VERSION 1.0

OKTOBER 4, 2021

[STRUKTUR DATA]

MODUL 5, TREE

DISUSUN OLEH: - BELLA DWI MARDIANA
- LIDYA FANKKY OKTAVIA P.

DIAUDIT OLEH: - DIDIH RIZKI CHANDRANEGARA, S.KOM., M.KOM.
- IR. GITA INDAH MARTHASARI, ST., M.KOM.

PRESENTED BY: TIM LAB-IT

UNIVERSITAS MUHAMMADIYAH MALANG

[STRUKTUR DATA]

PERSIAPAN MATERI

Mahasiswa diharapkan mempelajari materi sebelum mengerjakan tugas, materi yang tercakup antara lain:

1. Tree
2. Binary Tree

TUJUAN

Mahasiswa mampu menguasai & menjelaskan konsep dari struktur data dari Tree

TARGET MODUL

Mahasiswa mampu memahami & menerapkan Tree beserta contohnya

PERSIAPAN SOFTWARE/APLIKASI

1. Java Development Kit
2. Java Runtime Environment
3. IDE (Intellij IDEA, Eclipse, Netbeans, dll)

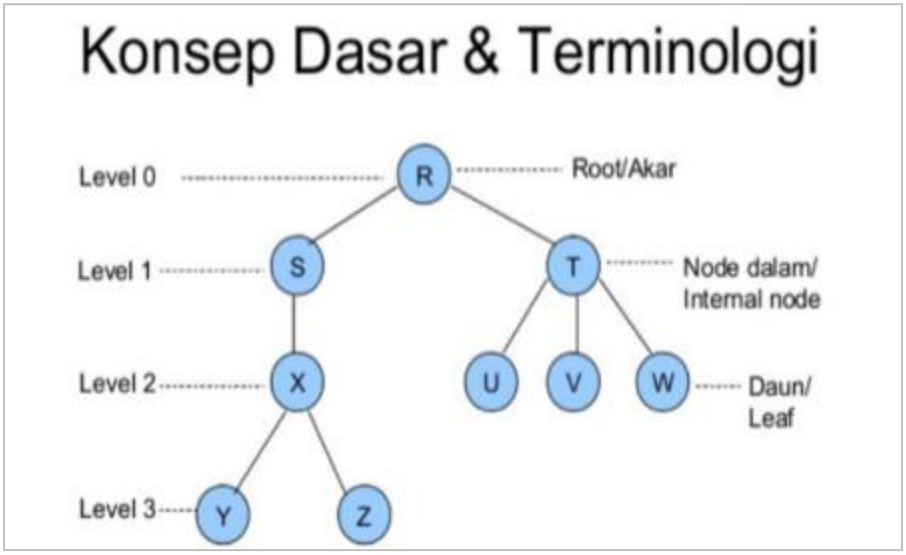
MATERI POKOK

Penjelasan:

1. Tree

Tree merupakan salah satu bentuk struktur data tidak linear yang menggambarkan hubungan bersifat hierarki antara elemen – elemen. Tree didefinisikan sebagai kumpulan simpul (node) dengan salah satu simpul yang dijadikan akar (root). Simpul lainnya terbagi menjadi himpunan yang saling tak berhubungan satu sama lain (subtree).

Node – node tersebut dihubungkan oleh sebuah vektor. Setiap node dapat memiliki 0 atau lebih node anak (child). Sebuah node yang memiliki node anak disebut node induk (parent). Sebuah node anak hanya memiliki satu node induk. Sesuai konvensi ilmu komputer, tree tumbuh ke bawah, tidak seperti pohon di dunia nyata yang tumbuh ke atas. Dengan demikian, node anak akan digambarkan berada dibawah node induknya. Berikut adalah contoh ilustrasi bentuk tree:



Gambar 1 Konsep Dasar Tree

Istilah Pada Tree

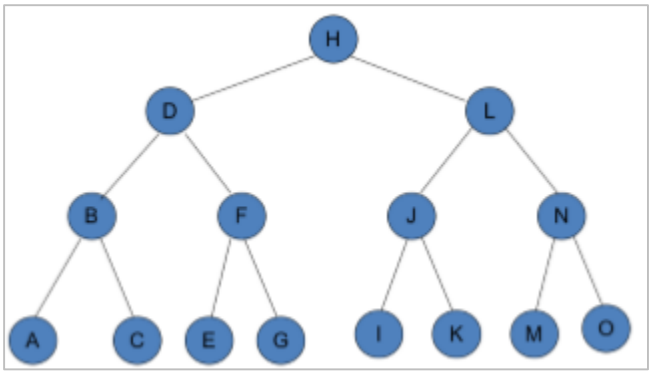
Istilah	Keterangan
Predecessor	Node yang berada di atas node tertentu
Successor	Node yang berada di bawah node tertentu
Ancestor	Seluruh node yang terletak sebelum node tertentu dan terletak pada jalur yang sama
Descendant	Seluruh node yang terletak setelah node tertentu dan terletak pada jalur yang sama
Parent	Predecessor satu level di atas suatu node
Child	Successor satu level di bawah suatu node
Sibling	Node – node yang memiliki parent yang sama
Subtree	Suatu node beserta descendant-nya
Size	Banyaknya node dalam suatu tree
Height	Banyaknya tingkatan dalam suatu tree
Root	Node khusus yang tidak memiliki predecessor
Leaf	Node – node dalam tree yang tidak memiliki successor
Degree	Banyaknya child dalam suatu node

2. Binary Tree

Binary Tree adalah tree dengan syarat bahwa tiap node hanya boleh memiliki maksimal 2 subtree dan kedua subtree tersebut harus dipisah. Sesuai dengan definisi tersebut, maka tiap node dalam Binary Tree hanya boleh memiliki paling banyak dua child. Berikut adalah jenis – jenis Binary Tree:

a. Full Binary Tree

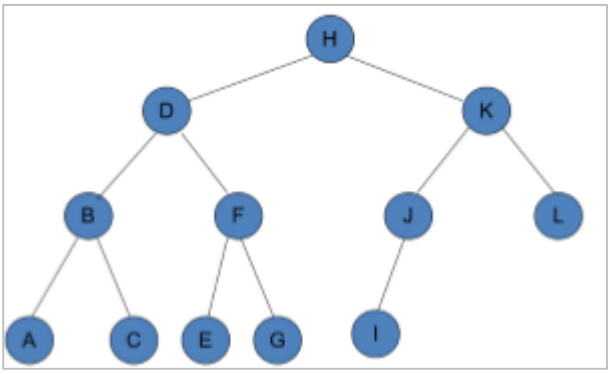
Merupakan Binary Tree yang tiap node-nya (kecuali leaf) memiliki dua child dan tiap subtree harus mempunyai panjang path yang sama. Berikut contohnya:



Gambar 2 Full Binary Tree

b. Complete Binary tree

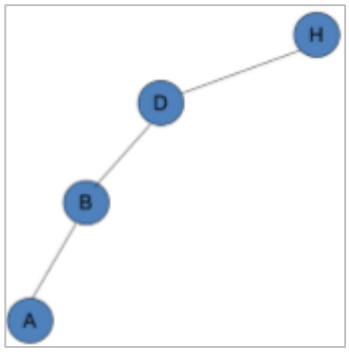
Complete Binary tree ini hampir sama dengan Full Binary Tree, namun tiap subtree boleh memiliki panjang path yang berbeda. Node kecuali leaf memiliki 0 atau 2 child. Berikut contohnya:



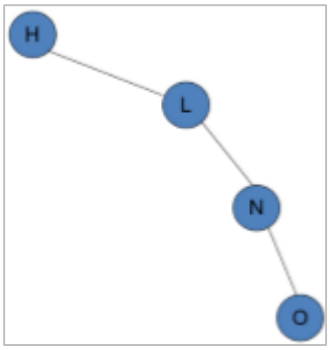
Gambar 3 Complete Binary Tree

c. Skewed Binary Tree

Merupakan Binary Tree yang semua node-nya (kecuali leaf)hanya memiliki satu child. Berikut contohnya:



Gambar 4 Left Skewed

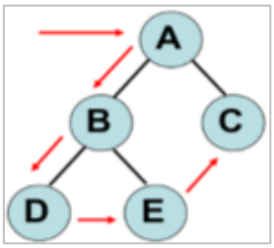


Gambar 5 Right Skewed

3. Binary Tree Traversal

Binary Tree Traversal adalah proses mengunjungi node tepat satu kali dan tiap node hanya boleh memiliki maksimal 2 subtree yang disebut sub pohon kiri (left subtree) dan sub pohon kanan (right subtree). Dengan melakukan kunjungan secara lengkap, maka akan didapatkan urutan informasi secara linier yang tersimpan dalam sebuah Binary Tree. Terdapat 3 metode dalam Binary Tree Traversal, yaitu:

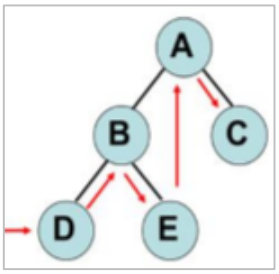
a. PreOrder



Urutan PreOrder:

- Cetak isi simpul yang dikunjungi (root)
- Kunjungi cabang kiri
- Kunjungi cabang kanan

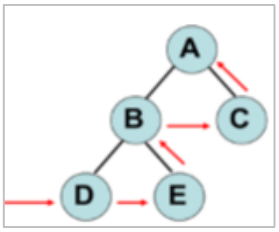
b. InOrder



Urutan InOrder:

- Kunjungi cabang kiri
- Cetak isi simpul yang dikunjungi (root)
- Kunjungi cabang kanan

c. PostOrder



Urutan PostOrder:

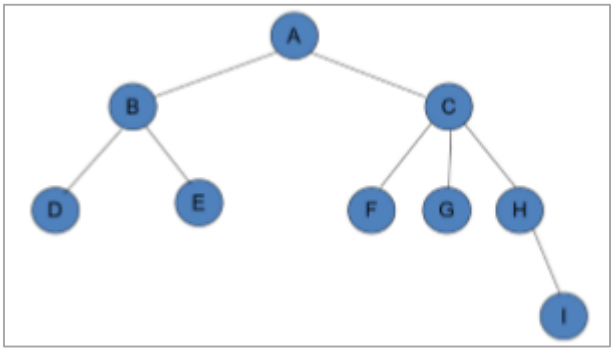
- Kunjungi cabang kiri
- Kunjungi cabang kanan
- Cetak isi simpul yang dikunjungi (root)

Kenapa Menggunakan Tree?

Tree merupakan teori yang sangat bermanfaat dalam struktur data karena dapat digunakan sebagai struktur dalam penyimpanan data yang baik dalam berbagai kasus. Dapat digunakan untuk menyimpan dan mencari data dalam teknik pemrograman serta bagaimana cara struktur data Tree dapat dimanfaatkan untuk menyimpan dan mencari data dengan cepat dan efisien serta mengurangi bug dalam komputer.

Contoh dan Istilah Dalam Binary Tree

Disediakan gambar tree seperti berikut:



Dengan melihat gambar diatas, dapat diketahui bahwa:

a. Predesesor (F) = **A, B, C**
b. Successor (B) = **D, E, F, G, H, I**
c. Ancestor (F) = **C, A**
d. Descendant (B) = **D, E**
f. Parent (I) = **H**
g. Child (C) = **F, G, H**
g. Sibling (G) = **F, H**
h. Degree (C) = **3**
i. Height = **4**
j. Root = **A**
k. Leaf = **D, E, F, G, I**
l. Size = **9**

*jika dilihat dari soal diatas, yang menjadi node patokan adalah didalam tanda kurung ().

LATIHAN PRAKTIKUM

Contoh Kasus Merubah Urutan Data Menjadi Binary Tree

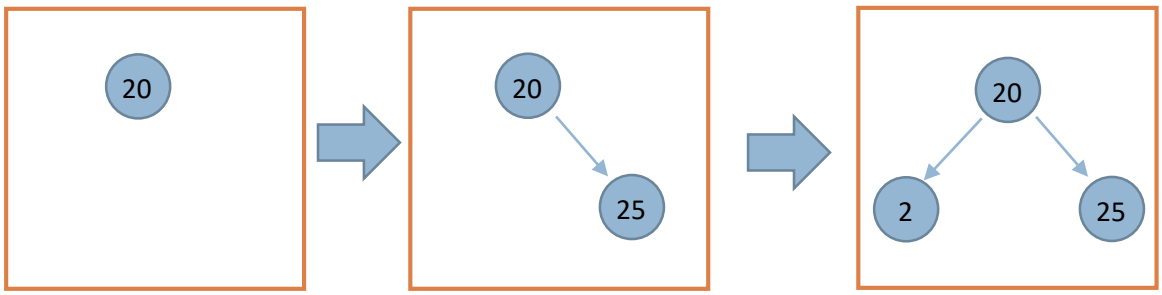
Aturan umum:

a) Jika binary tree masih kosong, maka nilai pertama langsung menjadi root
b) Jika nilai berikutnya yang akan dimasukkan lebih kecil (<) dari node yang sudah ada, maka diletakkan di bagian kiri bawah node yang sudah ada
c) Jika nilai berikutnya yang akan dimasukkan lebih besar atau sama dengan (>=) dari node yang sudah ada maka diletakkan di bagian kanan bawah node yang sudah ada tersebut
d) Pengecekan untuk peletakkan node dilakukan berulang – ulang sampai nilai yang akan dimasukkan tersebut menjadi leaf

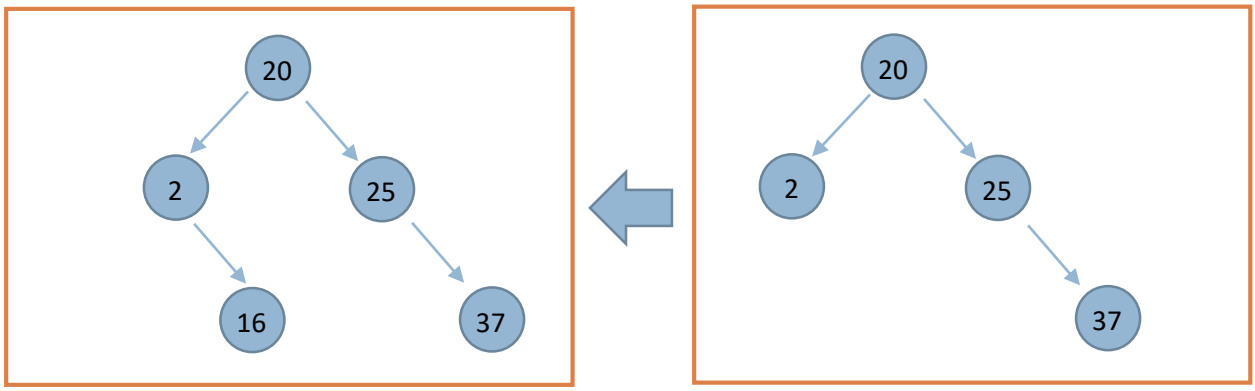
Contoh:

Urutan data : 20 2 25 37 16

Jika diubah menjadi Binary Tree yaitu:



HASIL AKHIR:



NB: Jika data berupa selain angka, maka pengurutannya berdasarkan nilai ASCII

Tree Traversal Source Code

Dari data pohon yang sudah ada diatas, dapat dibuat dalam program untuk menentukan Tree Traversal seperti berikut:

a. Membuat class Node.java untuk deklarasikan node

```java
public class Node {
    int data;
    Node left;
    Node right;

    public Node(int data) {
        this.data = data;
    }
}
```

b. Membuat class BinaryTree.java untuk memproses menjadi urutan (pre order, in order, dan post order)

```java
public class BinaryTree {
    public Node root;

    public void NewNode(int data){
        root = NewNode(root, new Node(data));
    }

    private Node NewNode(Node root, Node newData) {
        if (root == null) {
            root = newData;
            return root;
        }
        if (newData.data < root.data) {
            root.left = NewNode(root.left, newData);
        } else {
            root.right = NewNode(root.right, newData);
        }
        return root;
    }
    public void inOrder(Node node) {
        if(node != null){
            inOrder(node.left);
            System.out.print(node.data + " ");
            inOrder(node.right);
        }
    }

    public void preOrder(Node node){...}
    public void postOrder(Node node){...}
```

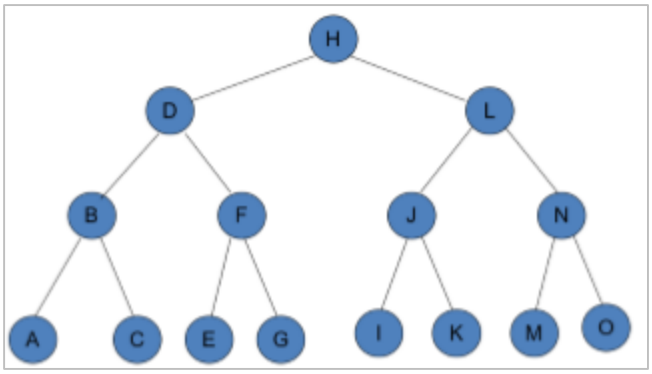
c. Membuat driver class Main.java

```
public class Main {

    public static void main(String[] args){
        BinaryTree pohon = new BinaryTree();

        pohon.NewNode( data: 20);
        pohon.NewNode( data: 2);
        pohon.NewNode( data: 25);
        pohon.NewNode( data: 37);
        pohon.NewNode( data: 16);

        System.out.print("\nPre Order : ");
        pohon.preOrder(pohon.root);
        System.out.print("\nIn Order : ");
        pohon.inOrder(pohon.root);
        System.out.print("\nPost Order : ");
        pohon.postOrder(pohon.root);
    }
}
```

TUGAS PRAKTIKUM

KEGIATAN 1

Disediakan gambar Full Binary Tree berikut:



Dari gambar yang sudah ada diatas, berikan jawaban dengan jelas dari pertanyaan yang diberikan oleh asisten (pertanyaan akan diberikan langsung pada saat praktikum, pertanyaan berhubungan dengan istilah pada Binary Tree). Berikut adalah contoh pertanyaan yang akan ditanyakan oleh asisten:

a. **Predesesor (F)**
b. **Successor (B)**
c. **Ancestor (N)**

*pastikan untuk mempelajari dan memahami **semua** istilah Binary Tree agar dapat menjawab pertanyaan dengan benar. Asisten diperbolehkan memberikan maksimal 8 poin pertanyaan.

KEGIATAN 2

Buatlah sebuah source code (**program kreasi sendiri)** yang dapat menerima inputan dari user berupa kombinasi (bisa memanfaatkan huruf dan angka) dan gambarkan **Binary Tree-nya**. Program yang dibuat harus bisa menerapkan konsep Tree Traversal (**pre order, in order, dan post order**).

NB:

a. Arti dari kreasi sendiri adalah topik program yang diambil berasal dari pemikiran Praktikan masing – masing dan pastinya setiap orang akan memiliki pemikiran yang berbeda.
b. Inputan kombinasi dapat bersifat opsional jika praktikan dapat memberikan alasan yang jelas dan bisa digantikan dengan pemahaman materinya.

CATATAN

Aturan umum penulisan JAVA agar mudah dikoreksi oleh asisten:

1. Untuk nama class, enum, dan yang lainnya biasakan menggunakan gaya CamelCase (diawali dengan huruf besar pada tiap kata untuk mengganti spasi) seperti: Kursi, JalanRaya, ParkiranGedung, dan lain seterusnya.
2. Untuk penulisan nama method dan attribute diawali dengan huruf kecil di awal kata dan menggunakan huruf besar untuk kata setelahnya, seperti: getNamaJalan, namaJalan, harga, setNamaJalan, dan lain seterusnya.
3. Jika menggunakan IDE Intellij jangan lupa untuk memformat kode agar terlihat rapi menggunakan menu code -> show reformat file dialog -> centang semua field dan klik ok.

DETAIL PENILAIAN TUGAS

Kriteria	Nilai
Kegiatan 1	40
Kegiatan 2	45
Pemahaman Materi	15