LAPORAN PRAKTIKUM

STRUKTUR DATA DAN ALGORITMA

TREE

DISUSUN OLEH

Latifah Hukma Shobiyya (M0520044)

Asisten:

1. Nuha Lina Atqia (M0519070)
2. Rizal Aji Purbadinata (M0519073)

PROGRAM STUDI S-1 INFORMATIKA

FAKULTAS MATEMATIKA DAN ILMU PENGETAHUAN ALAM

UNIVERSITAS SEBELAS MARET

2021

Source code post order transversal

```cpp
#include<iostream>
using namespace std;
struct node {
   int data;
   struct node *left;
   struct node *right;
};
struct node *createNode(int val) {
   struct node *temp = (struct node *)malloc(sizeof(struct node));
   temp->data = val;
   temp->left = temp->right = NULL;
   return temp;
}
void postOrder(struct node *root) {
    if (root == NULL) return;

    postOrder(root->left);
    postOrder(root->right);
    cout<<root->data<<" ";
}
struct node* insertNode(struct node* node, int val) {
   if (node == NULL) return createNode(val);
   if (val < node->data)
   node->left = insertNode(node->left, val);
   else if (val > node->data)
   node->right = insertNode(node->right, val);
   return node;
}
int main() {
   struct node *root = NULL;
   root = insertNode(root, 57);
   int arr[] = {32,78,24,48,85,39,81,96,35,43,90};
   for(int i=0; i<11; i++){
    insertNode(root, arr[i]);
  }

   cout<<"Post Order traversal of the Binary Search Tree: " << endl;
   postOrder(root);
   cout << endl;
   return 0;
}
```

Analisis source code

```cpp
1  #include<iostream>
```

Baris 1 merupakan preprocessor directive yang berfungsi untuk menyisipkan berkas lain ke dalam berkas yang ingin dikompilasi.

```cpp
2  using namespace std;
```

Baris 2 digunakan untuk mengakses semua fungsi, class, atau file yang terdapat di dalam namespace ‘std’ tanpa harus menuliskan ‘std::’

```cpp
3  struct node {
4      int data;
5      struct node *left;
6      struct node *right;
7  };
```

Baris 3-7 merupakan struct dengan tipe data bernama node yang berisi data dan pointer ke node lain, yaitu pointer ke left dan right. Struct ini merupakan *self refential structure* karena berisi pointer dari tipe struct yang sama.

```
8  struct node *createNode(int val) {
9      struct node *temp = (struct node *)malloc(sizeof(struct node));
10     temp->data = val;
11     temp->left = temp->right = NULL;
12     return temp;
13 }
```

Baris 8-13 merupakan fungsi createNode dengan tipe data node yang memiliki 1 buah parameter (val dengan tipe integer). Fungsi ini membuat temp pointer dan mengalokasikan memorinya menggunakan malloc. Pada baris 10, fungsi akan mengambil val sebagai argumen untuk kemudian disimpan pada temp data. Sedangkan pada baris 11, temp yang pointer ke kanan dan kiri diisi dengan null pointer. Selanjutnya, fungsi akan mereturn temp.

```
14 void postOrder(struct node *root) {
15      if (root == NULL) return;
16
17      postOrder(root->left);
18      postOrder(root->right);
19      cout<<root->data<<" ";
20 }
```

Baris 14-20 merupakan fungsi postOrder dengan parameter berupa pointer root. Pada baris 15, program akan mengecek keberadaan root. Jika root tidak ada, program akan langsung return tanpa nilai apapun. Selanjutnya, terjadi fungsi rekursif dengan argumen berupa root yang pointer ke left (baris 17) dan ke right (baris 18) yang akan menampilkan nilai pada masing-masing cabang sesuai urutan.

```
21 struct node* insertNode(struct node* node, int val) {
22     if (node == NULL) return createNode(val);
23     if (val < node->data)
24     node->left = insertNode(node->left, val);
25     else if (val > node->data)
26     node->right = insertNode(node->right, val);
27     return node;
28 }
```

Baris 21-28 merupakan fungsi insertNode dengan tipe data node yang mengambil 2 parameter. Pada baris 22, jika node berisi nullptr (masih kosong), maka akan direturn ke fungsi createNode dengan nilai val sebagai argumen. Pada baris 23, jika nilai val kurang dari nilai node yang pointer ke data, maka val akan disimpan pada node sebelah kiri. Sebaliknya pada baris 25, jika nilai val lebih besar dari nilai node yang pointer ke data, maka val akan disimpan pada node sebelah kanan.

```
29 int main() {
30     struct node *root = NULL;
31     root = insertNode(root, 57);
32     int arr[] = {32,78,24,48,85,39,81,96,35,43,90};
33     for(int i=0; i<11; i++){
34      insertNode(root, arr[i]);
35    }
36 
37     cout<<"Post Order traversal of the Binary Search Tree: " << endl;
38     postOrder(root);
39     cout << endl;
40     return 0;
41 }
```

Baris 29-41 merupakan main function. Pada baris 30, pointer root diinisiasi sebagai nullptr (masih kosong). Pada baris 31, program akan memanggil fungsi insertNode yang nantinya nilai return dari fungsi ini akan disimpan sebagai root dalam binary tree. Pada baris 32 dideklarasikan array yang berisi 11 elemen nilai, dimana masing-masing nilai akan digunakan sebagai argumen pada fungsi insertNode di dalam for loop. Semua nilai ini kemudian akan disusun dengan posisi yang benar dalam binary tree. Pada baris 38, fungsi postOrder dipanggil dengan menggunakan node root dari binary tree untuk kemudian ditampilkan dalam bentuk post order.

Output

```
clang++-7 -pthread -std=c++17 -o main main.cpp
./main
Post Order traversal of the Binary Search Tree:
24 35 43 39 48 32 81 90 96 85 78 57
```