Seatwork 9.2	
Implementing Trees 2	
Course Code: CPE010	**Program:** Computer Engineering
Course Title: Data Structures and Algorithms	**Date Performed:** 9/25/25
Section: CPE21S4	**Date Submitted:** 9/25/25
Name(s): Avila, Vince Gabriel V.	**Instructor:** Engr. Jimlord Quejado
6. Output	

1. What is a binary search tree?

A binary search tree is a type of data structure that allows each node to have a maximum of two children. In a BST all the values on the left are smaller, where all the values on the right are larger. This property allows for quick perceptions of the value while also conveniently keeping data in a certain order which provides ease of search. BST allows adding and removing values with ease and structure.

2. Where can binary search trees be used?

A BST is useful in searching data, generating databases, and ordering information. These data structures are helpful, they save time when searching for a particular object. BSTs are used in applications including but not limited to dictionaries and file systems.

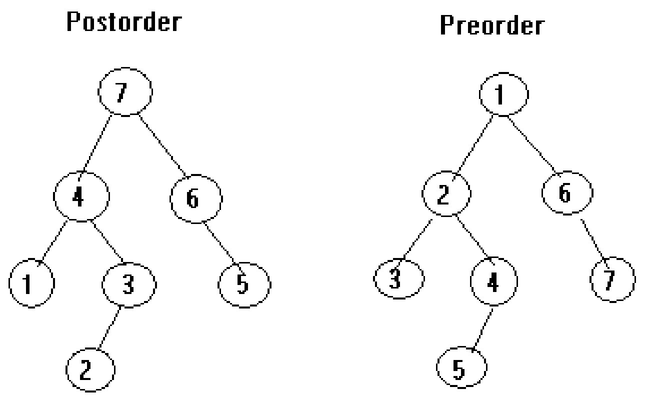
3. What is a tree traversal?

Traversing a tree is the implicit way of visiting each node in a tree in some order, so that the data is read or processed. While traversing the tree, it goes through different orderings of the values stored in the tree.

4. Differentiate a post-order and pre-order traversal give examples.

In pre-order tree traversal, we visit the root first, then move to the left and then the right value. The order is (A-B-C). In post-order tree traversing, we visit the left first, then right and then root last. The order is (B-C-A). Pre-order tree traversing is useful in making copies of trees and post-order may be used to delete trees, while both are used to solve different strategies of problems from traversing trees.

Examples for post-order and pre-order traversal:



4. What is a parse tree and its use?

A parse tree demonstrates how a sentence or set of code adheres to grammar. Parse trees are important in compilers to check code and translate it into machine language. When dealing with complex code, it helps break down pieces into smaller parts which are easier for computers to process and execute.

References:

GeeksforGeeks. (2025, August 19). *Introduction to Binary tree*. GeeksforGeeks.

https://www.geeksforgeeks.org/dsa/introduction-to-binary-tree/

GeeksforGeeks. (2025a, July 23). *Postorder traversal of binary tree*. GeeksforGeeks.

https://www.geeksforgeeks.org/dsa/postorder-traversal-of-binary-tree/

GeeksforGeeks. (2025b, July 23). *Preorder traversal of binary tree*. GeeksforGeeks.

https://www.geeksforgeeks.org/dsa/preorder-traversal-of-binary-tree/

GeeksforGeeks. (2025a, July 15). *Parse tree in compiler design*. GeeksforGeeks.

https://www.geeksforgeeks.org/compiler-design/parse-tree-in-compiler-design/

Operations on Binary Search Tree�s. (n.d.).

https://www.cs.cmu.edu/~clo/www/CMU/DataStructures/Lessons/lesson4_3.htm

8. Conclusion

In my conclusion, binary trees help keep data organized for easy searchability. Tree traversals are methods for visiting and reading all the parts of a tree. Two of the traversals, pre-order and post-order, have different applications. Using parse trees in computer programs allows them to check or translate computer code.

9. Assessment Rubric