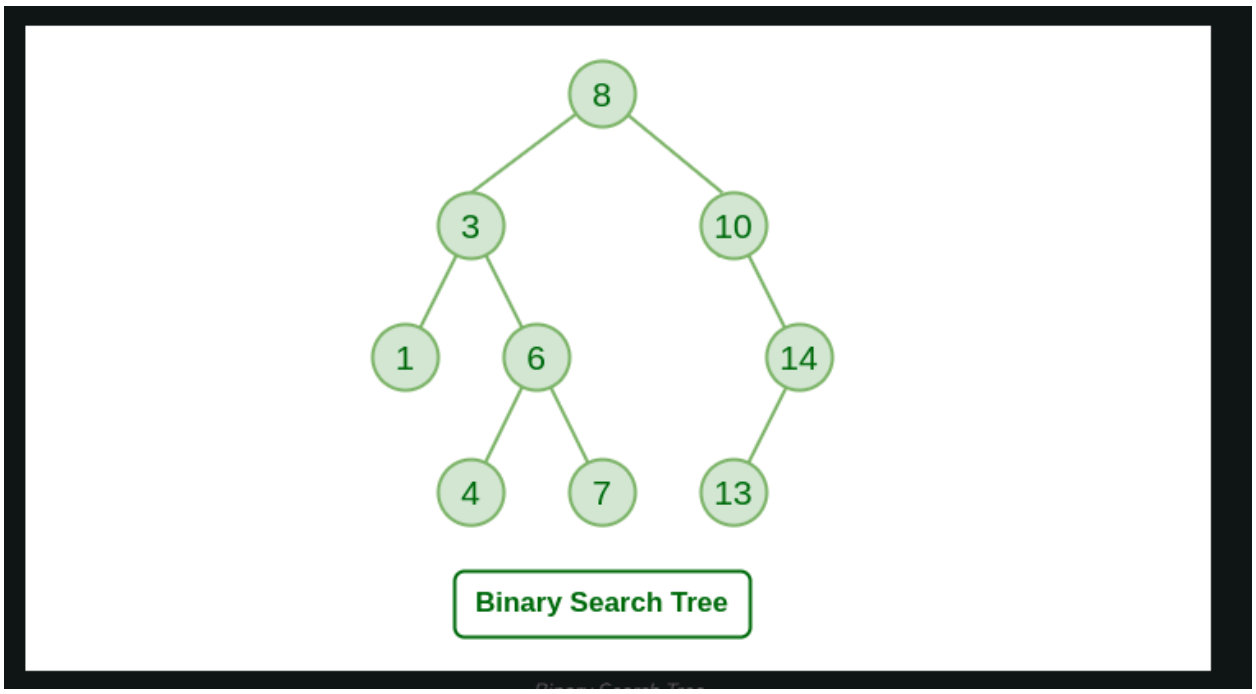
Answer the following questions:

1. What is a binary search tree?

A binary search tree is a data structure that is used for storing data in an organized manner. In some binary tree implementations, they have a property that on the left side of the key/root, is less than key and on the right side is greater than the key. Additionally, binary search trees are good when it comes to insert, delete, traversing, and finding the max and min value on the tree. According to, (GeeksforGeeks, 2025).



This is a well structured binary search tree, where the value on the left is less than the key/root and on the right is greater than the key.
GeeksforGeeks. (2025, July 15). *Introduction to binary search tree*. GeeksforGeeks. https://www.geeksforgeeks.org/dsa/introduction-to-binary-search-tree/

2. Where can binary search tree be used?

According to the, (*Real-World Applications of Binary Search Algorithm*, n.d.) one specific application of binary search tree is Financial application. For example, in stock market analysis, a binary search tree can locate specific price points or time periods in some long historical data. Traders and analysts rely on this capability to make informed decisions based on some patterns spreading on the internet. This is essential for financial institutions that need to price derivatives and manage risk effectively.

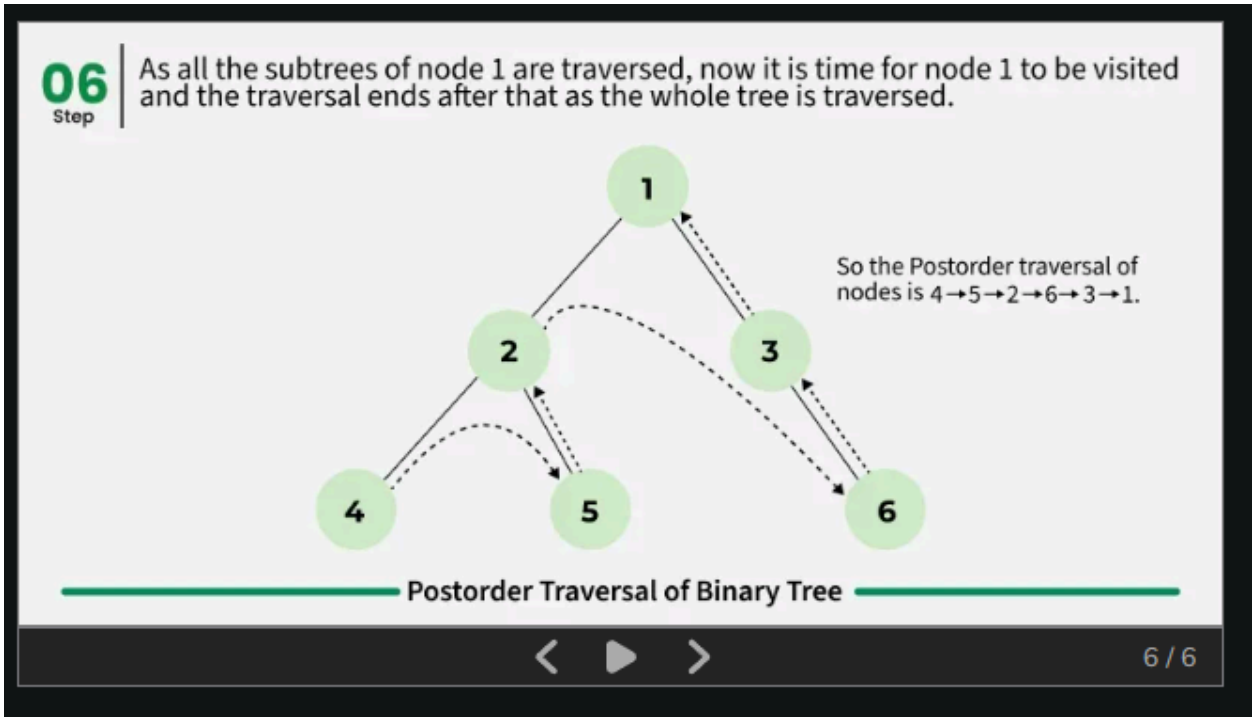
Real-World applications of binary search algorithm. (n.d.). Youcademy. https://youcademy.org/applications-of-binary-search/

3. What is a tree traversal.

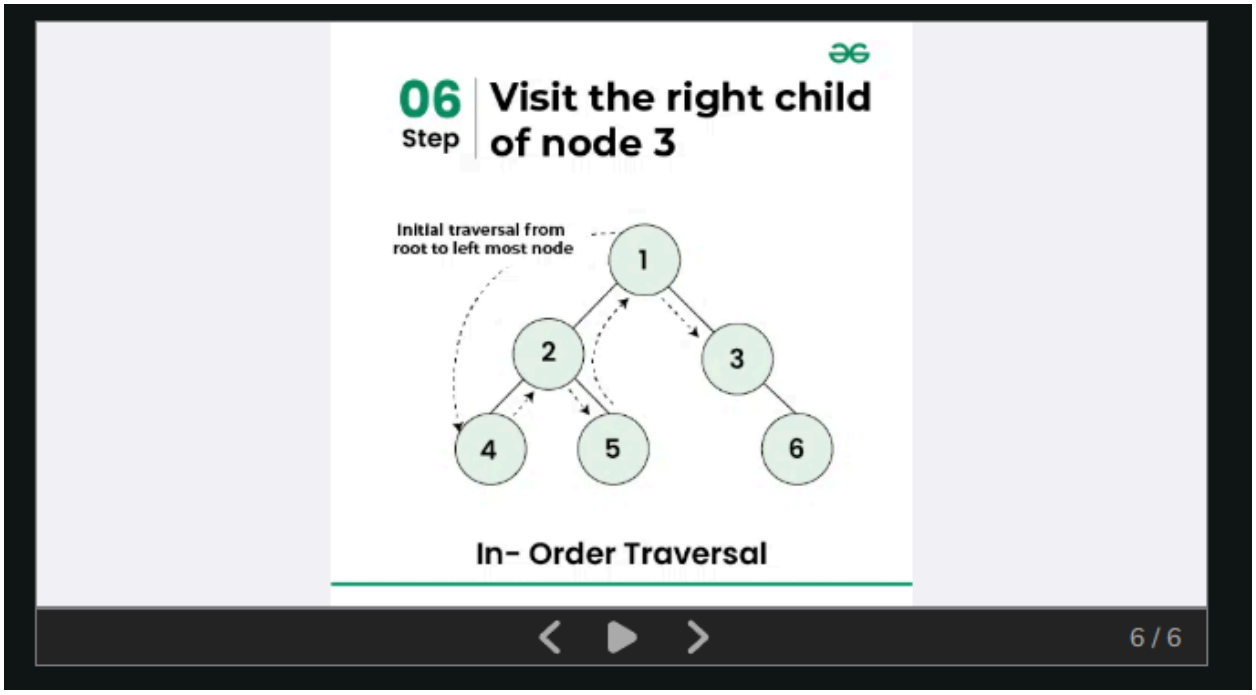
A tree traversal is a simple way to visit all the nodes in the tree, when we need to see or find something inside the tree. We are using a traversal tree using a variable to traverse all the nodes inside the tree. We always have to start at the key whether the node that we are looking for is at the bottom.

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

So when we say post-order traversal. We are traversing the tree, starting at the left most node, we call this as the subtree, then move to its sibling node, parent node, then to the other part of the tree, then to its parent node and lastly to the key/root node. An example illustrated by, (GeeksforGeeks, 2025b)



This is an illustration on how the post order works. Where in the pre-order traversal, starts at the again left most part of the subtree, then moves to the parent node, to the sibling node, then the key/root node. After the left side of the node it will start again at the right side but this time it will start at the parent node, then the child node. An example illustrated by, (GeeksforGeeks, 2025b)



GeeksforGeeks. (2025b, July 23). Postorder traversal of binary tree. GeeksforGeeks. https://www.geeksforgeeks.org/dsa/postorder-traversal-of-binary-tree/

5. What is a parse tree and its use?

A parse tree, also known as a derivation tree or a syntax tree, is a data structure used to represent the structure of a string generated by a context-free grammar. It provides a visual representation of how the string can be derived from the grammar rules. This parse tree is essential when it comes to analyzing a complex context language. A parse tree represents the syntactic structure of a string generated by a context-free grammar. It is a hierarchical tree structure where each node represents a symbol. According to, (*Introduction to Parsing and Syntax Trees: Unraveling the Structure of Code – AlgoCademy Blog*, n.d.). One of its uses is that Parse trees play a central role in compiler construction, where they capture the derivations built by a parser and facilitate syntax analysis, semantic interpretation, and code generation. Parse trees represent the syntactic structure of sentences, enabling applications such as grammar checking, machine translation, and information extraction. We can also find them in foundational treebanks, which are syntactically annotated corpora pairing sentences with their corresponding parse trees for linguistic analysis and parser evaluation.

Introduction to Parsing and Syntax Trees: Unraveling the Structure of code – AlgoCademy blog. (n.d.). https://algocademy.com/blog/introduction-to-parsing-and-syntax-trees-unraveling-the-structure-of-code/