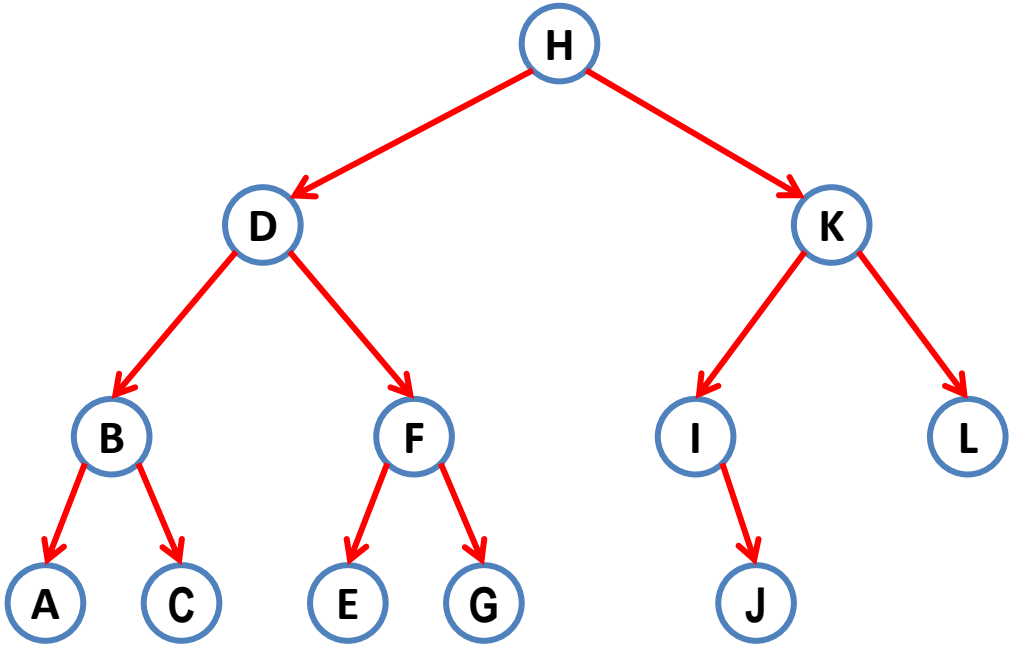
Binary Tree Traversal

- Binary Tree Traversal
  - Breadth First : Level order
  - Depth First :
    - Preorder
    - Inorder
    - Postorder



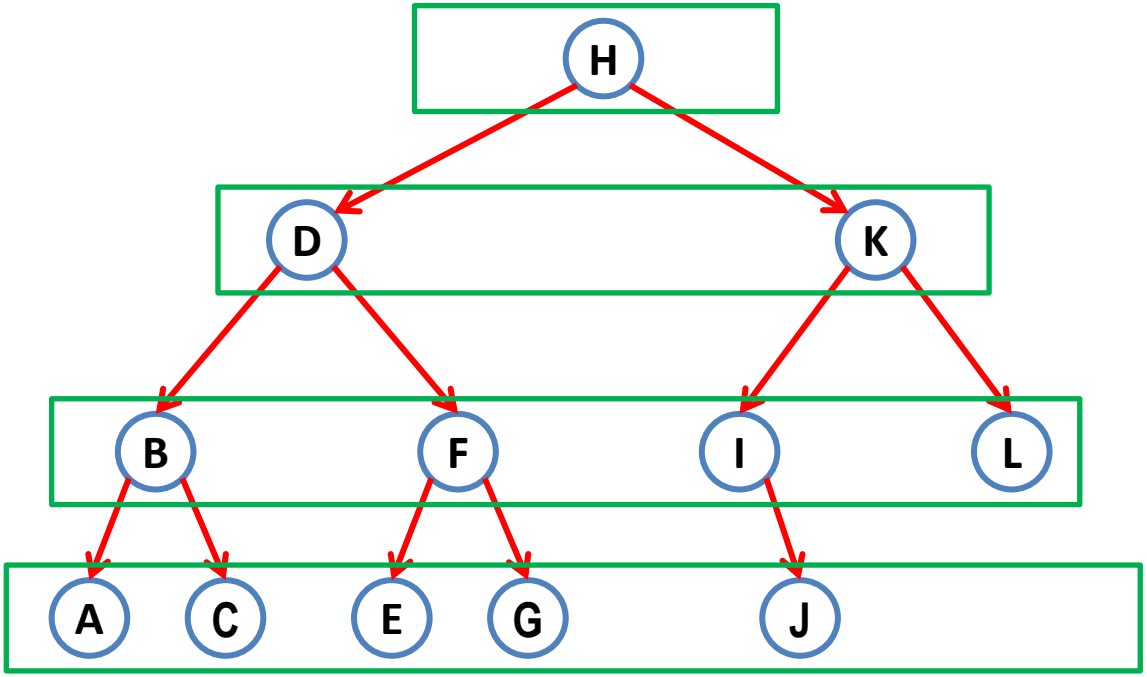
Binary Tree Traversal

- Binary Tree Traversal
  - Level Order Traversal
    mengunjungi setiap node dari level teratas kemudian bergerak ke node sebelah kiri kemudian node sebelah kanan pada level dibawahnya.



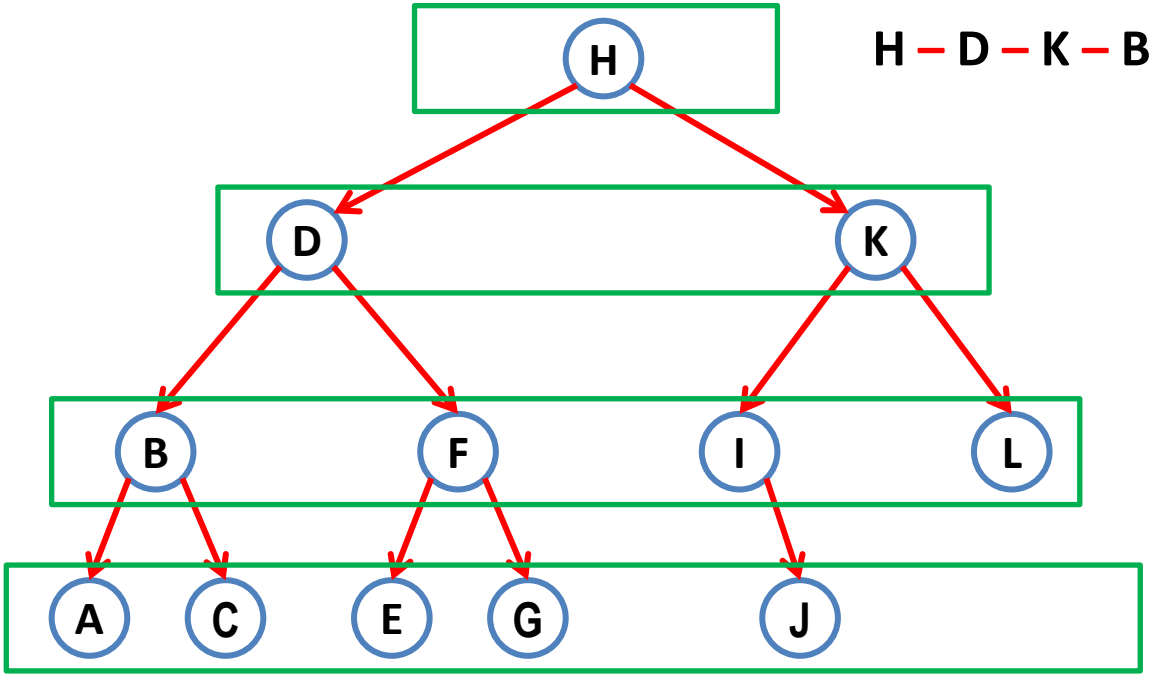
Binary Tree Traversal

- Binary Tree Traversal
  - Level Order Traversal
    mengunjungi setiap node dari level teratas kemudian bergerak ke node sebelah kiri kemudian node sebelah kanan pada level dibawahnya.



Binary Tree Traversal

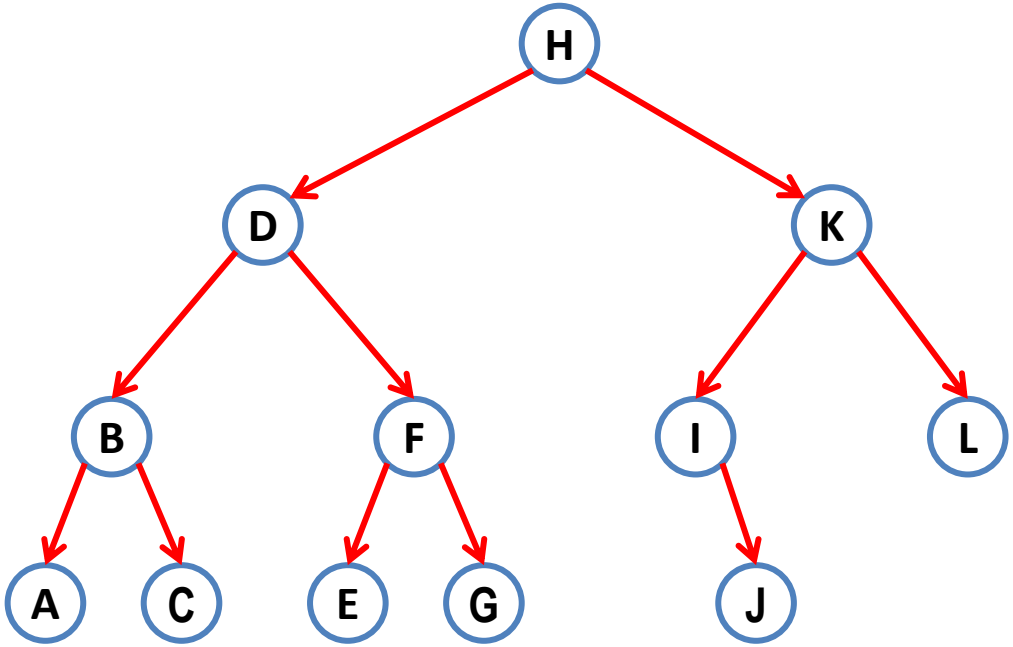
- Binary Tree Traversal
  - Level Order Traversal
    mengunjungi setiap node dari level teratas kemudian bergerak ke node sebelah kiri kemudian node sebelah kanan pada level dibawahnya.



H – D – K – B – F – I – L – A – C – E – G – J

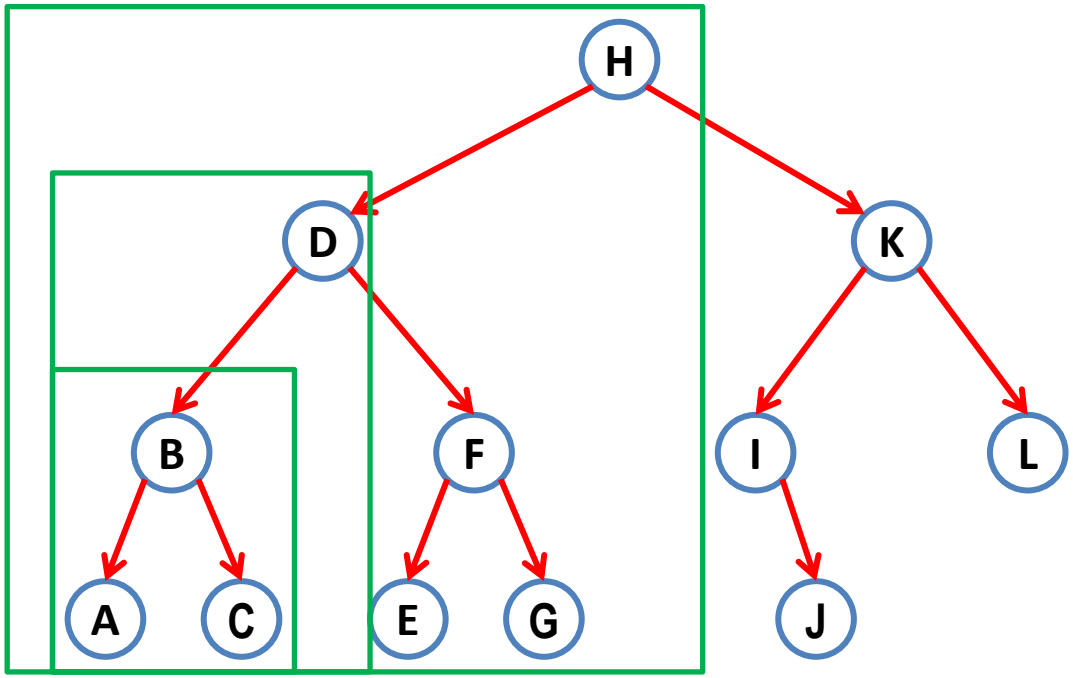
Binary Tree Traversal

- Binary Tree Traversal
  - Preorder traversal
    mengunjungi node terbawah hingga mencapai setiap children node dengan urutan:
    Data/parent → Left children → Right children



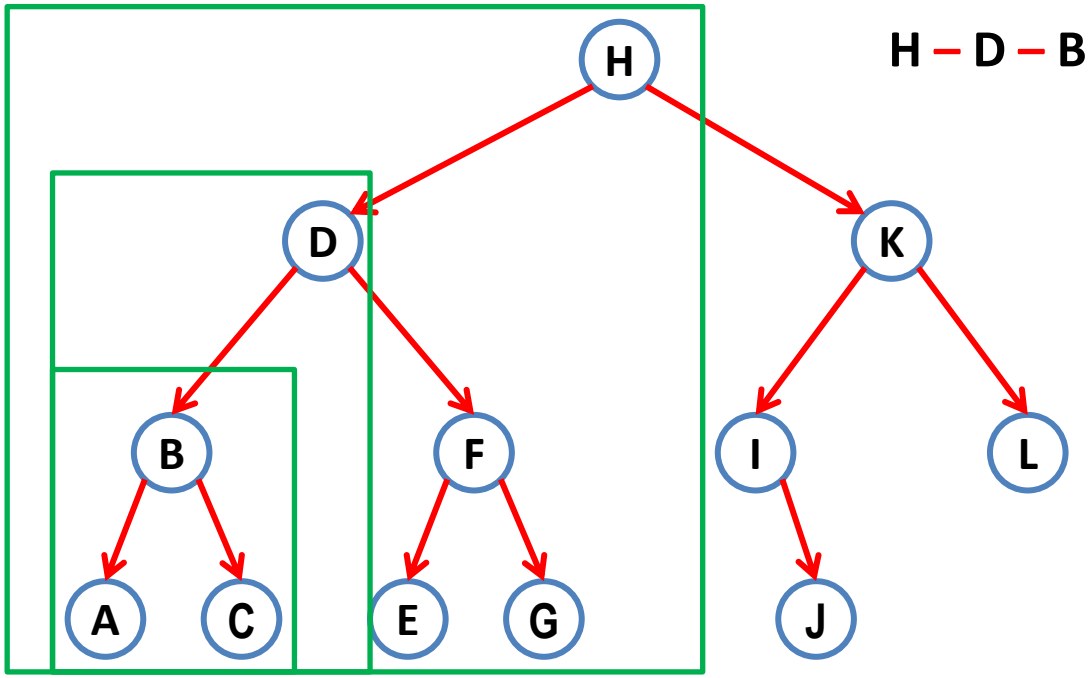
Binary Tree Traversal

- Binary Tree Traversal
  - Preorder traversal
    mengunjungi node terbawah hingga mencapai setiap children node dengan urutan:
    Data/parent → Left children → Right children



Binary Tree Traversal

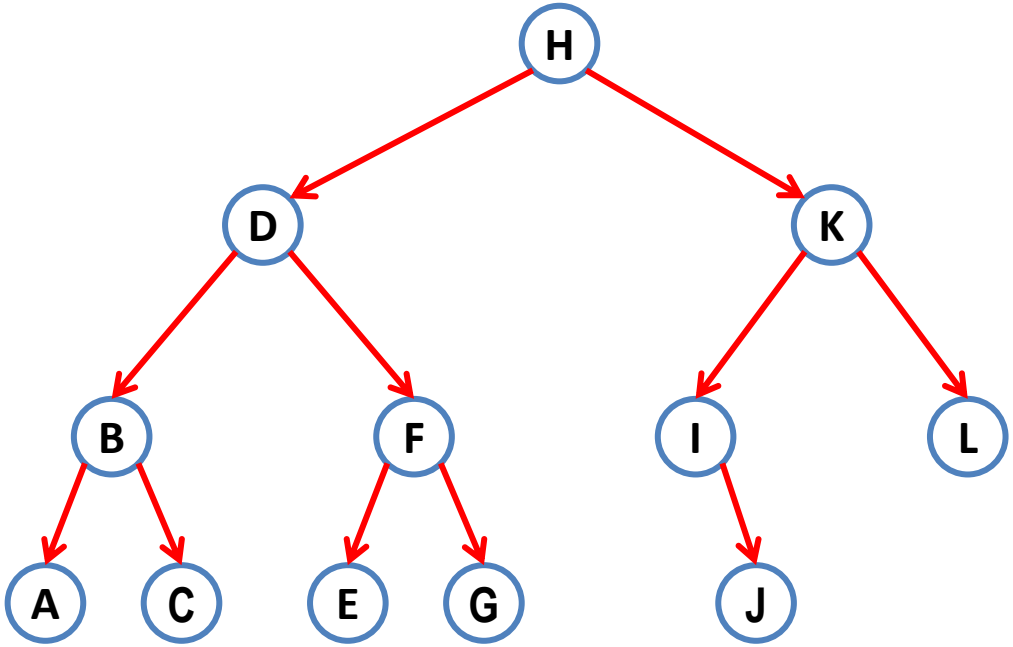
- Binary Tree Traversal
  - Preorder traversal
    mengunjungi node terbawah hingga mencapai setiap children node dengan urutan:
    Data/parent → Left children → Right children



H – D – B – A – C – F – E – G – K – I – J – L

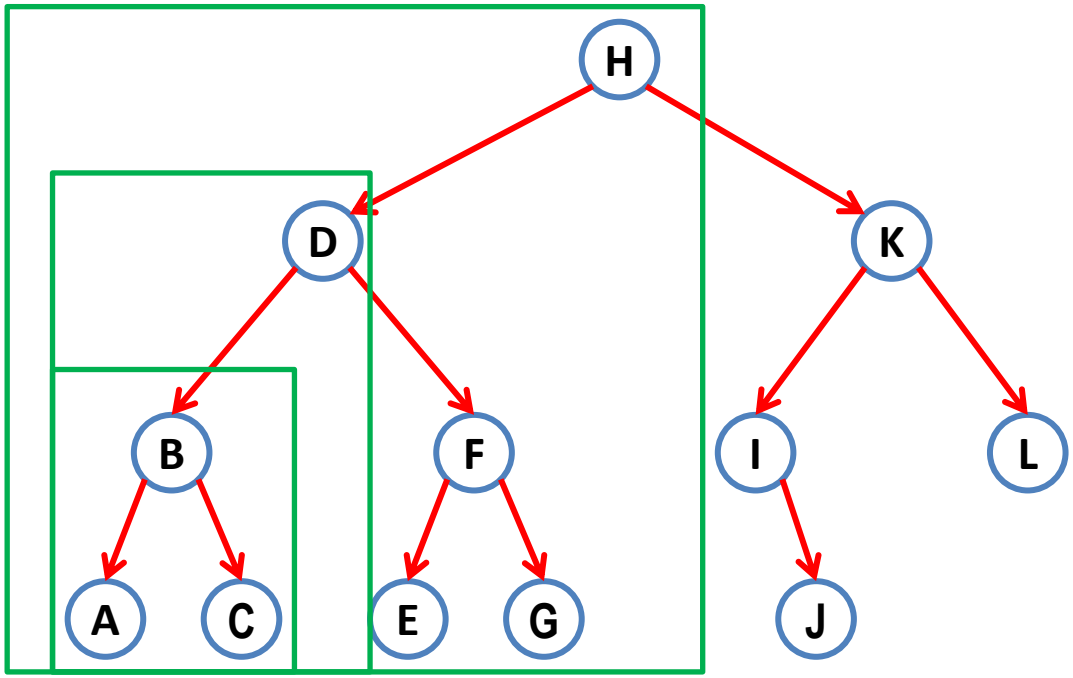
Binary Tree Traversal

- Binary Tree Traversal
  - Inorder traversal
    mengunjungi node terbawah hingga mencapai setiap children node dengan urutan:
    Left children → Data/parent → Right children



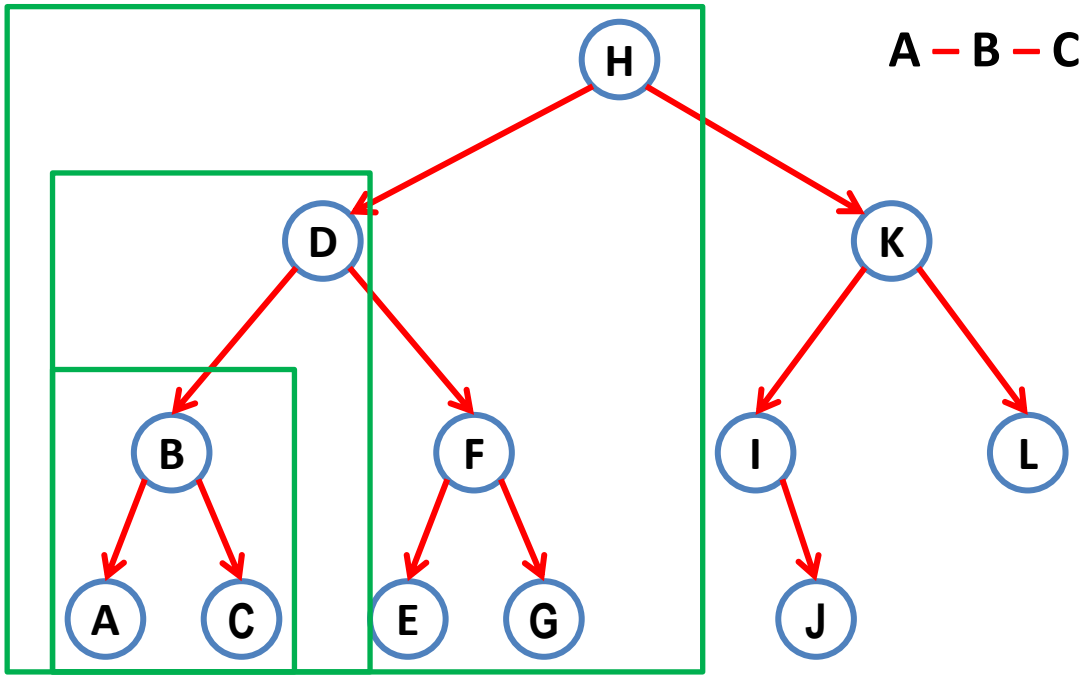
Binary Tree Traversal

- Binary Tree Traversal
  - Inorder traversal
    mengunjungi node terbawah hingga mencapai setiap children node dengan urutan:
    Left children → Data/parent → Right children



Binary Tree Traversal

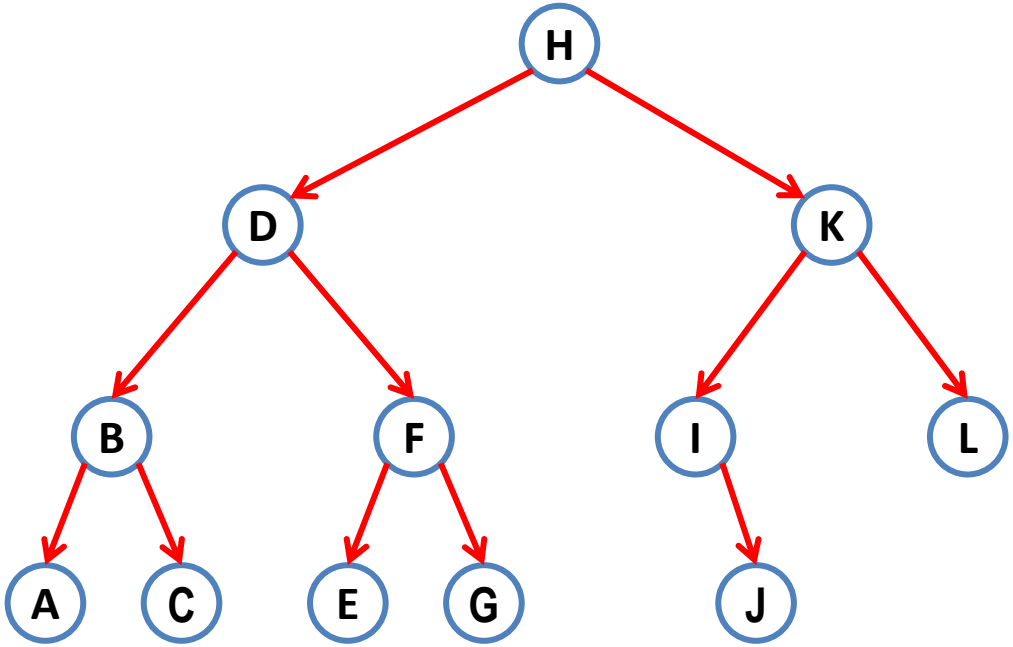
- Binary Tree Traversal
  - Inorder traversal
    mengunjungi node terbawah hingga mencapai setiap children node dengan urutan:
    Left children → Data/parent → Right children



A – B – C – D – E – F – G – H – I – J – K – L

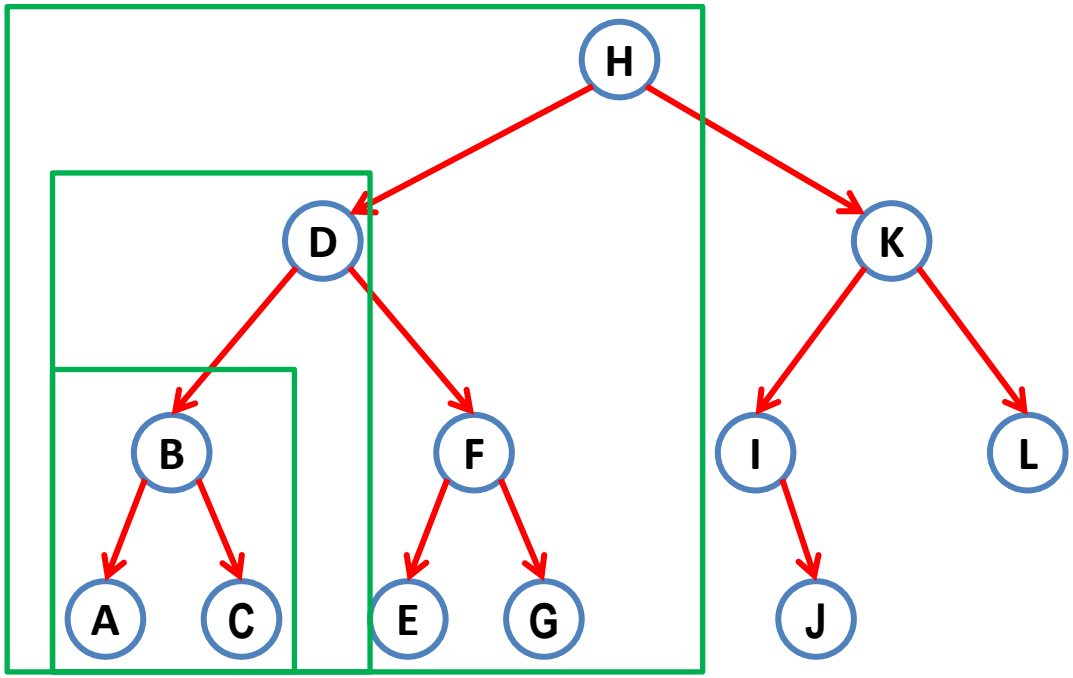
Binary Tree Traversal

- Binary Tree Traversal
  - Postorder traversal
    mengunjungi node terbawah hingga mencapai setiap children node dengan urutan:
    Left children → Right children → Data/parent



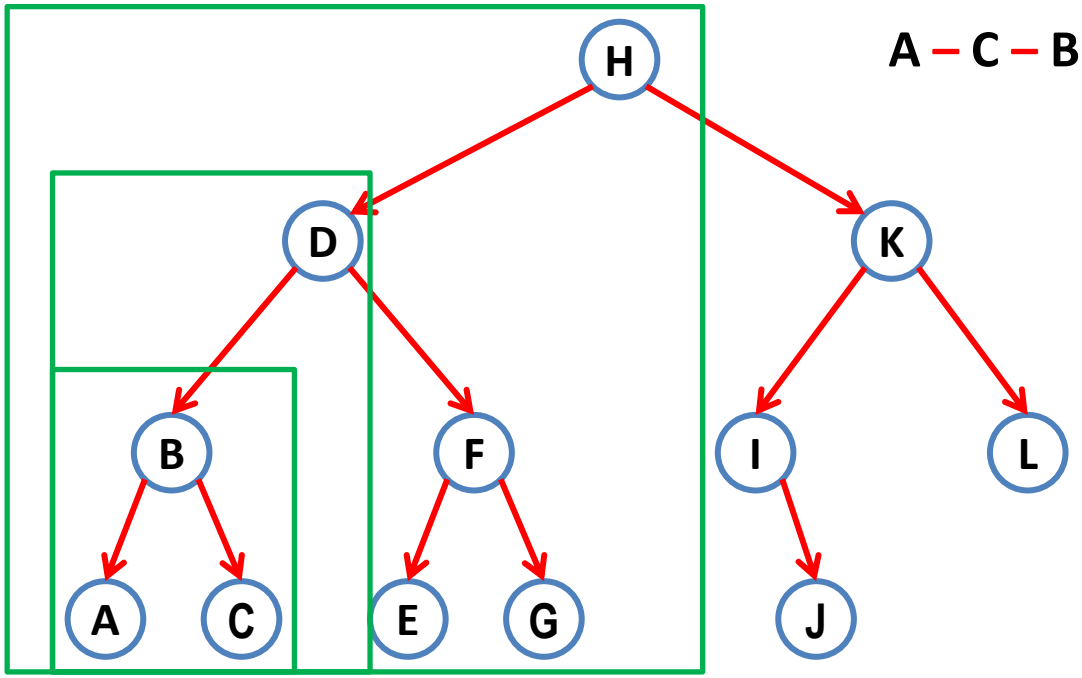
Binary Tree Traversal

- Binary Tree Traversal
  - Postorder traversal
    mengunjungi node terbawah hingga mencapai setiap children node dengan urutan:
    Left children → Right children → Data/parent



Binary Tree Traversal

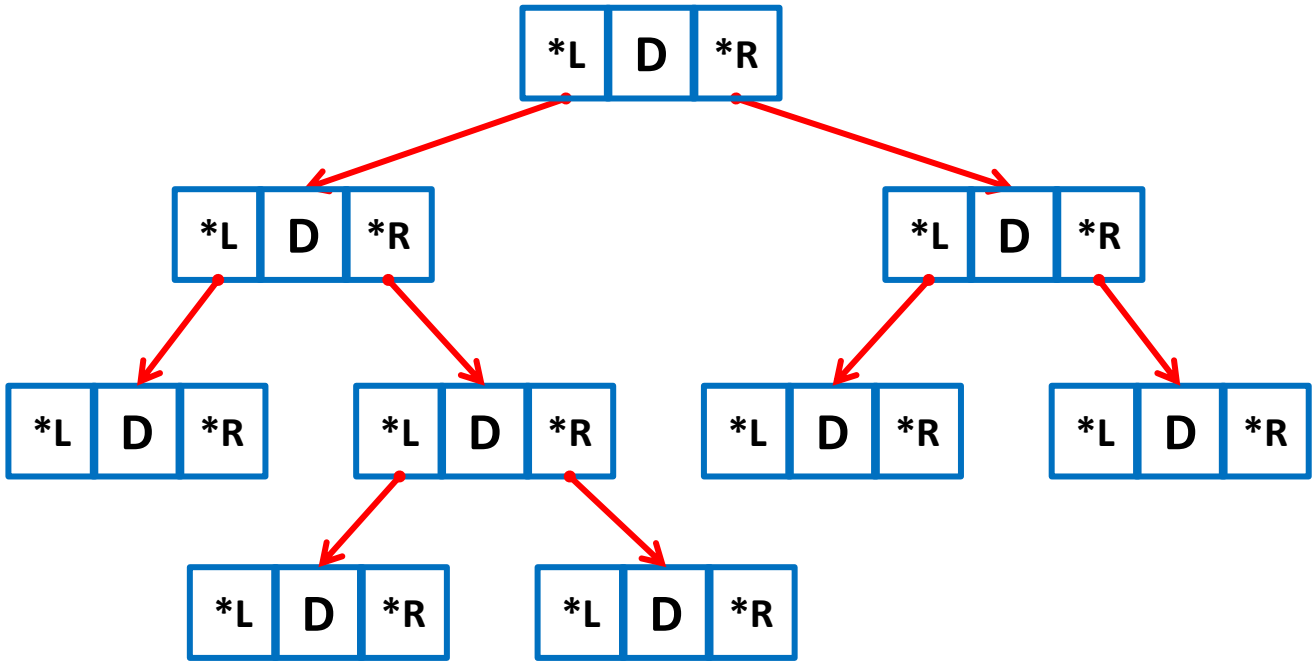
- Binary Tree Traversal
  - Postorder traversal
    mengunjungi node terbawah hingga mencapai setiap children node dengan urutan:
    Left children → Right children → Data/parent



A – C – B – E – G – F – D – J – I – L – K – H

Pembentukkan Binary Tree

- Binary tree dibentuk dengan node yang mempunyai **D**ata dan dua buah pointer/link (***L**eft dan ***R**ight)



Sekian

TERIMAKASIH