# **Binary Trees Report**

# Design

## Introduction

For this assignment it would have been difficult to search through a list to look for a student because you would need to look at every node until you find it. This is very troublesome as it is time consuming especially for large amounts of data. Therefore binary trees would be more efficient as a student could be found quicker on the basis that a nodes left child is smaller and its right is greater. Using these rules nodes can be found very quickly making it a more suitable data structure.

## Requirements

***R1.*** Allow the user to enter details for a student.

***R2.*** Allow the user to request that the whole tree be printed in numerical order of student id.

***R3.*** Allow the user to enter a student id to look for in the tree and show the exam mark for that student.

***R4.*** Allow the user to request that a particular student is removed from the tree.

***R5.*** Write out your tree to a text file, and then read it in again at the start of the program

***R6.*** Implement a preorder and postorder traversal to display the tree contents.

## Pseudocode

### insert method

*if node is equal to null*

*return a new treendode*

*if id is less than the node’s student id*

*inserts the new node to the left node*

*else if id is greater than the node’s student id*

*inserts the new node to the right node*

*else*

*Prints out “Error - entry already exists”*

*Returns node*

### find method

*Creates “search” of type Treenodes and sets it to null*

*If the node is not equal to null*

*If id is less than the node’s student id*

*sets search to find and passes left node and id*

*if id is greater than the node’s student id*

*sets search to find and passes right node and id*

*if id is equal to the node’s student id*

*return node*

*return search*

### delete method

*If the node is not equal to null*

*Print out* *"Entry does not exist"*

*return node*

*If id is less than the node’s student id*

*sets left node to delete and passes left node and id*

*if id is greater than the node’s student id*

*sets right node to delete and passes right node and id*

*if id is equal to the node’s student id*

*if left node is equal to null*

*Print out "Successfully deleted"*

*return right node*

*else if right node is equal to null*

*Print out "Successfully deleted"*

*return left node*

*sets “tmp” of type Treenodes to equal successor passing right node*

*sets node’s student id to temporary student id*

*sets node’s exam mark to temporary exam mark*

*sets right node to delete and passes right node and node’s student id*

*return node*

|  |
| --- |
| **Menu** |
| * tree: Tree |
| * mainMenu() * insertChoice() * findChoice() * deleteChoice() |

## Class diagram

Creates object

|  |
| --- |
| **Tree** |
| * root: Treenodes |
| * insert(mark, id, node) * printInOrder(node) * printPostOrder(node) * printPreOrder(node) * find(node, id) * delete(node, id) * successor(node) * traverseInOrder(node) * save() * load(fname) |

Creates object

|  |
| --- |
| **Main** |
| * indent: int |
| * main() * displayTree(p) |

|  |
| --- |
| **Treenodes** |
| * examMark: int * studentId: int * left: Treenodes * right: Treenodes * parent: Treenodes |
| * Treenodes(mark, id) |

Creates object

## AC12001 – Test Plan

Name: …Isaac Lowry…………………………………………………………………………..

Matric number: …170025555 …………………………………………………….……….

Lab Title: …AC12001 assignment 3: Binary Trees………………………………..……….

Test number/date/version: 07/03/18 ……………………………………….……..

Test Notes: …Tests completed manually ……………………………

|  |  |  |  |
| --- | --- | --- | --- |
| **Test Description** | **Test Data** | **Expected result** | **Worked?** |
| Inserting student mark | 123 90 | Asks for next student | Y |
| Inserting student mark | 12390 | Outputs “Wrong input” error | Y |
| Inserting student mark | a b | Outputs “Wrong input” error | Y |
| Inserting student mark | 12,9 | Outputs “Wrong input” error | Y |
| Inserting student mark | 123 56 | Outputs message saying entry already exists | Y |
| Inserting student mark | 5, 25 | Outputs “Wrong input” error | Y |
| Inserting student mark | 2+2 | Outputs “Wrong input” error | Y |
| Finding student | 123 | Displays student ID and exam mark | Y |
| Finding student | 123 90 | Outputs “Wrong input” error | Y |
| Finding student | 8 | Outputs message saying entry does not exist | Y |
| Finding student | A5 | Outputs “Wrong input” error | Y |
| Deleting a student | 123 90 | Outputs “Wrong input” error | Y |
| Deleting a student | 8 | Outputs message saying entry does not exist | Y |
| Deleting a student | 123 | Deletes and outputs “successfully deleted” | Y |
| Deleting a student | 123 | Outputs message saying entry does not exist | Y |

# Self-evaluation

I found this assignment relatively challenging as I had to learn the concept of binary trees. I found completing the assignment to be enjoyable as understanding the data structure’s system was tricky especially when putting it into code. I was proud to have achieved all the requirements as well as two of the optional requirement of saving and loading text files as well as preorder and postorder transversal. However I found extending the program to do tree balancing to be too difficult to understand or use in my assignment because of my lack of coding experience.