# LAPORAN PRAKTIKUM

# “Pertemuan ke-9:Post Test - Pohon”

Diajukan untuk memenuhi salah satu praktikum Mata Struktur Data Informatika yang di ampu oleh:

Dr., Ardiansyah, S.T., M.Cs.


Disusun Oleh:

Mohammad Farid Hendianto 2200018401

A /Selasa 10.30 – 13.30 Lab. Komputasi Dasar

## PROGRAM STUDI INFORMATIKA

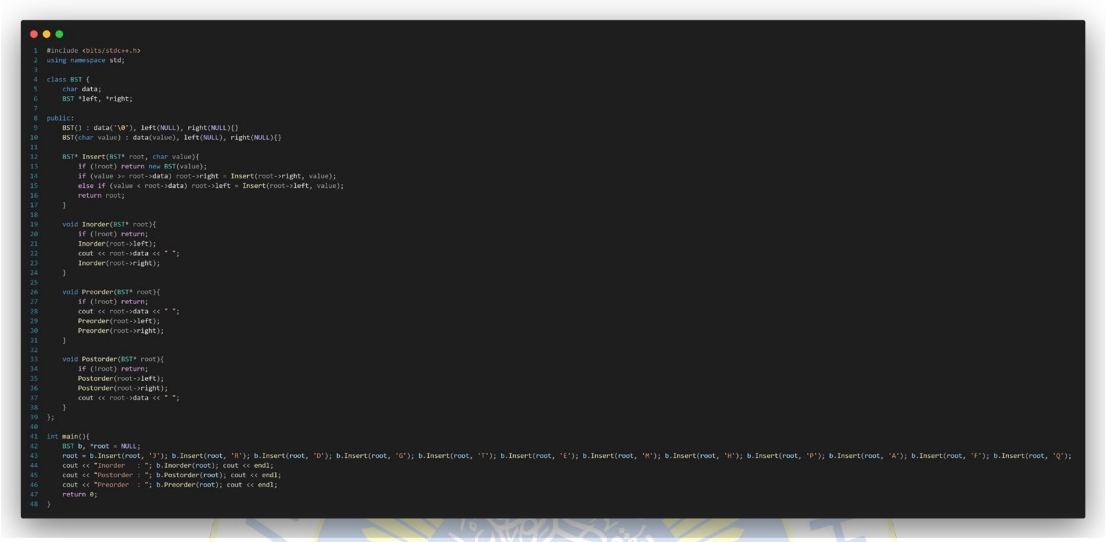
## UNIVERSITAS AHMAD DAHLAN

## FAKULTAS TEKNOLOGI INDUSTRI

## TAHUN 2023

Dari hasil pretes di atas, buatlah main function (lihat contoh main function di atas). Cocokkan hasil penelusuran Inorder, Postorder, dan Preordernya dari pretes anda.

Source Kode:



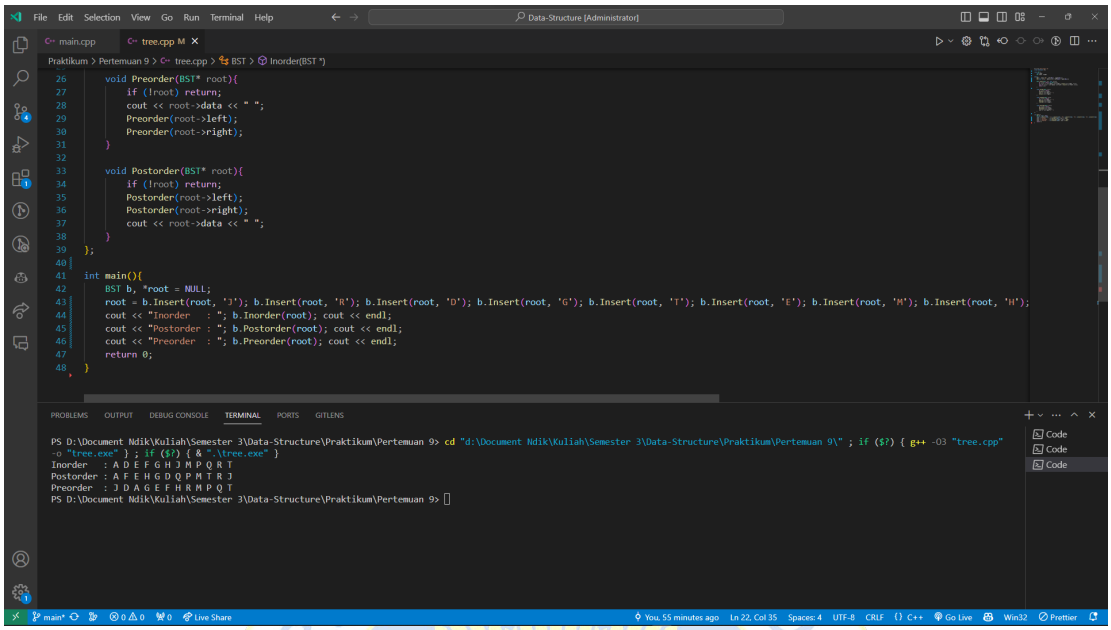
```cpp
#include <bits/stdc++.h>
using namespace std;

class BST {
    char data;
    BST *left, *right;

public:
    BST() : data('\0'), left(NULL), right(NULL){}
    BST(char value) : data(value), left(NULL), right(NULL){}

    BST* Insert(BST* root, char value){
        if (!root) return new BST(value);
        if (value >= root->data) root->right = Insert(root->right, value);
        else if (value < root->data) root->left = Insert(root->left, value);
        return root;
    }

    void Inorder(BST* root){
        if (!root) return;
        Inorder(root->left);
        cout << root->data << " ";
        Inorder(root->right);
    }

    void Preorder(BST* root){
        if (!root) return;
        cout << root->data << " ";
        Preorder(root->left);
        Preorder(root->right);
    }

    void Postorder(BST* root){
        if (!root) return;
        Postorder(root->left);
        Postorder(root->right);
        cout << root->data << " ";
    }
};

int main(){
    BST b, *root = NULL;
    root = b.Insert(root, 'J'); b.Insert(root, 'R'); b.Insert(root, 'D'); b.Insert(root, 'G'); b.Insert(root, 'T'); b.Insert(root, 'E'); b.Insert(root, 'M'); b.Insert(root, 'H'); b.Insert(root, 'P'); b.Insert(root, 'A'); b.Insert(root, 'F'); b.Insert(root, 'Q');
    cout << "Inorder   : "; b.Inorder(root); cout << endl;
    cout << "Postorder : "; b.Postorder(root); cout << endl;
    cout << "Preorder  : "; b.Preorder(root); cout << endl;
    return 0;
}
```

Source code

Fungsi main dan Program jalan (Screenshot input output) (Skor 40)

```cpp
int main(){
    BST b, *root = NULL;
    root = b.Insert(root, 'J'); b.Insert(root, 'R'); b.Insert(root, 'D'); b.Insert(root, 'G'); b.Insert(root, 'T'); b.Insert(root, 'E'); b.Insert(root, 'M'); b.Insert(root, 'H'); b.Insert(root, 'P'); b.Insert(root, 'A'); b.Insert(root, 'F'); b.Insert(root, 'Q');
    cout << "Inorder   : "; b.Inorder(root); cout << endl;
    cout << "Postorder : "; b.Postorder(root); cout << endl;
    cout << "Preorder  : "; b.Preorder(root); cout << endl;
    return 0;
}
```

Int main




Output program jalan

Fungsi dan output inorder (Skor 15)

```
void Inorder(BST* root){
    if (!root) return;
    Inorder(root->left);
    cout << root->data << " ";
    Inorder(root->right);
}
```

Fungsi inorder

OUTPUT

Inorder : A D E F G H J M P Q R T

Fungsi dan output postorder (Skor 15)

```
33  void Postorder(BST* root){
34      if (!root) return;
35      Postorder(root->left);
36      Postorder(root->right);
37      cout << root->data << " ";
38  }
```

Fungsi postorder

OUTPUT

Postorder : A F E H G D Q P M T R J

Fungsi dan output preorder (Skor 15)

```
26  void Preorder(BST* root){
27      if (!root) return;
28      cout << root->data << " ";
29      Preorder(root->left);
30      Preorder(root->right);
31  }
```

OUTPUT

Preorder : J D A G E F H R M P Q T

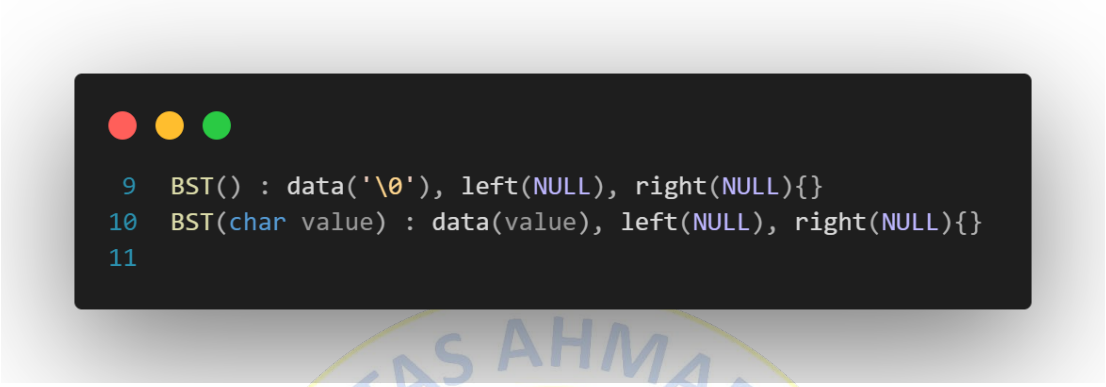
Class

```
3
4   class BST {
5       char data;
6       BST *left, *right;
7
8   public:
9       BST() : data('\0'), left(NULL), right(NULL){}
10      BST(char value) : data(value), left(NULL), right(NULL){}
11
```

Penggunaan class BST (Binary Search Tree)

Penjelasan program:

Kode ini mendefinisikan struktur data pohon biner pencarian (BST/ binary search tree) dan beberapa operasi yang dapat dilakukan pada BST.



```
 9   BST() : data('\0'), left(NULL), right(NULL){}
10   BST(char value) : data(value), left(NULL), right(NULL){}
11
```

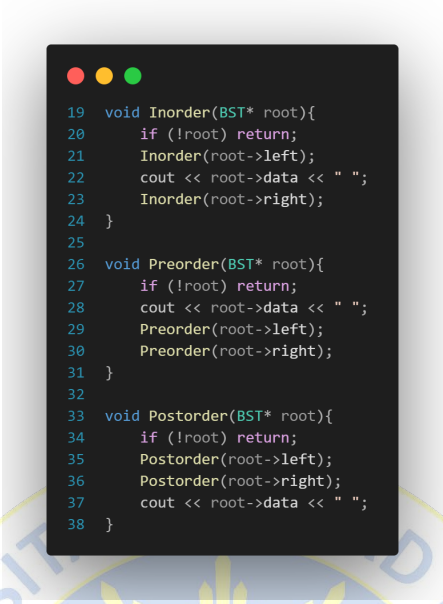
constructor

`BST() : data('\0'), left(NULL), right(NULL){}` dan `BST(char value) : data(value), left(NULL), right(NULL){}` adalah konstruktor untuk kelas BST. Konstruktor pertama membuat BST kosong, sedangkan konstruktor kedua membuat BST dengan nilai awal.

```
12  BST* Insert(BST* root, char value){
13      if (!root) return new BST(value);
14      if (value >= root->data) root->right = Insert(root->right, value);
15      else if (value < root->data) root->left = Insert(root->left, value);
16      return root;
17  }
```

Fungsi insert

`BST* Insert(BST* root, char value)` adalah fungsi untuk memasukkan elemen baru ke dalam BST. Jika BST kosong, fungsi ini akan membuat BST baru. Jika tidak, fungsi ini akan memasukkan elemen baru ke sub-pohon kiri jika elemen tersebut lebih kecil dari root, atau ke sub-pohon kanan jika elemen tersebut lebih besar atau sama dengan root.



```
19  void Inorder(BST* root){
20      if (!root) return;
21      Inorder(root->left);
22      cout << root->data << " ";
23      Inorder(root->right);
24  }
25
26  void Preorder(BST* root){
27      if (!root) return;
28      cout << root->data << " ";
29      Preorder(root->left);
30      Preorder(root->right);
31  }
32
33  void Postorder(BST* root){
34      if (!root) return;
35      Postorder(root->left);
36      Postorder(root->right);
37      cout << root->data << " ";
38  }
```

Fungsi Inorder,Preorder, dan Postorder

`void Inorder(BST* root)`, `void Preorder(BST* root)`, dan `void Postorder(BST* root)` adalah fungsi untuk melakukan penjelajahan pohon (tree traversal) dalam urutan inorder, preorder, dan postorder.

- Dalam penjelajahan inorder, fungsi ini akan mengunjungi sub-pohon kiri, root, dan sub-pohon kanan.
- Dalam penjelajahan preorder, fungsi ini akan mengunjungi root, sub-pohon kiri, dan sub-pohon kanan.
- Dalam penjelajahan postorder, fungsi ini akan mengunjungi sub-pohon kiri, sub-pohon kanan, dan root.

```
41  int main(){
42      BST b, *root = NULL;
43      root = b.Insert(root, 'J'); b.Insert(root, 'H'); b.Insert(root, 'D'); b.Insert(root, 'G'); b.Insert(root, 'T'); b.Insert(root, 'E'); b.Insert(root, 'M'); b.Insert(root, 'N'); b.Insert(root, 'P'); b.Insert(root, 'A'); b.Insert(root, 'F'); b.Insert(root, 'Q');
44      cout << "Inorder   : "; b.Inorder(root); cout << endl;
45      cout << "Postorder : "; b.Postorder(root); cout << endl;
46      cout << "Preorder  : "; b.Preorder(root); cout << endl;
47      return 0;
48  }
```

Fungsi utama

`int main()` adalah fungsi utama yang membuat BST dan melakukan beberapa operasi pada BST tersebut, termasuk memasukkan beberapa elemen dan melakukan penjelajahan pohon.