**STRUKTUR DATA NON LINEAR**

**“BINARY SEARCH TREE”**

**MODUL KE – 2 : CREATE, INSERT, SEARCH**



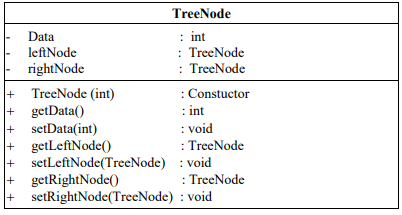
DISUSUN OLEH :

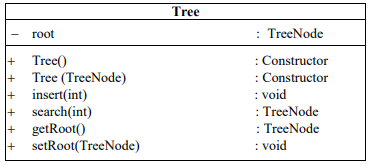
Nama : Andreas Nathanael Priambodo

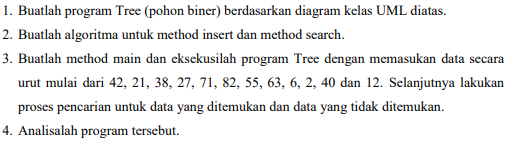
NIM : 215314043

**TEKNIK INFORMATIKA FAKULTAS SAINS DAN TEKNOLOGI UNIVERSITAS SANATA DHARMA YOGYAKARTA 2023**

1. Soal







1. Algoritma
2. Program

* MainTree

|  |
| --- |
| package binary\_search\_tree;  import java.util.Scanner;  public class MainTree {  public static void main(String[] args) {  Scanner dtSc = new Scanner(System.in);  Tree Ob = new Tree();  Ob.Insert(42);  Ob.Insert(21);  Ob.Insert(38);  Ob.Insert(27);  Ob.Insert(71);  Ob.Insert(82);  Ob.Insert(55);  Ob.Insert(63);  Ob.Insert(6);  Ob.Insert(2);  Ob.Insert(40);  Ob.Insert(12);  System.out.println();  System.out.print("Search Data : ");  int search = dtSc.nextInt();  TreeNode dataSearch = Ob.Search(search);  if (dataSearch == null) {  System.out.println("Data not Found");  } else {  System.out.println("Data " + dataSearch.getData() + " Found");  }  }  } |

* Tree

|  |
| --- |
| package binary\_search\_tree;  public class Tree {  private TreeNode root;  public Tree() {  root = null;  }  public Tree(TreeNode root) {  this.root = root;  }  public TreeNode getRoot() {  return root;  }  public void setRoot(TreeNode root) {  this.root = root;  }  public void Insert(int in) {  TreeNode tr = new TreeNode(in);  if (root == null) {  root = tr;  System.out.println("Root " + root.getData());  } else {  TreeNode trB = root;  while (true) {  if (in <= trB.getData()) {  if (trB.getLeftNode() == null) {  trB.setLeftNode(tr);  System.out.println(tr.getData() + " LEFT " + trB.getData());  break;  } else {  trB = trB.getLeftNode();  }  } else {  if (trB.getRightNode() == null) {  trB.setRightNode(tr);  System.out.println(tr.getData() + " RIGHT " + trB.getData());  break;  } else {  trB = trB.getRightNode();  }  }  }  }  }  public TreeNode Search(int cari) {  TreeNode data = root;  while (data != null) {  if (cari == data.getData()) {  return data;  } else if (cari < data.getData()) {  data = data.getLeftNode();  } else {  data = data.getRightNode();  }  }  return null;  }  } |

* TreeNode

|  |
| --- |
| package binary\_search\_tree;  public class TreeNode {  private int data;  private TreeNode leftNode;  private TreeNode rightNode;  public TreeNode(int data) {  this.data = data;  leftNode = rightNode = null;  }  public int getData() {  return data;  }  public void setData(int data) {  this.data = data;  }  public TreeNode getLeftNode() {  return leftNode;  }  public void setLeftNode(TreeNode leftNode) {  this.leftNode = leftNode;  }  public TreeNode getRightNode() {  return rightNode;  }  public void setRightNode(TreeNode rightNode) {  this.rightNode = rightNode;  }  } |

1. Output
2. Analisa