Washington State University Vancouver
CS 360 Precheck
Pure-Recursive, Void Binary Search Trees

1 Overview

The accompanying header file (`tree.h`) contains a C header for a binary search tree which uses void pointers to be able to store keys and values of arbitrary types. The tree provides standard insert, search, and empty (but not node delete) functionality, along with two special operations: A tree walk function, which traverses the tree in order, calling a function on each key/value pair, and a path search function, which returns a linked list of all key/value pairs between the root and a given key (if present). Your task is to create a working implementation of the header's functionality and submit it as `tree.c`.

The header itself contains documentation for how each function needs to work. You may create and use additional helper functions if you like, but this is not required. (The reference implementation implements all required functionality using only the functions prototyped in the header file.) You may not, however, make use of any loops. All of your work must be done recursively.

2 Requirements

2.1 You Must

- Submit a single file named `tree.c`
- Implement all functions specified in the header, according to their descriptions
- Make use of the provided `tree_node` and `tree_path_name` structures where relevant
- Provide an implementation that is free of memory leaks or errors
- Compile without warning or error with the GCC flags `--std=c99`, `--pedantic`, and `-Wall`

2.2 You Must *NOT*

- Make use of any loops (`for`, `while`, or `do`)
- Make any changes to the provided header file
- Use global or static variables
- Include a `main` function in your submitted code
- Write any output to `stdout`

2.3 You Should

- Implement your own `main` function in its own file, for testing

2.4 You May

- Implement helper functions as part of your `tree.c`, but these are not required.