|  |
| --- |
| **Department of Computer and Software Engineering – ITU** |
| **SE200T: Data Structures & Algorithms** |

|  |  |
| --- | --- |
| **Course Instructor: Usama Bin Shakeel** | **Dated: 5th Oct 2024** |
| **Teaching Assistant: Zainab, Sadia & Ryan** | **Semester: Fall 2024** |
| **Session: 2024-2028** | **Batch: BSSE2023B** |

# **Assignment 7. Implementing a Tree Structure in a Single Class**

|  |  |  |
| --- | --- | --- |
| **Name** | **Roll number** | **Obtained Marks/35** |
| Muhammad Mukarram Raza | BSSE-23029-B |  |

Checked on: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Signature: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**Submission:**

• Email instructor or TA if there are any questions. You cannot look at others’ solutions or use others’ solutions, however, you can discuss it with each other. Plagiarism will be dealt with according to the course policy.

• Submission after due time will not be accepted.

**In this assignment you have to do following tasks:**

**Task 1:** Ensure that you have installed all three softwares in your personal computer (Github, Cygwin & CLion). Now, accept the assignment posted in the classroom (e.g Google, LMS etc) and after accepting, clone the repository to your computer. Make sure you have logged into the github app with your account.

**Task 2:** Open Cygwin app, Move to your code directory with following command “cd <path\_of\_folder>”, <path\_of\_folder> can be automatically populated by dragging the folder and dropping it to the cygwin window.

Run the code through Cygwin, use command “make run”, to get the output of the code

**Task 3:** Solve the given problems, write code using **CLion** or any other IDE.

**Task 4:** Keep your code in the respective git cloned folder.

**Task 5:** Commit and Push the changes through the Github App

**Task 5**: Write the code in separate files **(as instructed**). Ensure that file names are in lowercase (e,g **main.cpp**).

**Task 6:** Run ‘**make run**’ to run C++ code

**Task 7:** Run ‘**make test**’ to test the C++ code

Write code in functions, after completing each part, verify through running code using **“make run”** on Cygwin. Make sure to test the code using **“make test”.**

## **Objective**

The goal of this assignment is to create a tree data structure using a single class, encapsulating all functionalities required to manage the tree's nodes, insert values, search for values, delete nodes, and calculate properties like height and size.

## **Overview of Tree Implementation**

A tree can be efficiently represented using a single class that contains attributes and methods for managing nodes and their relationships. Each node in the tree can have multiple children, making it suitable for various applications like hierarchical data representation.

**Create a global variable MaxChild and set it to 3:** This variable will store the max no. of child a node can hold

### Implement the Tree class

**Data Members:**

* **T data**: The value stored in the node. (Template variable)
* **Tree\* parent :** Parent Node
* **Tree\* children[MaxChild]:** Array to store child objects of the same class

**Member Functions to Implement:**

* **Tree(T value)**: Constructor to initialize the node with a given value.
* **void insert(T value):** Insert value in tree using level order traversal
* **int getData()**: Returns the data stored in the node.
* **Tree<T>\* getChild(**int data, int childNumber**) :** Returns the childNumber Tree address
* **Tree<T>\*\* delete(T value):** Delete value from tree
* **int height():** Returns height of tree
* **int degreeOfTree():** Returns degree of tree
* **void update (T oldValue, T newValue):** Update old value of node with new value
* **Tree<T>\*searchInTree(T value):** Search using level order traversal
* **int treeSize():** Returns the total number of nodes in a tree

*Please read the following instructions carefully:*

1. ***Do Not Modify test.cpp:*** *You are strictly prohibited from making any changes to the test.cpp file. This file is designed to test your implementation and any modifications will lead to the assignment being graded as zero.*
2. ***Class Definitions:*** *All class definitions and implementations must be provided solely within the files functions.h and functions.cpp. You are not allowed to create any additional files for your class definitions or implementations.*

*Any deviation from these rules, including creating additional files or modifying the test.cpp file, will result in your assignment receiving a grade of zero.*

**Assessment Rubric for Assignment**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Performance metric** | **CLO** | **Able to complete the task over 80% (4-5)** | **Able to complete the task 50-80% (2-3)** | **Able to complete the task below 50% (0-1)** | **Marks** |
| 1. Realization of experiment | 3 | Executes without errors excellent user prompts, good use of symbols, spacing in output. The testing has been completed. | Executes without errors, user prompts are understandable, minimum use of symbols or spacing in output. Some testing has been completed. | Does not execute due to syntax errors, runtime errors, user prompts are misleading or non- existent. No testing has been completed. |  |
| 1. Conducting experiment | 2 | Able to make changes and answer all questions. | Partially able to make changes and few incorrect answers. | Unable to make changes and answer all questions. |  |
| 1. Computer use | 4 | Document submission timely. | Document submission late. | Document submission not done. |  |
| 1. Teamwork | 4 | Actively engages and cooperates with other group member(s) in an effective manner. | Cooperates with other group member(s) in a reasonable manner but conduct can be improved. | Distracts or discourages other group members from conducting the experiment |  |
| 1. Laboratory safety and disciplinary rules | 2 | Code comments are added and do help the reader to understand the code. | Code comments are added and do not help the reader to understand the code. | Code comments are not added. |  |
| 1. Data collection | 2 | Excellent use of white space, creatively organized work, excellent use of variables and constants, correct identifiers for constants, No line-wrap. | Includes name, and assignment, white space makes the program fairly easy to read. Title, organized work, good use of variables. | Poor use of white space (indentation, blank lines) making code hard to read, disorganized and messy. |  |
| 1. Data analysis | 3 | Solution is efficient, easy to understand, and maintain. | A logical solution that is easy to follow but it is not the most efficient. | A difficult and inefficient solution. |  |
| **Total (out of 35):** | | | | |  |