**DATA STRUCTURES – FALL 2021**

**LAB 11**

**Learning Outcomes**

In this laboratory, you will implement operations of Binary Search Tree.

**TASK 1**

A reflection of a binary tree is a tree in which all the left and right children of non-leaf nodes are interchanged. Consider the example below

Graphical user interface, text, application

Description automatically generated

Use iterative approach to solve this problem. Use a queue to save the values while level order traversal. A pseudocode for the example is given below

*Initialize queue Q*

*Enqueue the root*

*while Q is not empty, do*

*current-node = Q.front()*

*Dequeue the node*

*swap current-node.left and current-node.right*

*push back the left and right child into the queue*

**TASK 2**

An expression tree is a binary tree which is used to represent expressions. In expression tree, nodes correspond to the operator and each leaf node corresponds to the operand. For example, the equation **(a+b)\*(c\*(d+e))** results in the postfix expression **ab+cde+\*\***

The tree representation for this postfix notation could be

Graphical user interface

Description automatically generated with medium confidence

The construction of expression tree takes place by reading postfix expression one symbol at a time. If the symbol is an operand, a new binary tree node is created, and its pointer is pushed onto a stack. If the symbol is an operator, the pointers to two trees i.e., x and y are popped from the stack and a new tree whose root is the operator and whose left and right children point to y and x respectively is formed. A pointer to this new tree is then pushed to the stack. Finally, a pointer to the full expression three remains on stack.

For the algorithm of this task, please refer to the **Chapter 16- Tree implementation, Slide 33 (Expression Tree)** on google classroom.