File Recursion

For finding files in a directory containing subdirectories, the most efficient way to do it is by recursively searching through the directories.

The directory could be represented as an N-ary Tree, also known as a multiway tree. Initially when thinking of how I would go about implementing it with a tree data structure, I thought of converting the directory to a tree. But converting would mean traversing it first so I did not use any data structure other than the internal stack from recursion. So the way I searched the tree is by using a DFS pre-order traversal of the directory using recursion. The time complexity is O(log n) as with most tree traversal algorithms and the space complexity is O(n) since there is a need for a list to store the data and a stack, implicitly called, when doing recursion. For improved efficiency, rather than using a simple list, a linked list was used to achieve constant time insertion

Sources

N-Ary Tree Data Structure. Studytonight.com. (n.d.). From https://www.studytonight.com/advanced-data-structures/nary-tree

Sharma, A. (2021, March 19). *Implementing general Tree using Python*. Carbon Coffee. From https://carboncoffee.hashnode.dev/implementing-general-tree-using-python

Wikimedia Foundation. (2022, October 21). *Depth-first search*. Wikipedia. From https://en.wikipedia.org/wiki/Depth-first_search

Vishal. (2022, January 19). *Python list files in a directory*. PYnative. Retrieved October 30, 2022, from https://pynative.com/python-list-files-in-a-directory/