**Depth-first search**

Main article: [Depth-first search](https://en.wikipedia.org/wiki/Depth-first_search)

Trees can be traversed in *pre-order*, *in-order*, or *post-order*.[[1]](https://en.wikipedia.org/wiki/Tree_traversal#cite_note-holtenotes-1) These searches are referred to as *depth-first search* (DFS), as the search tree is deepened as much as possible on each child before going to the next sibling. For a binary tree, they are defined as display operations recursively at each node, starting with the root, whose algorithm is as follows:[[2]](https://en.wikipedia.org/wiki/Tree_traversal#cite_note-2) [[3]](https://en.wikipedia.org/wiki/Tree_traversal#cite_note-3)

The general recursive pattern for traversing a (non-empty) binary tree is this: At node N you must do these three things:

(L) recursively traverse its left subtree. When this step is finished you are back at N again.

(R) recursively traverse its right subtree. When this step is finished you are back at N again.

(N) Actually process N itself.

We may do these things *in any order* and still have a legitimate traversal. If we do (L) before (R), we call it left-to-right traversal, otherwise we call it right-to-left traversal.

**Pre-order**

1. Check if the current node is empty / null
2. Display the data part of the root (or current node).
3. Traverse the left subtree by recursively calling the pre-order function.
4. Traverse the right subtree by recursively calling the pre-order function.

**In-order**

1. Check if the current node is empty / null
2. Traverse the left subtree by recursively calling the in-order function.
3. Display the data part of the root (or current node).
4. Traverse the right subtree by recursively calling the in-order function.

In a [search tree](https://en.wikipedia.org/wiki/Search_tree), in-order traversal retrieves data in sorted order.[[4]](https://en.wikipedia.org/wiki/Tree_traversal#cite_note-4)

**Post-order**

1. Check if the current node is empty / null
2. Traverse the left subtree by recursively calling the post-order function.
3. Traverse the right subtree by recursively calling the post-order function.
4. Display the data part of the root (or current node).