Bab 7. TREE

1.1. Dfinisi tree

Tree merupakan salah satu bentuk struktur data tidak linear yang menggambarkan hubungan yang bersifat hierarkis (hubungan one to many) antara elemen-elemen. Tree biasa didefinisikan sebagai kumpulan simpul/node dengan elemen khusus yang disebut Root. Notde lainnya terbagi menjadi himpunan-himpunan yang saling tak berhubungan satu sama lain (disebut Subtree). Untuk lebih jelasnya, di bawah akan diuraikan istilahistilah umum dalam tree.

Predecessor	Node yang berada di atas node tertentu
Successor	Node yang berada dibawah node tertentu
Ancestor	Seluruh node yang terletak sebelum node tertentu dan terletak pada jalur yang sama
Descendant	Seluruh node yang terletak sebelum 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 dengan suatu node
Subtree	Bagian dari tree yang berupa suatu node beserta descendantnya dan memiliki semua karakteristik dari tree tersebut.
Size	Banyaknya node dalam suatu tree
Height	Banyaknya tingkatan / level dalam suatu tree
Root	Satu-satunya node khusus dalam tree yang tak punyak predecessor
Leaf	Node-node dalam tree yang tak memiliki successor
Degree	Banyaknya child yang dimiliki suatu node

1.2. Jenis Jenis Binary Tree

a. Binary Tree

Binary Tree adalah tree dengan syarat bahwa tiap node hanya boleh memiliki maksimal dua subtree dan kedua subtree tersebut harus terpisah. Sesuai dengan definisi tersebut tiap node dalam binary tree hanya boleh memiliki paling banyak dua child.

Jenis- Jenis Binary Tree :

b. Full Binary Tree

Jenis binary tree ini tiap nodenya (kecuali leaf) memiliki dua child dan tiap subtree harus mempunyai panjang path yang sama.

c. Complete Binary Tree

Jenis ini mirip dengan Full Binary Tree, namun tiap subtree boleh memiliki panjang path yang berbeda dan setiap node kecuali leaf hanya boleh memiliki 2 child.

d. Skewed Binary Tree

Skewed Binary Tree adalah Binary Tree yang semua nodenya (kecuali leaf) hanya memiliki satu child.

Implementasi Binary Tree

Binary tree dapat diimplementasikan dalam C++ dengan menggunakan double linkedlist.

1.3. Operasi Pada Binary search Tree

Create	Membentuk binary tree baru yang masih kosong
Clear	Mengosongkan binary tree yang sudah ada
Empty Function	untuk memeriksa apakah binary tree masih kosong
Insert	Memasukkan sebuah node ke dalam tree. Ada tiga pilihan insert : sebagai root, left child, atau right child. Khusus insert sebagai root, tree harus dalam keadaan kosong
Find	Mencari root, parent, left child, atau right child dari suatu node. (tree tidak boleh kosong).
Update	Mengubah isi dari node yang ditunjuk oleh pointer curret (Tree tidak boleh kosong)
Retrieve	Mengetahui isi dari node yang ditunjuk oleh pointer current (Tree tidak boleh kosong)
DeleteSub	Menghapus sebuah subtree (node beserta seluruh descendantnya) yang ditunjuk current. Tree tidak boleh kosong. Setelah itu, pointer current dakan berpindah ke parent dari node yang dihapus.
Characteristic	Mengetahui karakteristik dari suatu tree, yakni: size, height, serta average length. Tree tidak boleh kosong.

Traverse	Mengunjungi seluruh node-node pada tree, masing-masing sekali. Hasilnya adalah urutan informasi secara linear yang tersimpan dalam tree. Ada tiga cara traverse,yaitu PreOrder, InOrder, dan PostOrder.

Langkah Langkah Transverse :

- PreOrder : cetak isi node yang dikunjungi, kunjungi Left Child, kunjungi Right Child
- InOrder : kunjungi Left Child, cetak isi node yang dikunjungi, kunjungi Right Child
- PostOrder : kunjungi Left Child, kunjungi Right Child cetak isi node yang dikunjungi.

1.4. Binary Search Tree

Binary Tree ini memiliki sifat dimana semua left child harus lebih kecil dari pada right child dan parentnya. Semua right child juga harus lebih besar dari left child serta parentnya. Binary search tree dibuat untuk mengatasi kelemahan pada binary tree biasa, yaitu kesulitan dalam searching / pendarian node tertentu dalam binary tree.
Pada dasarnya operasi dalam Binary Search Tree sama dengan Binary Tree biasa, kecuali pada operasi insert, update, dan delete.

Insert

Pada Binary Search Tree insert dilakukan setelah lokasi yang tepat ditemukan (lokasi tidak ditentukan oleh user sendiri).

Update

Update ini seperti yang ada pada Binary Tree biasa, namun di sini update akan berpengaruh pada posisi node tersebut selanjutnya. Bila update mengakibatkan tree tersebut bukan Binary Search Tree lagi, harus dilakukan perubahan pada tree dengan melakukan rotasi supaya tetap menjadi Binary Search Tree.

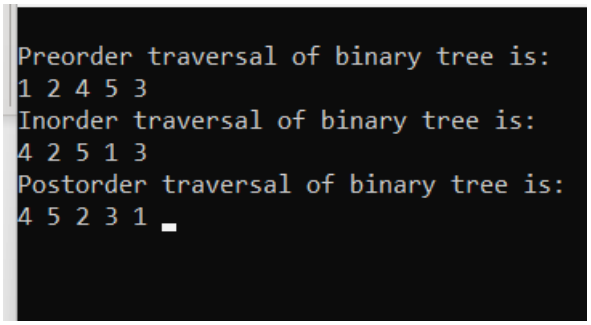
Delete

Seperti halnya update, delete dalam Binary Search Tree juga turut mempengaruhi struktur dari tree tersebut.

Contoh Program :

```cpp
#include <iostream>
using namespace std;

struct node {
    int data;
    struct node* left;
    struct node* right;
};

struct node* newNode(int data){
    struct node* node = new struct node;
    node->data = data;
    node->left = NULL;
    node->right = NULL;
    return(node);
}

void printPostorder(struct node* node){
    if (node == NULL)
        return;
    printPostorder(node->left);
    printPostorder(node->right);
    cout << node->data << " ";
}

void printInorder(struct node* node){
    if (node == NULL)
        return;
    printInorder(node->left);
    cout << node->data << " ";
    printInorder(node->right);
}

void printPreorder(struct node* node){
    if (node == NULL)
        return;
    cout << node->data << " ";
    printPreorder(node->left);
    printPreorder(node->right);
}

int main(){
    struct node *root = newNode(1);
    root->left = newNode(2);
    root->right = newNode(3);
    root->left->left = newNode(4);
    root->left->right = newNode(5);

    cout << "\nPreorder traversal of binary tree is: \n";
    printPreorder(root);
    cout << "\nInorder traversal of binary tree is: \n";
    printInorder(root);
    cout << "\nPostorder traversal of binary tree is: \n";
    printPostorder(root);

    getchar();
    return 0;
}
```

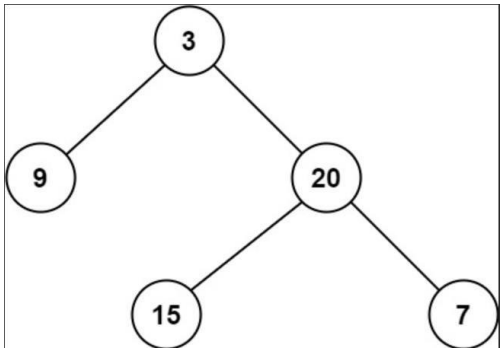
Output


Preorder traversal of binary tree is:
1 2 4 5 3
Inorder traversal of binary tree is:
4 2 5 1 3
Postorder traversal of binary tree is:
4 5 2 3 1

Latihan
Temukan batas kedalaman dari sebuah binary tree.

Batas kedalaman adalah jumlah node jalur terpendek dari akar hingga ke daun. Node daun adalah node yang tidak memiliki turunan (no child)

Contoh



Input : 3,9,20,null,null,15,7
Output : 2