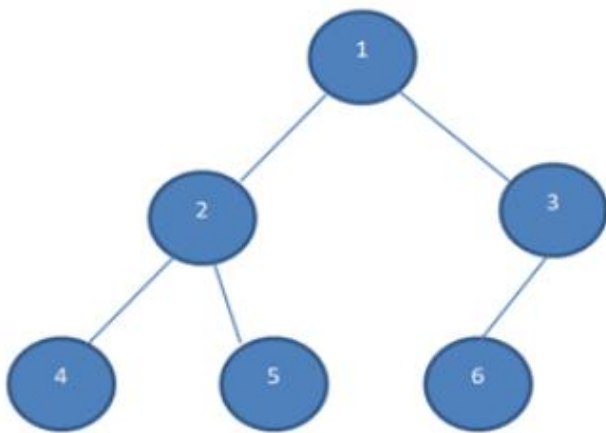


<b>Activity No. &lt;9&gt;</b>	
<b>Quiz 9.1 - Tree Structure</b>	
<b>Course Code:</b> CPE010	<b>Program:</b> Computer Engineering
<b>Course Title:</b> Data Structures and Algorithms	<b>Date Performed:</b> 10/2/25
<b>Section:</b> CPE21S4	<b>Date Submitted:</b> 10/2/25
<b>Name(s):</b> Alcantara, Jason P.	<b>Instructor:</b> Engr. Jimlord Quejado
<p><b>1. Identify The Following:</b></p> <p><b>A. Tree –</b> Trees are used in many ways in computer science, computer engineering including operating system graphics, database systems and computer networking. The tree data structure has a root, branches and leaves. It is also represent relationship in data elements in a structures.</p> <p><b>B. Child node –</b> In a binary tree the data structure where every node has at most 2 children node. So basically any node that that comes from another node it's like a family tree, so if there has a parent and their kids are the child nodes The parent node connects every single one of them, and they branch out from it.</p> <p><b>C. Parent node –</b> The parent node in tree data structure is basically that has one or more child nodes connected to it. It's like an ancestors who rise to other. So every time I think of a tree diagram. The parent node is always on the top of the nodes to the next ground level.</p> <p><b>D. Root node-</b> In the root this like the starting point of the tree it's on the top and this is the one who doesn't have a parent node. Every other nodes in tree branches out from it, it's either directly or indirectly. It's like from the real tree trunk, from there you will get a branches which are the child nodes, and those branches can have more branches. But we need to always remember that everything is starts from the root.</p> <p><b>E. Tree Traversal –</b> The tree traversal this is just other fancy way of saying “visiting every node in the tree. This is the process of view or obtaining each node of the tree and it can also perform an operation like correcting and updating the node's data. Tree traversal is also very essential for searching a specific value of the tree. And the tree traversal can be make the codes very neatly but if we make an excessive function calls it can cause memory to run out.</p> <p><b>F. Parse Tree –</b> The parse tree is a tree structure that shows how a string is generated from a grammar. It's used in compilers and language processing to understand the structure of the code and sentences clearly. The parse tree made to build a step-by-step full expression. It's like an grammar expression can be made if you have a grammar rule that says an expression can be made up of numbers and operators.</p>	

- G. **Order of Precedence** - The order of precedence is the rule that decides which operation happens first in a programming expression. Example for that is multiplication happens before subtraction unless the user uses parenthesis. This will help to avoid confusion and make sure everyone gets the same result or answer. Without it, the expression will be solved in different ways and gives the wrong answer or result to the user. This is like a set of instructions that will tell you what you need to do first and so on.
- H. **Parsing** – Parsing is the process of breaking down information into smaller parts so a computer can understand it. It's like translating a sentence into pieces that show what each word means and how they work together. In programming, parsing helps computers read code or data correctly. For example, when you type a math problem, parsing helps the computer figure out what to solve first. It's a key step in making sure instructions are followed the right way.
- I. **Preorder Traversal** – pre order traversal is where each node is visited in a very specific sequence like left subtree, right subtree and the root node. The user will always start from the root node first then go to the left children node then lastly to the right children node. This method is useful when the user wants to copy or print the structure of the tree. This is like reviewing or checking the main folder first then open each subfolder one by one left to right.
- J. **Inorder Traversal** - In-order traversal is a way to visit nodes in a binary tree in a specific order. You start with the left child, then visit the root, and finally the right child. It follows the pattern: left → root → right. This method is useful when you want to get values in sorted order from a binary search tree. It's like reading a book from left to right, one chapter at a time.

2. Identify the parts of the following tree. (10)



- Root
- Parent
- Children
- siblings
- Edge
- Node
- Path
- Level

- Height
- Subtree

**3. Delete the node with an element value of 2 in the above tree structure.**

Function deleteNode(root, value):

    If root is NULL:

        Return NULL

    If root.value == value:

        // Case 1: Node has no children

        If root.left is NULL and root.right is NULL:

            Return NULL

        // Case 2: Node has one child

        If root.left is NULL:

            Return root.right

        If root.right is NULL:

            Return root.left

        // Case 3: Node has two children

        // Find the inorder successor (smallest in right subtree)

        successor = findMin(root.right)

        root.value = successor.value

        root.right = deleteNode(root.right, successor.value)

        Return root

    // Recurse on left and right subtrees

    root.left = deleteNode(root.left, value)

    root.right = deleteNode(root.right, value)

    Return root

Function findMin(node):

    While node.left is not NULL:

        node = node.left

    Return node