LAPORAN PRAKTIKUM

Matakuliah	Struktur Data
Pertemuan ke	11
Nama Praktikan	Wijayanto Agung Wibowo
NIM	22.11.4552
NILAI (diisi oleh dosen / asisten praktikum)	

A. Tujuan

Setelah praktikum ini, praktikan diharapkan dapat:

1. Memahami tipe data dasar Tree
2. Memahami tipe data bentukan Tree BST
3. Stuktur Program menggunakan bahasa C++ Praktikum

B. Hasil Percobaan

1. Percobaan 1

a) Tampilan Coding

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

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

//membuat Node Baru
BstNode* GetNewNode(int data) {
    BstNode* newNode = new BstNode();
    newNode->data = data;
    newNode->left = newNode->right = NULL;
    return newNode;
}

//Memasukan data ke BST
BstNode* Insert(BstNode* root, int data) {
    if (root == NULL) {
        root = GetNewNode(data);
    }
    //jika data yang akan disisipkan lebih sedikit, sisipkan di subpohon kiri
    else if (data <= root->data) {
        root->left = Insert(root->left, data);
    }
    //lainnya, masukkan di subpohon kanan
    else {
        root->right = Insert(root->right, data);
    }
    return root;
}

```

```
34  //pencarian element di BST
35  bool search(BstNode* root, int data) {
36      if (root == NULL) {
37          return false;
38      }
39      else if (root->data == data) {
40          return true;
41      }
42      else if (data <= root->data) {
43          return search(root->left, data);
44      }
45      else {
46          return search(root->right, data);
47      }
48  }
49
50  //Tranverse
51  void printInorder(BstNode* root) {
52      if (root == NULL)
53          return;
54      printInorder(root->left);
55      cout << root->data << " ";
56      printInorder(root->right);
57  }
58
59
60
61  int main() {
62      cout << "NAMA: WIJAYANTO AGUNG WIBOWO\n";
63      cout << "NIM: 22.11.4552\n\n";
64      cout << "\t== BINARY SEARCH TREE==\n\n";
65      BstNode* root = NULL; //membuat tree kosong
66      //penambahan node pada tree
67      root = Insert(root, 15);
68      root = Insert(root, 10);
69      root = Insert(root, 20);
70      root = Insert(root, 25);
71      root = Insert(root, 8);
72      root = Insert(root, 12);
73
74      printf("Urutan Data Tree secara Inorder");
75      printf("\n================================\n");
76      printInorder(root);
77
78
79      int number;
80      cout << "\n\nMasukan nomor yang dicari : ";
81      cin >> number;
82      //menampilkan hasil pencarian
83      if (search(root, number) == true) cout << "\nData Ditemukan\n";
84      else cout << "Data Tidak Ditemukan\n";
85  }
```

b) Hasil Running

```
NAMA: WIJAYANTO AGUNG WIBOWO
NIM: 22.11.4552

        == BINARY SEARCH TREE==

Urutan Data Tree secara Inorder
================================
8 10 12 15 20 25

Masukan nomor yang dicari : 12

Data ditemukan

--------------------------------
Process exited after 7.361 seconds with return value 0
Press any key to continue . . .
```

c) Penjelasan

Pertama kali program dibuat maka akan membuat tree dengan hasil NULL.

Selanjutnya akan di masukan data 15,10,20,25,8,12 bergantian dengan memasukan pointer variable root terakhir. Pertama kali data dimasukan , maka itu yang akan dijadikan root, selanjutnya akan bergeser ke kiri dan ke kanan dengan aturan kiri adalah data selanjutnya nilainya lebih kecil dari data root akan dimasukan ke root->kiri. Apabila data root yang akan dimasukan lebih besar dari root, maka akan dimasukan ke root-> kanan. Begitu terus sampai akan terbentuk suatu tree yang terstruktur.

Pencarian data menggunakan transvers inorder, apabila data di temukan akan mengeluarkan bool value true, dan apabila data tidak ditemukan akan mengeluarkan bool value false

2. Percobaan 2

a) Tampilan Coding

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

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

//membuat Node Baru
BstNode* GetNewNode(int data) {
    BstNode* newNode = new BstNode();
    newNode->data = data;
    newNode->left = newNode->right = NULL;
    return newNode;
}

//Memasukan data ke BST
BstNode* Insert(BstNode* root, int data) {
    if (root == NULL) {
        root = GetNewNode(data);
    }
    //jika data yang akan disisipkan lebih sedikit, sisipkan di subpohon kiri
    else if (data <= root->data) {
        root->left = Insert(root->left, data);
    }
    //lainnya, masukkan di subpohon kanan
    else {
        root->right = Insert(root->right, data);
    }
    return root;
}
```

```
33
34  //pencarian element di BST
35  bool search(BstNode* root, int data) {
36      if (root == NULL) {
37          return false;
38      }
39      else if (root->data == data) {
40          return true;
41      }
42      else if (data <= root->data) {
43          return search(root->left, data);
44      }
45      else {
46          return search(root->right, data);
47      }
48  }
49
50  //Tranverse
51  void printInorder(BstNode* root) {
52      if (root == NULL)
53          return;
54      printInorder(root->left);
55      cout << root->data << " ";
56      printInorder(root->right);
57  }
58
59  int first(BstNode* root) {
60      BstNode* current = root;
61      while (current->left != NULL) {
62          current = current->left;
63      }
64      return (current->data);
65  }
66
67  int last(BstNode* root) {
68      BstNode* current = root;
69      while (current->right != NULL) {
70          current = current->right;
71      }
72      return (current->data);
73  }
```

```
74
75  int main() {
76      cout << "NAMA: WIJAYANTO AGUNG WIBOWO\n";
77      cout << "NIM: 22.11.4552\n\n";
78      cout << "\t== BINARY SEARCH TREE==\n\n";
79      BstNode* root = NULL;
80      int menu, no;
81      while (true) {
82          system("cls");
83          cout << "Menu:\n";
84          cout << "1. Input data\n";
85          cout << "2. Tampilkan Nilai Terkecil\n";
86          cout << "3. Tampilkan Nilai Terbesar\n";
87          cout << "4. Tampilkan Urutan Data Secara Inorder\n";
88          cout << "5. Pencarian Data \n";
89          cout << "6. Exit \n";
90          cout << " Inputkan Menu: "; cin >> menu;
91          switch (menu) {
92          case 1:
93              cout << "Masukan data ke node: "; cin >> no;
94              root = Insert(root, no);
95              break;
96          case 2:
97              cout << "Nilai minimum adalah : " << first(root) << endl;
98              system("pause");
99              break;
100         case 3:
101             cout << "Nilai Maksimalnya adalah : " << last(root) << endl;
102             system("pause");
103             break;
104         case 4:
105             printf("Urutan Data Tree secara Inorder");
106             printf("\n================================\n");
107             printInorder(root);
108             system("pause");
109             break;
110         case 5:
111             int number;
112             printf("Pencarian Elemen Data");
113             printf("\n================================\n");
114             printf("Masukan Elemen Data Yang Dicari : ");
115             cin >> number;
116             if (search(root, number) == true) {
117                 cout << "Data Ditemukan\n";
118                 system("pause");
119             }
120             else {
121                 cout << "Data tidak ditemukan\n";
122                 system("pause");
123             }
124             break;
125         case 6:
126             exit;
```

b) Hasil Running

```
Menu:
1. Input data
2. Tampilkan Nilai Terkecil
3. Tampilkan Nilai Terbesar
4. Tampilkan Urutan Data Secara Inorder
5. Pencarian Data
6. Exit
 Inputkan Menu: 2
Nilai minimum adalah : 1
Press any key to continue . . .
```

```
Menu:
1. Input data
2. Tampilkan Nilai Terkecil
3. Tampilkan Nilai Terbesar
4. Tampilkan Urutan Data Secara Inorder
5. Pencarian Data
6. Exit
 Inputkan Menu: 3
Nilai Maksimalnya adalah : 56
Press any key to continue . . .
```

```
D:\kuliah\SEMESTER 2\Struktur Data -laptop\CONSOLE\Project1\x64\Debug\Project1.exe
Menu:
1. Input data
2. Tampilkan Nilai Terkecil
3. Tampilkan Nilai Terbesar
4. Tampilkan Urutan Data Secara Inorder
5. Pencarian Data
6. Exit
 Inputkan Menu: 4
Urutan Data Tree secara Inorder
================================
1 8 21 23 45 45 56 Press any key to continue . . .
```

```
Menu:
1. Input data
2. Tampilkan Nilai Terkecil
3. Tampilkan Nilai Terbesar
4. Tampilkan Urutan Data Secara Inorder
5. Pencarian Data
6. Exit
 Inputkan Menu:
5
Pencarian Elemen Data
================================
Masukan Elemen Data Yang Dicari : 23
Data Ditemukan
Press any key to continue . . .
```

c) Penjelasan

Untuk melakukan pengoperasian pencarian nilai terkecil digunakan fungsi root->left != NULL. Sampai node terakhir. Karena tree nya sudah terurut, maka akan otomatis ditemukan nilai dari terkecilnya di node paling kiri.

Untuk melakukan pengoperasian pencarian nilai terbesar digunakan fungsi root->right != NULL. Sampai node terakhir. Karena tree nya sudah terurut, maka akan otomatis ditemukan nilai dari terbesar di node paling kanan.

Pengurutan dari min ke max, akan bisa dicapai dengan Metode transverse printInorder. Dimana root->left dulu lalu root, lalu root->right.

C. Kesimpulan

Setelah melakukan percobaan pada praktikum saya dapat memahami, pengaplikasian BST akan dapat mempermudah pengolahan data. Dimana apabila sudah terarah tree nya, maka akan bisa digunakan untuk mencari nilai terkecil, nilai terbesar dan urutan nilai ascend atau descend dengan mudah.