Seatwork 9.2	
Implementing Trees 2	
Course Code: CPE010	**Program:** Computer Engineering
Course Title: Data Structures and Algorithms	**Date Performed:** 09 - 25 - 25
Section: CPE21S4	**Date Submitted:** 09 - 25 - 25
Name(s): Santiago, David Owen A.	**Instructor:** Engr. Jimlord Quejado
6. Output	

Objectives: To create a program in C++ using search trees.

1. To be familiarized with the search tree algorithm.

2. To be able to create a program with a search tree algorithm.

Answer the following questions:

1. What is a binary search tree?
   A binary search tree is a data structure that contains a root node that branches out into other sub nodes. All nodes under this root node can only branch out to two other nodes maximum, hence, its name—binary.

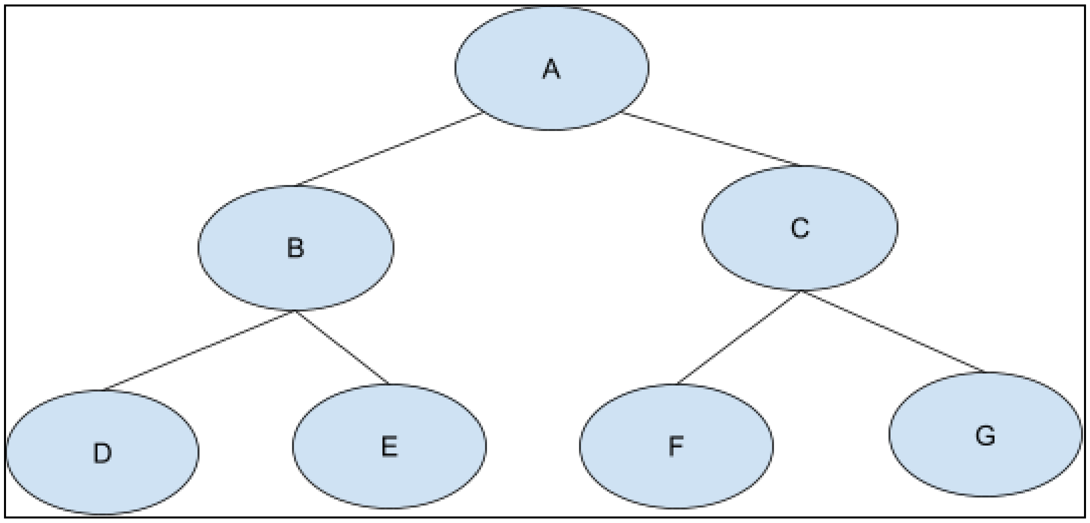
2. Where can binary search trees be used?
   They can be used for yes or no algorithms where each decision or each traversal per node will depend on whether the input is 1 or 0. However, once the nodes have specific conditions like; the left child must be less than the right child or vice versa; then binary search trees become useful algorithms in different uses such as with sorting, inserting, and deletion of data.

3. What is a tree traversal?
   Tree traversal refers to travelling or accessing each node all in a specific order. It can begin from the root node all the way to the leaf nodes or it can begin at the opposite—starting from the leaf nodes all the way back to the root node.

4. Differentiate a post-order and pre-order traversal give examples.
   Post-order refers to accessing the binary tree by beginning at the leaf node, to its partnered leaf node, to their parent node, to the partnered node of the parent node, then back to the parent nodes' parent node, all the way back to the root node. In simple terms, it refers to travelling backwards starting from the leaves all the way to the roots.

Pre-order traversal refers to accessing the binary tree by beginning at the root node, choosing one main child node, accessing the child nodes' child nodes, then back to the other main child node and accessing its child nodes. Simply put, it accesses the binary tree by beginning at the root node then traversing through its child nodes and subchild nodes.

For example:



Pre-Order Traversal:

A B D E C F G

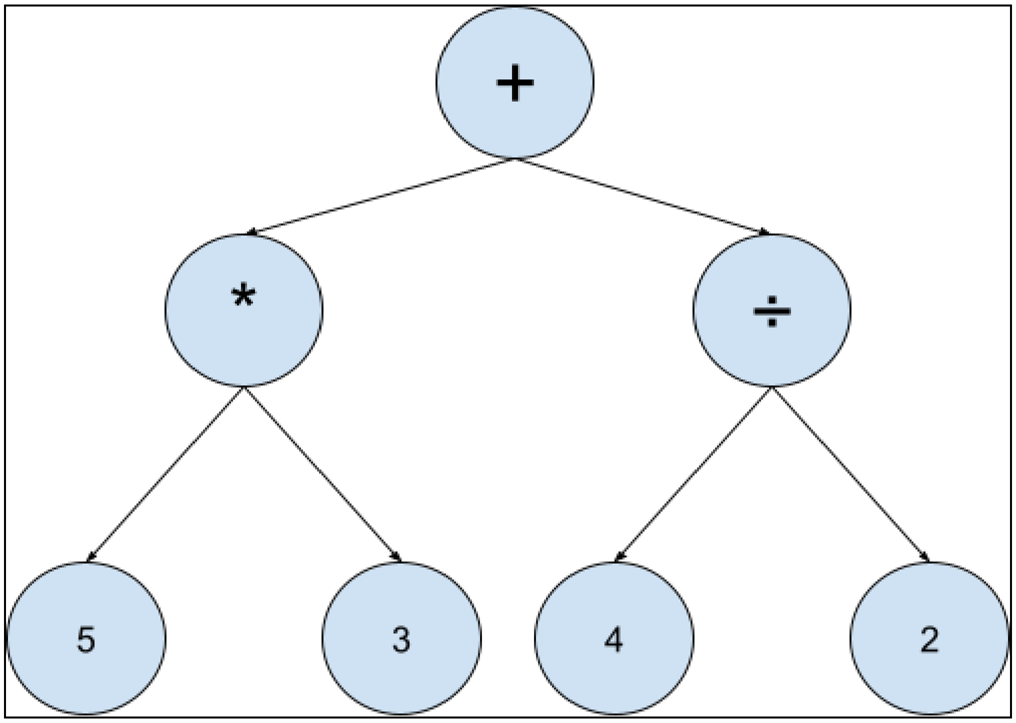
Post-Order Traversal:

D E B F G C A

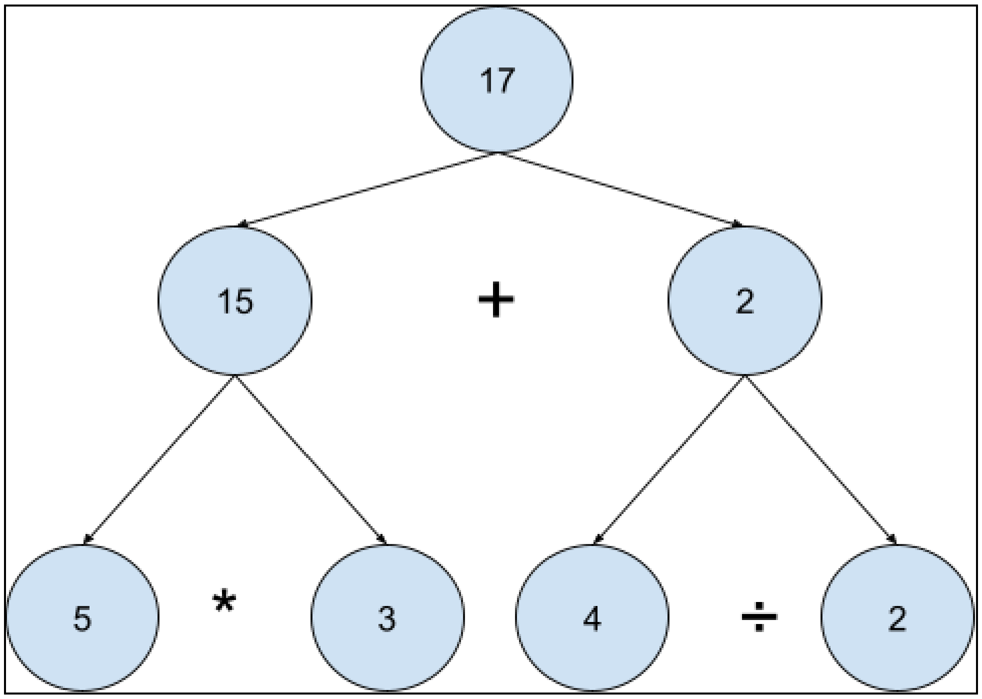
5. What is a parse tree and its use?

   A parse tree is a representation of how a string is broken down depending on a context-free grammar or CFG. By following this specific grammar, we can derive the elements of a string or an expression individually and order them hierarchically. The components of a string or an expression are broken down into parts and arranged into a hierarchical manner to determine the order of operations to be done using sub trees.

   For an example, in the mathematical expression: [(5 * 3) + (4 / 2)], the parse tree would look like this:



In this parse tree, we can clearly see the order of operations to resolve the expression. Before we can do the main operation of addition, we must first resolve the two subtrees. The first subtree is 5*3 then the second subtree is 4 ÷ 2, once we finish those operations, we can then proceed to addition. The resultant subtree would look like this:



The most common use-case of a parse tree is in compiler design. It depicts the syntactic structure of a string based on the given grammar. Through a parse tree, we can see how it is derived and broken. User input strings are broken down into smaller units that are translated based on the given grammar to be turned into a program. A parse tree clearly shows the steps taken to break down the syntax of an input string to be readable code by a computer.

References:

GeeksforGeeks. (2025a, July 23). *Applications, Advantages and Disadvantages of Binary Search Tree*. GeeksforGeeks. https://www.geeksforgeeks.org/dsa/applications-advantages-and-disadvantages-of-binary-search-tree/

GeeksforGeeks. (2025b, September 24). *Binary search tree*. GeeksforGeeks. https://www.geeksforgeeks.org/dsa/binary-search-tree-data-structure/

GeeksforGeeks. (2025a, July 15). *Parse tree in compiler design*. GeeksforGeeks. https://www.geeksforgeeks.org/compiler-design/parse-tree-in-compiler-design/

GeeksforGeeks. (2025, September 16). *Tree traversal techniques*. GeeksforGeeks. https://www.geeksforgeeks.org/dsa/tree-traversals-inorder-preorder-and-postorder/

Miller, B., Ranum, D., & Pearce, J. (2018). *Parse Tree*. In *Problem Solving with Algorithms and Data Structures using C++*. Retrieved from https://runestone.academy/ns/books/published/cppds/Trees/ParseTree.html

Parse tree in automata theory. (n.d.). https://www.tutorialspoint.com/automata_theory/automata_theory_parse_tree.htm

ScienceDirect. (n.d.). Parse Tree. In Computer Science Topics. Retrieved from https://www.sciencedirect.com/topics/computer-science/parse-tree

8. Assessment Rubric