Laporan Praktikum

Struktur Data



Disusun Oleh :

**Ahda Rindang Al-Amin (2311531003)**

Dosen Pengampu : Dr. Wahyudi, S.T, M.T

Departemen Informatika

Fakultas Teknologi Informasi

Universitas Andalas

Tahun 2024

**Binary Tree**

1. **Tujuan Praktikum**
2. Memahami Binary Tree pada struktur data.
3. **Pendahuluan**

Binary Tree merupakan salah satu bentuk struktur data tidak linear yang menggambarkanhubungan yang bersifat hirarkis (hubungan one to many) antara elemen-elemen. Binary Treeadalah tree dengan syarat bahwa tiap node hanya boleh memiliki maksimal dua subtree dankedua subtree tersebut harus terpisah. Binary Tree merupakan himpunan vertex-vertex yangterdiri dari 2 subtree (dengan disjoint) yaitu subtree kiri dan subtree kanan. Setiap vertexdalam binary tree mempunyai derajat keluar max = 2.

Sebuah pohon biner adalah grafik asiklis yang terhubung dimana setiap tingkatan darisusut tidak lebih dari 3. Ini dapat ditunjukkan bahwa dalam pohon biner manapun, terdapat persis dua atau lebih simpul dengan tingkat satu daripada yang terdapat dengan tingkat tiga,tetapi bisa terdapat angka apa saja dari simpul dengan tingkat dua. Sebuah pohon biner berakar merupakan sebuah grafik yang mempunyai satu dari sudutnya dengan tingkat tidaklebih dari dua sebagai akar.

Dengan akar yang dipilih, setiap sudut akan memiliki ayah khusus, dan diatas dua anak bagaimanapun juga, sejauh ini terdapat keterbatasan informasi untuk membedakan antaraanak kiri atau kanan. Jika kita membuang keperluan yang tak terkoneksi, membolehkan bermacam koneksi dalam komponen di grafik, kita memanggil struktur sebuah hutan.Sebuah jalan lain untuk mendefinisikan pohon biner melalui definisi rekursif pada grafiklangsung. Sebuah pohon biner dapat berarti :

* Sebuah sudut tunggal.
* Sebuah graf yang dibentuk dengan mengambil dua pohon biner, menambahkan sebuahsudut, dan menambahkan sebuah panah langsung dari sudut yang baru ke akar dai setiap pohon biner.

Pohon biner dapat dikontruksi dari bahasa pemrogaraman primitif dalam berbagai cara.Dalam bahasa yang menggunakan records dan referensi. Pohon biner secara khas dikontruksidengan mengambil sebuah struktur simpul pohon yang memuat beberapa data dan referensike anak kiri dan anak kanan.

Kadang-kadang itu juga memuat sebuah referensi ke ayahnya yang khas. Jika sebuah simpulmempunyai kurang dari dua anak, beberapa penunjuk anak diaatur kedalam nilai nol khususatau kesebuah simpul sentinel.

Pohon biner dapat juga disimpan sebagai struktur data implisit dalam array, dan jika pohon tersebut merupakan sebuah pohon biner lengkap, metode ini tidak boros tempat. Dalam penyusunan yang rapat ini, jika sebuah simpul memiliki indeks i, anaknya dapat ditemukan pada indeks ke-2i+1 dan 2i+2, meskipun ayahnya (jika ada) ditemukan pada indeks lantai ((i-1)/2) (asumsikan akarnya memiliki indeks kosong). Metode ini menguntungkan dari banyak penyimpanan yang rapat dan memiliki referensi lokal yang lebih baik, teristimewa selamasebuah preordeer traversal.

1. **Metode Praktikum**
2. **Kelas Node**

Kode berikut adalah implementasi dari kelas Node yang digunakan dalam pohon biner. Kelas Node ini mengandung informasi dan metode untuk mengelola node individu dalam pohon biner. Kelas Node ini menyediakan implementasi dasar untuk node dalam pohon biner, termasuk pengaturan anak kiri dan kanan, pengambilan dan pengaturan data, dan berbagai metode untuk traversal dan pencetakan struktur pohon. Kelas ini melengkapi kelas BTree dengan menyediakan fungsi dasar yang dibutuhkan untuk mengelola node individu dalam pohon biner.

public class Node {

int data;

Node left;

Node right;

public Node(int data) {

this.data = data;

left=null;

right=null;

}

public void setLeft(Node node) {

if (left==null)

left=node;

}

public void setRight(Node node) {

if (right==null)

right=node;

}

public Node getLeft(){

return left;

}

public Node getRight(){

return right;

}

public int getData(){

return data;

}

public void setData(int data) {

this.data=data;

}

void printPreorder(Node node) {

if(node==null)

return;

System.***out***.print(node.data + " ");

printPreorder(node.left);

printPreorder(node.right);

}

void printPostorder(Node node) {

if (node==null)

return;

printPostorder(node.left);

printPostorder(node.right);

System.***out***.print(node.data+" ");

}

void printInorder(Node node) {

if (node==null)

return;

printInorder(node.left);

System.***out***.print(node.data+" ");

printInorder(node.right);

}

public String print() {

return

this.print("",true,"");

}

public String print (String prefix, boolean isTail, String sb) {

if (right != null) {

right.print(prefix + (isTail ?"| " : " "), false, sb);

}

System.***out***.println(prefix+(isTail ?"\\-- ":"/-- ")+data);

if (left != null) {

left.print(prefix + (isTail ?" " : "| "), true, sb);

}

return sb;

}

}

1. **Kelas Binary Tree**

Kelas BinaryTree ini mendefinisikan struktur dasar dan fungsi-fungsi untuk mengelola pohon biner, termasuk pencarian data, traversal (penjelajahan) dalam berbagai urutan, penghitungan jumlah node, dan pengaturan node saat ini dan akar pohon. Implementasi lengkap dari kelas Node diperlukan untuk menjalankan fungsi-fungsi seperti printInorder, printPreorder, printPostorder, dan print.

public class BinaryTree {

private Node root;

private Node currentNode;

public BinaryTree() {

root = null;

}

public boolean search(int data) {

return search(root, data);

}

private boolean search(Node node, int data) {

if(node.getData()== data)

return true;

if (node.getLeft()!=null)

if(search(node.getLeft(),data))

return true;

if(node.getRight()!=null)

if(search(node.getRight(),data))

return true;

return false;

}

public void printInorder() {

root.printInorder(root);

}

public void printPreorder() {

root.printPreorder(root);

}

public void printPostorder() {

root.printPostorder(root);

}

public Node getRoot() {

return root;

}

public boolean isEmpty() {

return root==null;

}

public int countNodes() {

return countNodes(root);

}

private int countNodes(Node node) {

int count = 1;

if (node==null) {

return 0;

} else {

count += countNodes(node.getLeft());

count += countNodes(node.getRight());

return count;

}

}

public void print() {

root.print();

}

public Node getCurrent() {

return currentNode;

}

public void setCurrent(Node node) {

this.currentNode = node;

}

public void setRoot(Node node) {

this.root = node;

}

}

1. **Kelas Tree Main**

Kelas TreeMain menjadi program utama yang menggunakan kelas BTree dan Node untuk membangun dan mengelola pohon biner. Pada kelas ini dilakukan beberapa operasi pada pohon biner dan ditampilkan hasilnya. Kelas TreeMain ini mendemonstrasikan bagaimana membangun pohon biner, menambah node, menghitung jumlah node, melakukan traversal pohon dalam berbagai urutan, dan menampilkan struktur pohon.

public class TreeMain {

public static void main(String[] args) {

//membuat pohon

BinaryTree tree = new BinaryTree();

System.***out***.print("Jumlah simpul pohon : ");

System.***out***.println(tree.countNodes());

//menambahkan simpul data 1

Node root = new Node(1);

//menjadikan simpul 1 sebagai root

tree.setRoot(root);

System.***out***.print("Jumlah simpul jika hanya ada root : ");

System.***out***.println(tree.countNodes());

Node node2 = new Node(2);

Node node3 = new Node(3);

Node node4 = new Node(4);

Node node5 = new Node(5);

Node node6 = new Node(6);

Node node7 = new Node(7);

root.setLeft(node2);

node2.setLeft(node4);

node2.setRight(node5);

node5.setLeft(node7);

root.setRight(node3);

node3.setRight(node6);

//set root

tree.setCurrent(tree.getRoot());

System.***out***.print("menampilkan simpul terakhir : ");

System.***out***.println(tree.getCurrent().getData());

System.***out***.print("Jumlah simpul setelah simpul 7 ditambahkan");

System.***out***.println(tree.countNodes());

System.***out***.print("InOrder : ");

tree.printInorder();

System.***out***.print("\nPreOrder : ");

tree.printPreorder();

System.***out***.print("\nPostOrder : ");

tree.printPostorder();

System.***out***.println("\nMenampilkan simpul dalam bentuk pohon");

tree.print();

}

}

1. **Kesimpulan Praktikum**

Dari praktikum yang telah dilaksanakan dapat diambil Kesimpulan bahwa ketiga program diatas saling mendefinisikan, membangun, dan memanipulasi pohon biner (binary tree) yang Dimana setiap program memiliki fungsinya masing-masing Kelas BTree mengelola struktur pohon biner dan menyediakan berbagai metode untuk berinteraksi dengan pohon tersebut. Kelas Node mendefinisikan node individual dalam pohon biner dan menyediakan metode untuk mengelola node tersebut. Kelas TreeMain adalah kelas utama yang menggunakan BTree dan Node untuk membangun dan mengelola pohon biner.