EXPLANATION OF THE CODE (DSA)

This code implements an interactive menu-driven program to perform operations on three different data structures: Binary Trees, Binary Search Trees (BSTs), and Heaps. Each data structure is handled in separate modules with functions dedicated to their respective operations. The program starts with a main menu offering choices to explore these three structures or exit. Depending on the user's selection, corresponding submenus allow further operations.

For the Binary Tree, nodes are inserted level-by-level, using left-to-right traversal. Users can perform preorder, inorder, and postorder traversals to explore the tree. It also supports searching for a specific node and deleting a node, albeit with simplified logic focused on removing leaf nodes.

The Binary Search Tree (BST) provides an ordered structure where smaller values reside on the left and larger values on the right. Operations include node insertion, searching for a specific value, and deletion with proper reorganization of the tree. The deletion function handles three cases: leaf nodes, nodes with one child, and nodes with two children, utilizing an auxiliary function to find the smallest node in the right subtree during the latter case.

The Heap module enables the manipulation of a dynamic array-based representation of a binary heap. Users can insert nodes, build max-heaps (where parent nodes are greater than their children), and build min-heaps (where parents are smaller than their children). The heapify function ensures heap property is maintained during construction, and the program offers a simple print function to display the heap.

The program ensures modularity and user interactivity by splitting operations into clearly defined functions and providing intuitive menus for navigation. This structure simplifies the understanding and usability of various data structure algorithms for educational purposes.