

**Laporan Praktikum**

**Algoritma dan Struktur Data**

Ganjil 2023/2024

Program Studi Teknik Informatika

Institut Teknologi Sumatera

|  |  |
| --- | --- |
| **Modul :** | **Binary Tree** |
| **Nama :** | **Rayhan Fadel Irwanto** |
| **NIM :** | **122140236** |
| **Kelas (Kelas Asal) :** | **ASD RA** |

Instruksi sederhana :

* Disarankan kepada **Praktikan Algoritma Struktur Data** untuk mengeditnya menggunakan Google Docs agar tidak berantakan dan rapi,
* Silahkan mengganti **Nama Modul** baik yang ada pada **Cover** dan **Header** sesuai dengan materi praktikum,
* Gunakan text styling seperti **Heading 1**, **Normal Text** yang telah terformat / Text Style lainnya yang digunakan untuk menjaga estetika laporan,
* Gunakan [**Syntax Highlighter**](https://highlight.hohli.com/index.php)untuk merapikan kode yang sudah Praktikan buat ke dalam Laporan Praktikum.

**Materi Praktikum**

Binary Tree

**Link repl.it Source Code**

<https://replit.com/@RayhanFadel/tree-LM5#main.cpp>

**Source Code**

Gunakan [**Syntax Highlighter**](https://highlight.hohli.com/index.php)untuk merapikan Source Code yang dipindahkan dari text editor anda ke dokumen ini.

1. #include <iostream>
2. using namespace std;
4. struct Tree {
5. **int info;**
7. Tree \*left;
8. Tree \*right;
9. Tree \*parent;
10. **};**
12. Tree \*Initialize(int value) {
13. Tree \*newNode = new Tree;
15. **newNode->info = value;**
16. newNode->left = nullptr;
17. newNode->right = nullptr;
18. newNode->parent = nullptr;
20. **return newNode;**
21. }
23. bool IsTreeEmpty(Tree \*P) { return P == nullptr; }
25. **void InsertLeft(Tree \*P, int value) {**
26. Tree \*newNode = Initialize(value);
28. if (P->left == nullptr) {
29. newNode->parent = P;
30. **P->left = newNode;**
31. cout << "Success insert left node with value " << value << " with "
32. << P->info << " as parrent**\n**";
33. } else {
34. cout << "Tree left was declared!";
35. **}**
36. }
38. void InsertRight(Tree \*P, int value) {
39. Tree \*newNode = Initialize(value);
40. **if (P->right == nullptr) {**
41. newNode->parent = P;
42. P->right = newNode;
43. cout << "Success insert right node with value " << value << " with "
44. << P->info << " as parrent**\n**";
45. **} else {**
46. cout << "Tree right was declared!";
47. }
48. }
50. **void PreOrder(Tree \*P) {**
51. if (IsTreeEmpty(P)) {
53. } else {
54. cout << P->info << " ";
55. **PreOrder(P->left);**
56. PreOrder(P->right);
57. }
58. }
60. **void InOrder(Tree \*P) {**
61. if (IsTreeEmpty(P)) {
63. } else {
64. InOrder(P->left);
65. **cout << P->info << " ";**
66. InOrder(P->right);
67. }
68. }
70. **void display(Tree \*P, int indent = 0) {**
71. cout << "<>Pre Order : ";
72. PreOrder(P);
73. cout << endl;
74. cout << "<>In Order : ";
75. **InOrder(P);**
76. cout << endl;
77. **}**
79. int main() {
80. Tree \*P = Initialize(1);
81. InsertLeft(P, 2);
82. **InsertRight(P, 3);**
84. InsertLeft(P->left, 4);
85. InsertRight(P->left, 5);
87. **InsertLeft(P->right, 6);**
88. InsertRight(P->right, 7);
90. InsertLeft(P->left->left, 8);
92. **cout << endl << "Display tree : " << endl;**
93. display(P);
95. return 0;
96. }

**Dokumentasi Hasil Running**

|  |
| --- |
|  |
| **Gambar 1.** Output Code Binary Tree Latihan Minggu 5.  Program di atas merupakan implementasi pohon biner dalam C++. Dalam struktur data `Tree`, setiap node memiliki informasi (info), pointer ke anak kiri (left), anak kanan (right), dan induk (parent). Fungsi `Initialize` digunakan untuk membuat dan menginisialisasi node baru dengan nilai tertentu. Fungsi `IsTreeEmpty` mengembalikan true jika pohon kosong. `InsertLeft` dan `InsertRight` digunakan untuk menyisipkan node baru di sebelah kiri atau kanan dari node P. Fungsi `PreOrder` dan `InOrder` melakukan penelusuran pohon secara pre-order dan in-order, masing-masing. Contoh penggunaan pohon biner ditunjukkan dalam fungsi `main`, di mana node-node baru disisipkan dengan nilai tertentu, dan hasilnya ditampilkan dengan pre-order dan in-order traversal. |

|  |
| --- |
|  |
|  |