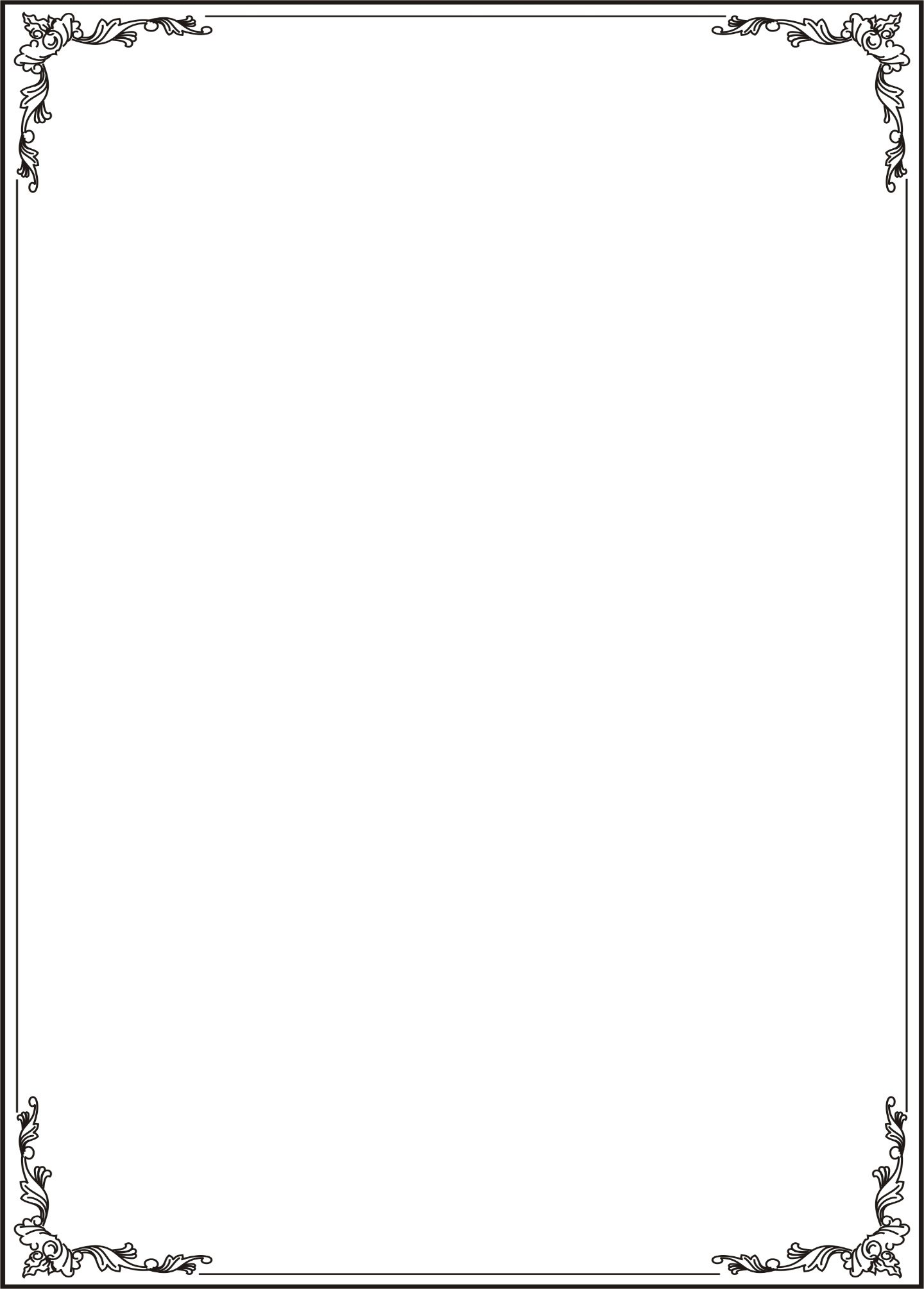
****

**HCMC UNIVERSITY OF TECHNOLOGY AND EDUCATION**

***Faculty For High Quality Training***

-----🙞🙜🕮🙞🙜-----

**FINAL PROJECT REPORT**

**SIMULATION OF BINARY SEARCH TREE STRUCTURE PROGRAM**

**LECTURER: Mr. Le Vinh Thinh**

**STUDENT INVOLVEMENT:**

**Thiều Quang Khánh 18110022**

**Đinh Văn Trường 18110060**

**Ho Chi Minh City**

SCORE

|  |  |  |  |
| --- | --- | --- | --- |
| CRITERIA | CONTENT | PRESENTATION | TOTAL |
| POINT |  |  |  |

LECTURER’S COMMENT

**EVALUATION:**

* Thiều Quang Khánh:
* Đinh Văn Trường:

LECTURER

Contents

[**Chapter 1: Program Overview** 3](#_Toc43416675)

[**1.** **General Introduction** 3](#_Toc43416676)

[**1.1.** **Simulation of binary search tree structure software** 4](#_Toc43416677)

[**1.1.1.** **Project request** 4](#_Toc43416678)

[**1.1.2.** **Project analysis** 4](#_Toc43416679)

[**2.** **Simulation of binary search tree structure software specification** 4](#_Toc43416680)

[**2.1.** **Simulation of binary search tree structure** 4](#_Toc43416681)

[**2.1.1.** **Simulation of binary search tree structure introduction** 4](#_Toc43416682)

[**2.1.2.** **Use case diagram** 4](#_Toc43416683)

[**2.1.3.** **Data Input - Output** 5](#_Toc43416684)

[**2.1.4.** **Main feature** 5](#_Toc43416685)

[**2.1.5.** **Expected interface** 5](#_Toc43416686)

[**Chapter 2: Implementation plan** 6](#_Toc43416687)

[**1.** **Plan** 6](#_Toc43416688)

[**2.** **Assignment of work** 7](#_Toc43416689)

[**Chapter 3: Design Software** 8](#_Toc43416690)

[**1.** **Interface** 8](#_Toc43416691)

[**2.** **Algorithm** 9](#_Toc43416692)

[**3.** **Diagram** 17](#_Toc43416693)

[**Chapter 4: Testing Software** 18](#_Toc43416694)

[**Chapter 5: Group’s conclusion** 30](#_Toc43416695)

[**1.** **Conclusion** 30](#_Toc43416696)

[**2.** **Development** 30](#_Toc43416697)

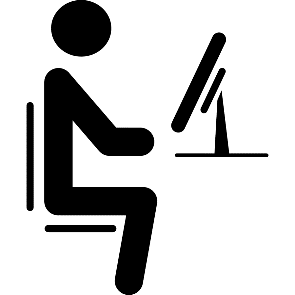
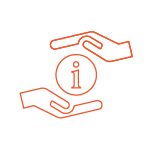
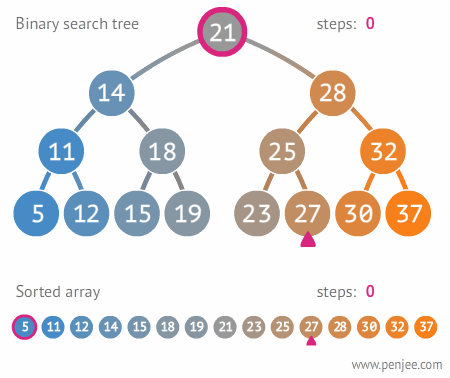
# **Chapter 1: Program Overview**

1. **General Introduction**
   1. **Simulation of binary search tree structure software**
      1. **Project request**

* Design and build object-oriented software to solve searching note on binary tree fast and right answer.
  + 1. **Project analysis**
* Building object-oriented software.
* Integer analysis to get input data
* Store and display as image when the user requests.
  + 1. **Direction of implementation**
* Building object-oriented software with C #.

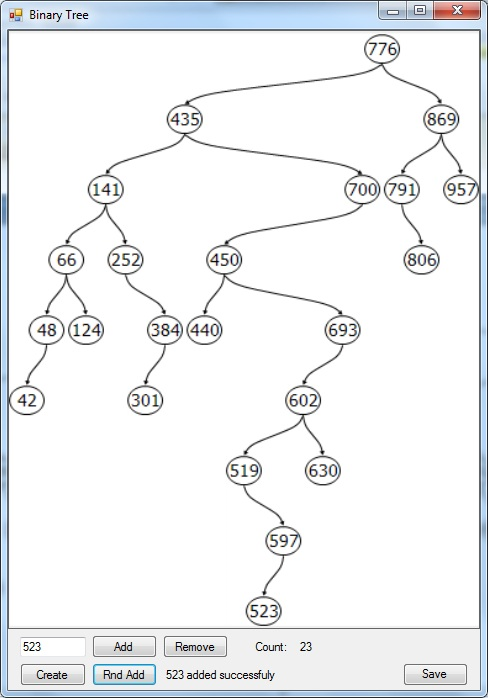
1. **Simulation of binary search tree structure software specification** 
   1. **Simulation of binary search tree structure**
      1. **Simulation of binary search tree structure introduction**

* A binary search tree (BST) is a binary tree where each node has a comparable key (and an associated value) and satisfies the restriction that the key in any node is larger than the keys in all nodes in that node's left subtree and smaller than the keys in all nodes in that node's right subtree.
  + 1. **Use case diagram**

User Simulation of binary search tree structure Image of tree

* + 1. **Data Input - Output**
* **Input:** Number entered from keyboard
* **Output:** Display the saved user number under a binary tree interface
  + 1. **Main feature**
* Write and store nodes in memory, allowing users to find, add, delete notes.
* Display detailed tree information (Node count, tree height).
* Search for node by node value.
* Delete node by node value
* Genarate binary tree with random value.
  + 1. **Expected interface**
* The interface displays in an elegant square style, using a light background color that does not cause eye strain for the user.
* Simple click and type operations.



# **Chapter 2: Implementation plan**

1. **Plan**

**Work plan**

1- Learn about Binary search tree structure, outline ideas about interfaces and functions.

2- Analyze input and software requirements. Start designing the interface.

3- Build the main interface layers and main functions for the software

* Genarate a new random tree
* Edit node (add, delete)
* Save tree as image
* Search node

4- Building extended functions for the software.

* Print value out to three ways: in-order traverse , pre-order traverse, post-order traverse

5- Building a binary search tree panel interface for the software.

6- Create the class: Binarytree.

7- Fixing errors and completing the software.

8- Write the report.

1. **Assignment of work**

|  |  |  |  |
| --- | --- | --- | --- |
|  | Name | Task description |  |
| 1 | Thiều Quang Khánh | * Divide the work for member * Create Binary tree class * Genarate random tree * Check and fix bugs * Write report | 50% |
| 2 | Đinh Văn Trường | * Create Binary tree panel interface * Write function: Save tree as image * Add, delete, search node * Print tree: in-order traverse, pre-order traverse, post-order traverse * Create powerpoint | 50% |

# **Chapter 3: Design Software**

1. **Interface**

|  |
| --- |
| Main view  This design’s program was created by Winform. |
|  |
|  |

1. **Algorithm**

|  |  |  |  |
| --- | --- | --- | --- |
| TT | Class | Purpose | Student name |
| 1 | BSTreePanel | Show the image of binary tree | Dinh  Van  Truong |
| 2 | Form1 | Is the main form | Dinh  Van  Truong,  Thieu Quang Khanh |
| 3 | Organizational Unit | Color control | Dinh  Van  Truong |
| 4 | BinaryTree | Add, search, clear, remove Binary tree node and print Binary tree | Thieu Quang Khanh |
| 5 | BinaryTreeNode | Declare structure of Node includes: left child, right child, parent, value,….. | Thieu Quang Khanh |

B.Methods in class

B.1. Methods in class form1

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| TT | Method | Purpose | File name,line cotain | Student name |
| 1 | void btnGenerateTree\_Click(object sender, EventArgs e) | Method is used to genarate random binary tree and show in on panel when click on this button | Form1.cs/line 31 | Thieu  Quang Khanh |
| 2 | void btnSaveImage\_Click(object sender, EventArgs e) | Method is used to save binary tree as am image when click on this button | Form1.cs/line 39 | Dinh Van Truong |
| 3 | void btnSearch\_Click(object sender, EventArgs e) | Method is used to search node by value when click on this button | Form1.cs/line 78 | Dinh Van Truong |
| 4 | void btnDelete\_Click(object sender, EventArgs e) | Method is used to delete node by value when click on this button | Form1.cs/line 88 | Dinh Van Truong |
| 5 | void btnAdd\_Click(object sender, EventArgs e) | Method is used to add node by value when click on this button | Form1.cs/line 97 | Dinh Van Truong |
| 6 | void UpdateInfo() | Method is used to update node count and tree height | Form1.cs/line 106 | Dinh Van Truong |
| 7 | void btnInOrderTraverse\_Click(object sender, EventArgs e) | Method is used to print binary tree to in-order traverse when click on this button | Form1.cs/line 112 | Dinh Van Truong |
| 8 | void btnPreOrderTraverse\_Click(object sender, EventArgs e) | Method is used to print binary tree to pre-order traverse when click on this button | Form1.cs/line 117 | Dinh Van Truong |
| 9 | void btnPostOrderTraverse\_Click(object sender, EventArgs e) | Method is used to print binary tree to post-order traverse when click on this button | Form1.cs/line 122 | Dinh Van Truong |

B.2.Methods in BSTreePanel

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| TT | Method | Purpose | File name,line cotain | Student name |
| 1 | void GenerateTree(int size, int min, int max) | Method is used to genarate random binary tree | BSTreePanel.cs/line 78 | Dinh Van Truong |
| 2 | void BeginDraw() | Method is used to start drawing tree on panel | BSTreePanel.cs/line 101 | Dinh Van Truong |
| 3 | void BeginDraw(bool resetAll) | Method is used to reset or not based on resetAll parameter | BSTreePanel.cs/line 105 | Dinh Van Truong |
| 4 | void CalculateSize(Graphics g, float left, BinaryTreeNode<int> node) | Method is used to calculate size of tree | BSTreePanel.cs/line 154 | Dinh Van Truong |
| 5 | void DrawTreeNode(Graphics g, PointF p, BinaryTreeNode<int> node, bool highlight) | Method is used to start drawing tree node | BSTreePanel.cs/line 189 | Dinh Van Truong |
| 6 | bool AddNode(int value) | Method is used to add node return true or false if successful or not | BSTreePanel.cs/line 279 | Dinh Van Truong |
| 7 | bool SearchNode(int value) | Method is used to search node return true or false if successful or not | BSTreePanel.cs/line 292 | Dinh Van Truong |
| 8 | bool DeleteNode(int value) | Method is used to delete node return true or false if successful or not | BSTreePanel.cs/line 306 | Dinh Van Truong |
| 9 | void InOrderTraverse() | Method is used to print tree to in-order travers | BSTreePanel.cs/line 319 | Dinh Van Truong |
| 10 | void PreOrderTraverse() | Method is used to print tree to pre-order traverse | BSTreePanel.cs/line 330 | Dinh Van Truong |
| 11 | void PostOrderTraverse() | Method is used to print tree to post-order traverse | BSTreePanel.cs/line 341 | Dinh Van Truong |
| 12 | void txtOutput\_KeyDown(object sender, KeyEventArgs e) | Method is used to handle event when a key is first pressed | BSTreePanel.cs/line 353 | Dinh Van Truong |
| 13 | void zoomInToolStripMenuItem\_Click(object sender, EventArgs e) | Method is used to zoom in panel | BSTreePanel.cs/line 420 | Dinh Van Truong |
| 14 | void zoomOutToolStripMenuItem\_Click(object sender, EventArgs e) | Method is used to zoom out panel | BSTreePanel.cs/line 430 | Dinh Van Truong |
| 15 | void showAllToolStripMenuItem\_Click(object sender, EventArgs e) | Method is used to show all panel on menu when click this button | BSTreePanel.cs/line 440 | Dinh Van Truong |
| 16 | void contextMenuStrip1\_Opening(object sender, CancelEventArgs e) | Method is used to handle event when menu strip is openning | BSTreePanel.cs/line 446 | Dinh Van Truong |

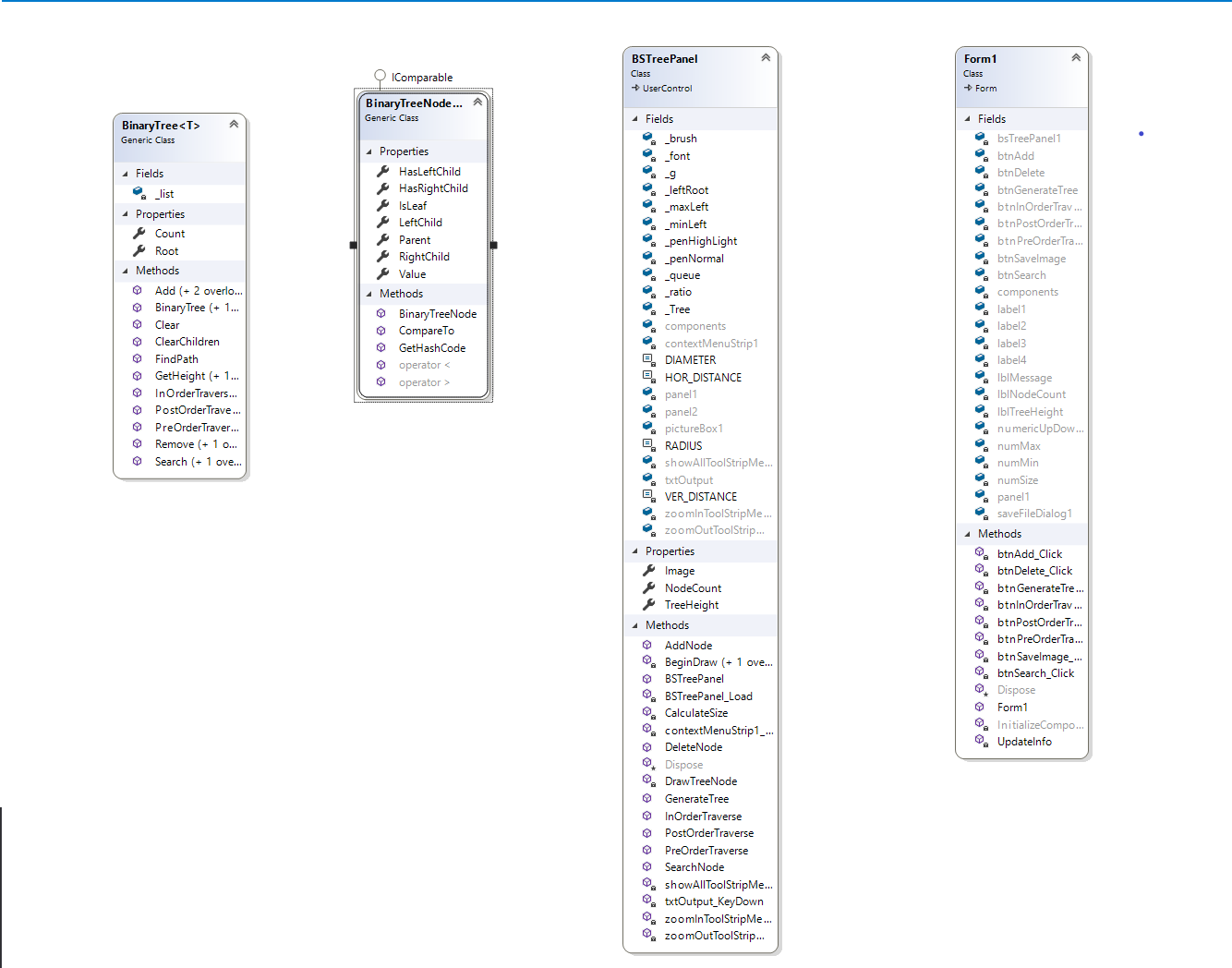
B.3.Method in BinaryTreeNode<T> : IComparable

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| TT | Method | Purpose | File name,line cotain | Student name |
| 1 | override int GetHashCode() | Method is used to get hash code of binary tree node | BinaryTree.cs/line 35 | Thieu Quang Khanh |
| 2 | BinaryTreeNode(T value) | This is constructor of binary tree node | BinaryTree.cs/line 41 | Thieu Quang Khanh |
| 3 | int CompareTo(object obj) | Method is used to compare between two nodes | BinaryTree.cs/line 48 | Thieu Quang Khanh |
| 4 | bool operator <(BinaryTreeNode<T> node1, BinaryTreeNode<T> node2) | Method is used to compare between two nodes and return true or false if node1 < node2 or not | BinaryTree.cs/line 57 | Thieu Quang Khanh |
| 5 | bool operator >(BinaryTreeNode<T> node1, BinaryTreeNode<T> node2) | Method is used to compare between two nodes and return true or false if node1 > node2 or not | BinaryTree.cs/line 61 | Thieu Quang Khanh |

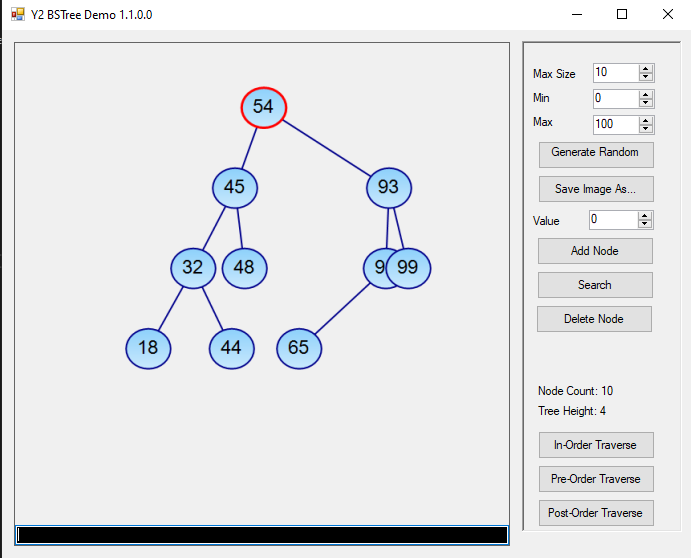
B.4.Method in BinaryTree<T>

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| TT | Method | Purpose | File name,line cotain | Student name | |
| 1 | BinaryTree() | This is constructor of binary tree | BinaryTree.cs/line 78 | | Thieu Quang Khanh |
| 2 | BinaryTree(T value) | This is constructor of binary tree with value | BinaryTree.cs/line 84 | | Thieu Quang Khanh |
| 3 | virtual List<T> InOrderTraverse() | Method is used to get list of tree in in-order traverse | BinaryTree.cs/line 91 | | Thieu Quang Khanh |
| 4 | void InOrderTraverse(BinaryTreeNode<T> node) | Method is used to add value of node in in-order traverse | BinaryTree.cs/line 97 | | Thieu Quang Khanh |
| 5 | virtual List<T> PreOrderTraverse() | Method is used to get list of tree in pre-order traverse | BinaryTree.cs/line 108 | | Thieu Quang Khanh |
| 6 | void PreOrderTraverse(BinaryTreeNode<T> node) | Method is used to add value of node in pre-order traverse | BinaryTree.cs/line 114 | | Thieu Quang Khanh |
| 7 | virtual List<T> PostOrderTraverse() | Method is used to get list of tree in post-order traverse | BinaryTree.cs/line 125 | | Thieu Quang Khanh |
| 8 | void PostOrderTraverse(BinaryTreeNode<T> node) | Method is used to add value of node in post-order traverse | BinaryTree.cs/line 131 | | Thieu Quang Khanh |
| 9 | virtual void Add(params T[] values) | Method is used to add new nodes with value | BinaryTree.cs/line 142 | | Thieu Quang Khanh |
| 10 | virtual bool Add(T value) | Method is used to add a new node and return true or false if successful or not | BinaryTree.cs/line 146 | | Thieu Quang Khanh |
| 11 | bool Add(BinaryTreeNode<T> parentNode, BinaryTreeNode<T> node) | Method is used to add new node into parent node. Return true or false if successful or not | BinaryTree.cs/line 158 | | Thieu Quang Khanh |
| 12 | virtual void ClearChildren(BinaryTreeNode<T> node) | Method is used to clear childrend of this node | BinaryTree.cs/line 190 | | Thieu Quang Khanh |
| 13 | virtual void Clear() | Method is used to clear binary tree | BinaryTree.cs/line 207 | | Thieu Quang Khanh |
| 14 | virtual int GetHeight() | Method is used to get height of tree | BinaryTree.cs/line 216 | | Thieu Quang Khanh |
| 15 | int GetHeight(BinaryTreeNode<T> startNode) | Method is used to get height of this tree from start node | BinaryTree.cs/line 220 | | Thieu Quang Khanh |
| 16 | virtual BinaryTreeNode<T> Search(T value) | Method is used to search node with value. Return this node. | BinaryTree.cs/line 227 | | Thieu Quang Khanh |
| 17 | virtual BinaryTreeNode<T> Search(BinaryTreeNode<T> node, T value) | Method is used to search node with value from node | BinaryTree.cs/line 231 | | Thieu Quang Khanh |
| 18 | virtual Queue<T> FindPath(T value) | Method is used to find path from first node to search node with value | BinaryTree.cs/line 246 | | Thieu Quang Khanh |
| 19 | virtual bool Remove(T value) | Method is used to remove node with value. Return true or false if succesful or not | BinaryTree.cs/line 278 | | Thieu Quang Khanh |
| 20 | bool Remove(BinaryTreeNode<T> node, T value) | Method is used to remove node with value from node.Return true or false if successful or not | BinaryTree.cs/line 283 | | Thieu Quang Khanh |

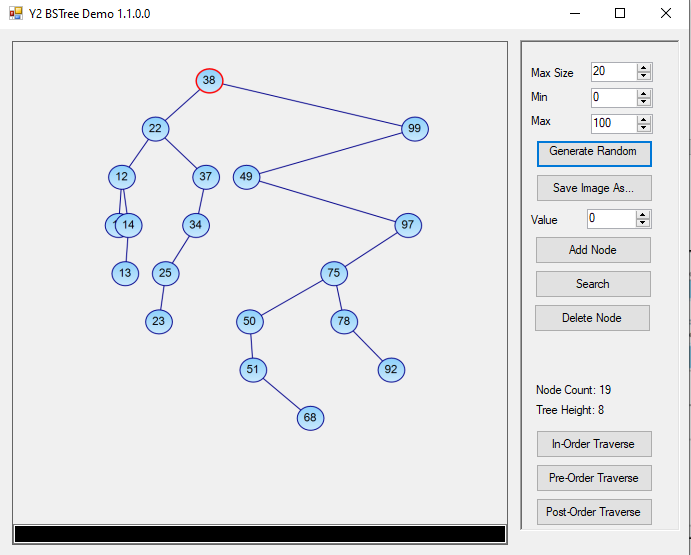
1. **Diagram**



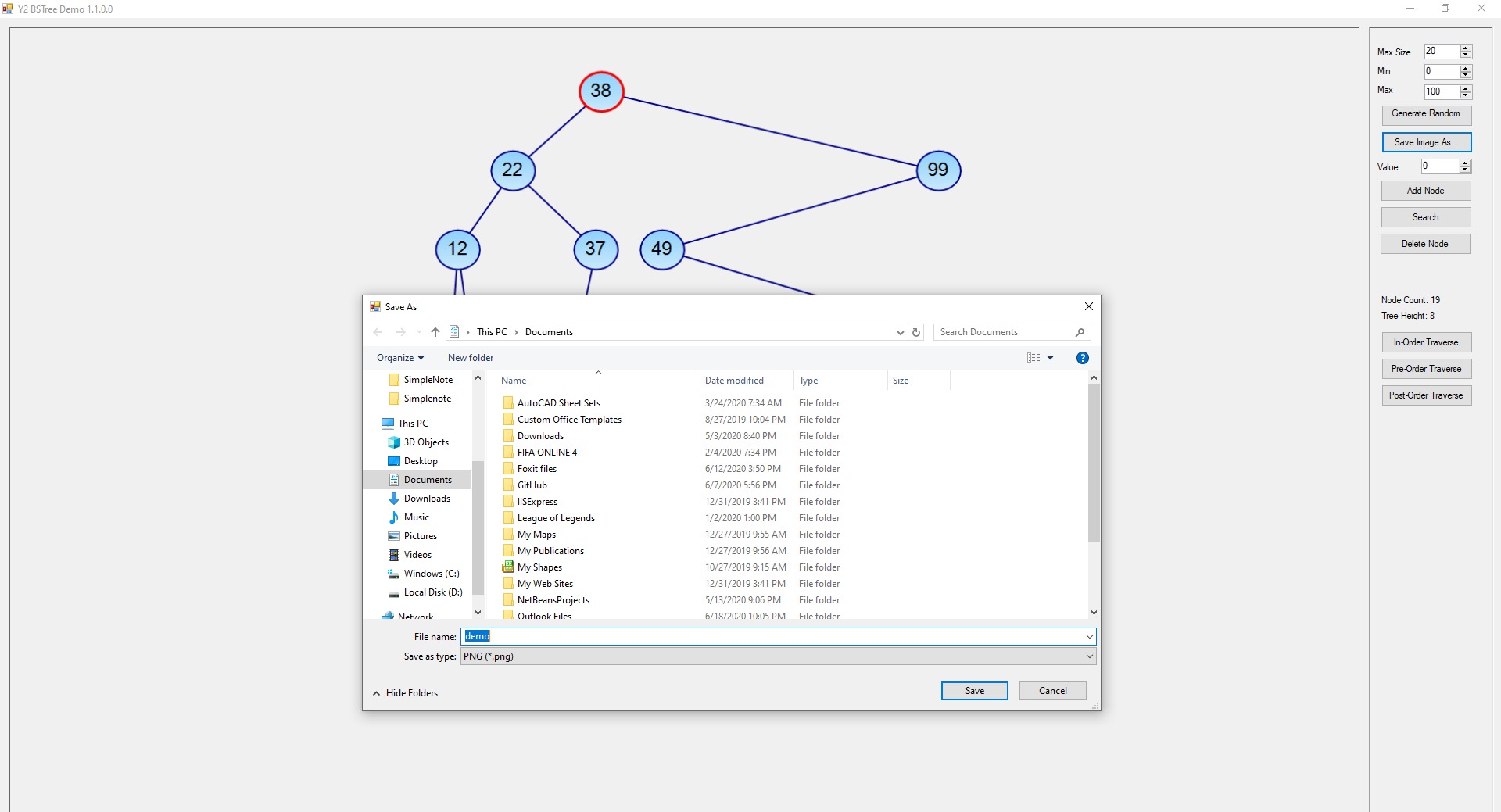
# **Chapter 4: Testing Software**



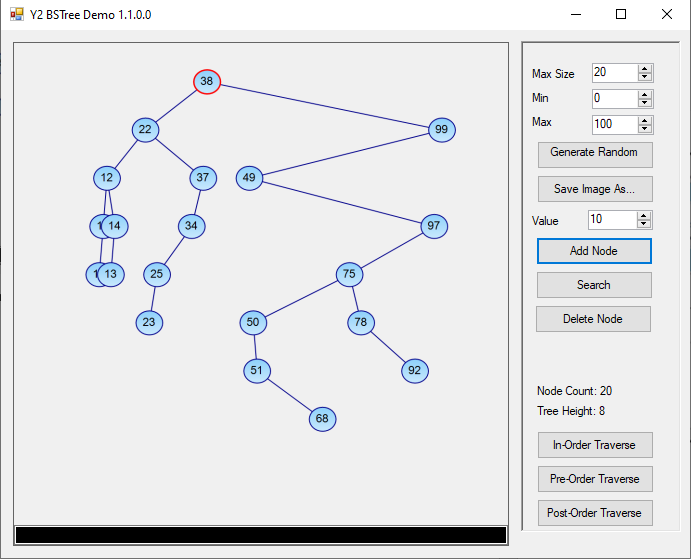
Picture 1: When first run the program



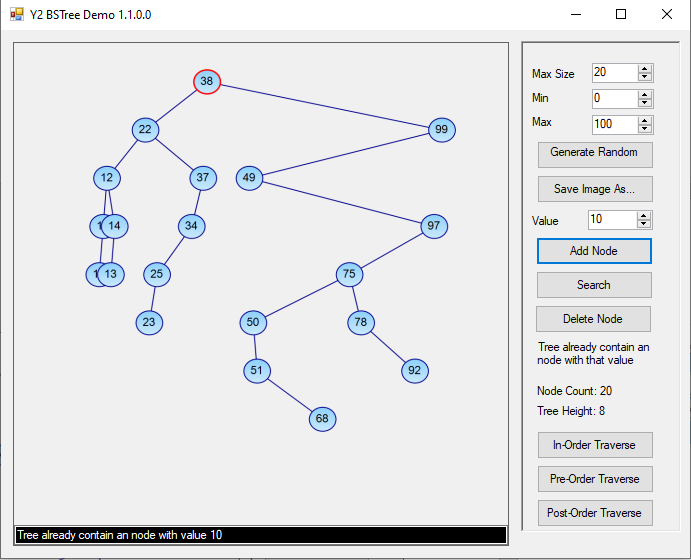
Picture2 : Generate random binary tree



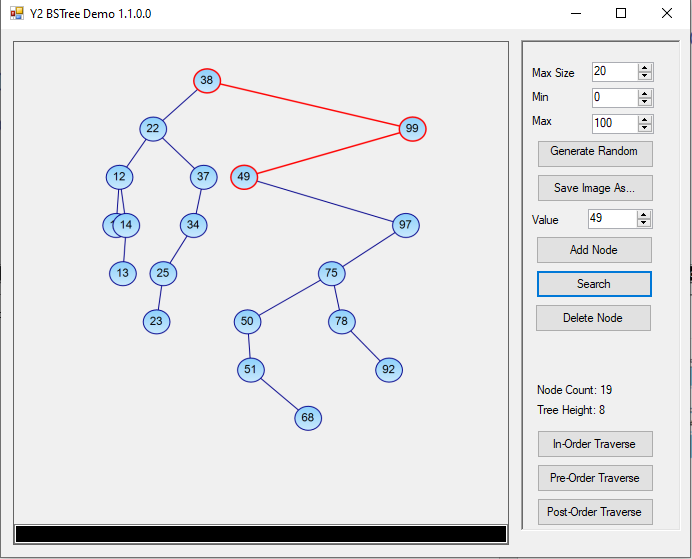
Picture 3 : Save binary tree as image



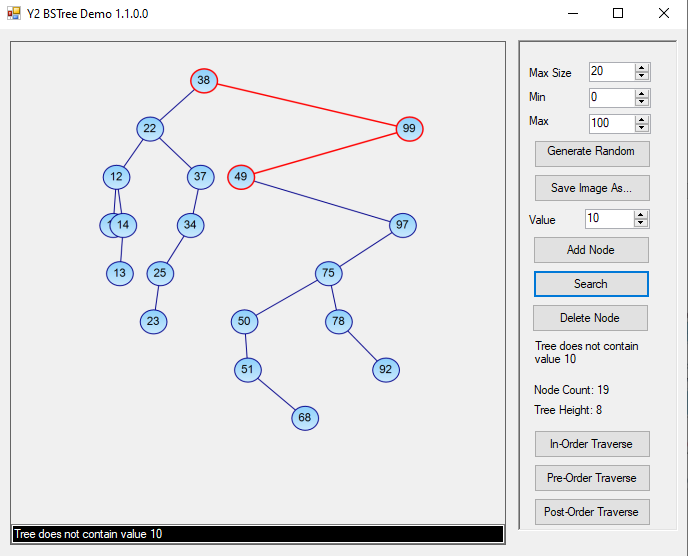
Picture4 : Test add new node with value



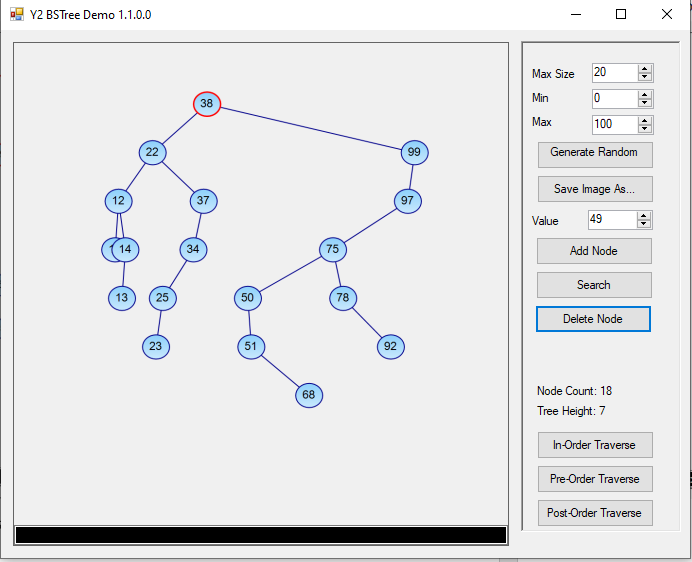
Picture 5 : Test add new node with value that has already appeared in tree



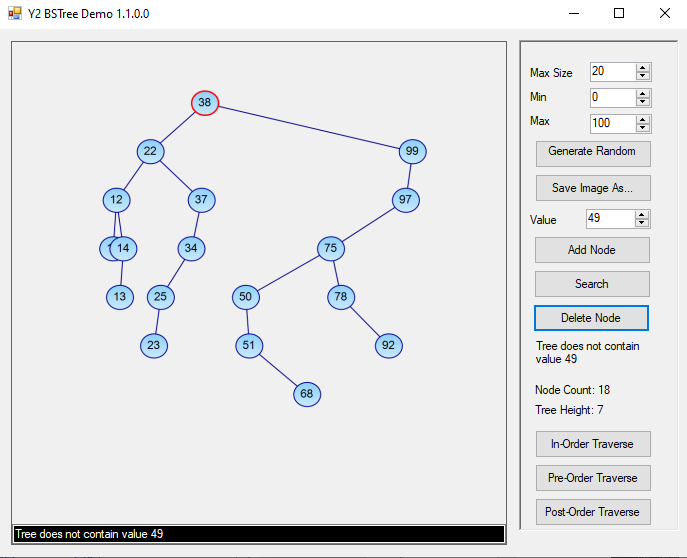
Picture 6 : Test search node with value



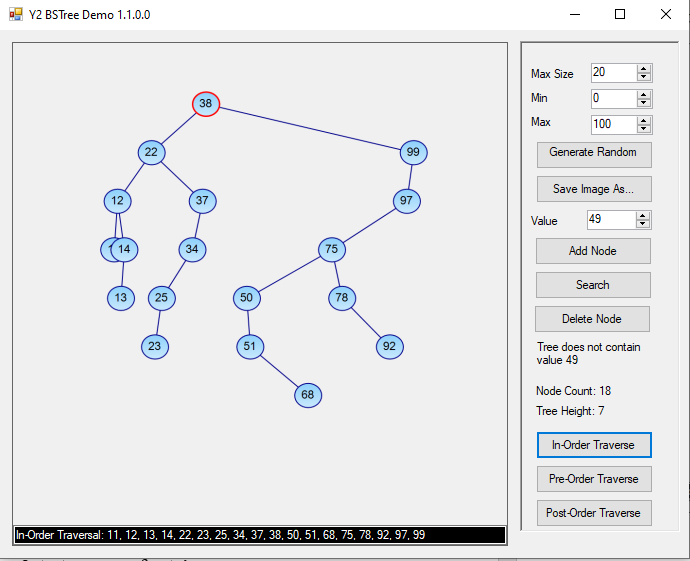
Picture 7: Test search node with value that has already appeared in tree



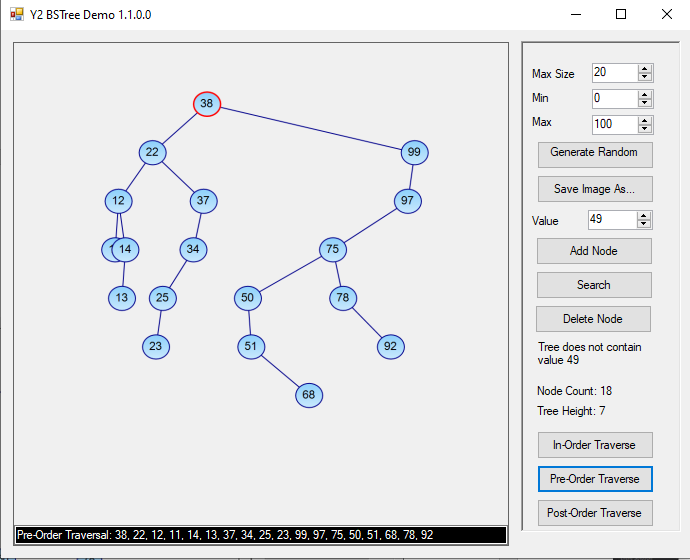
Picture 8: Test delete node with value



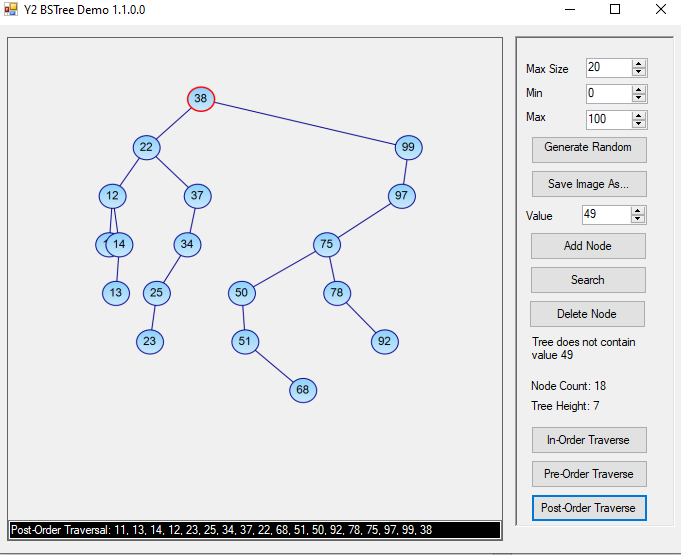
Picture 9: Test delete node with value that don’t exist in tree



Picture 10: Test print in-order traverse



Picture11 : Test print pre-order traverse



Picture 12 : Test print post -order traverse

# **Chapter 5: Group’s conclusion**

1. **Conclusion**

The team commented that the software has reached almost 100% of the requirements and features a simulation of binary search tree structure software should have.

* ***Advantages:***

The interface is neat, elegant and accessible.

Many useful features

Capacity quite light.

The program runs stably, no errors or crashes occur during use.

Do not heat the appliance.

* ***Defect:***

Cannot open a tree from an image

1. **Development**

* We want to insert an **AVL tree** (named after inventors **A**delson-**V**elsky and **L**andis) is a [self-balancing binary search tree](https://en.wikipedia.org/wiki/Self-balancing_binary_search_tree). It was the first such [data structure](https://en.wikipedia.org/wiki/Data_structure) to be invented. In an AVL tree, the [heights](https://en.wikipedia.org/wiki/Tree_height) of the two [child](https://en.wikipedia.org/wiki/Child_node) subtrees of any node differ by at most one; if at any time they differ by more than one, rebalancing is done to restore this property.