GTU Department of Computer Engineering

CSE 222/505 – Spring 2023

Homework #5 Report

REŞİT AYDIN

200104004019

1 Problem Definition

The problem requires the implementation of a Java program that reads information from a text file, represents it as a tree structure, and performs various operations on the tree using different algorithms. The program should have a proper object-oriented programming (OOP) structure.

We're asked to read Information and represent as Tree the program needs to read data from a text file (named "tree.txt") that contains rows representing data points, with each row split into categories by a ";" character. The program should parse this information into a tree structure and print the tree. The tree structure should be created using the JTree class from the javax.swing library.

We also had to implement some searching algorithm for tree such as BFS Search, DFS Search, Post Order Traversal. In last part which is moving a Node to Another Year. This part involves moving a problem/lecture/course node (which is not the root or a child of the root) from one year to another.

2 Problem Solution Approach

My **FileHierarchy** class aims to solve the problem of organizing file paths into a hierarchical tree structure and provides methods to search for specific files within the tree. The class follows a step-by-step approach to achieve this:

Reading Data from File: The **readFromFile()** method reads data from a specified file. Each line of the file represents a file path, and the method splits each line into elements using a semicolon delimiter. The file paths are stored in an organized manner using ArrayLists.

Converting Data to Tree Structure: The **arrayToTree()** method converts the file paths stored in the data field into a hierarchical tree structure. It iterates through each file path and checks if a child node with the same name already exists in the tree. If it does, the method moves to that child node. If not, it creates a new child node and adds it to the current node. This process is repeated for each level of the tree, resulting in a well-organized tree structure.

Visualizing the Tree: The **printTheImage()** method creates a visual representation of the hierarchical tree structure. It creates a graphical window (JFrame) and adds the tree to it. The method sets the necessary properties of the window, such as the title and size, and makes it visible. This allows users to view and understand the organization of the file paths.

Searching in the Tree: The **BFS(), DFS(),** and **postOrderTraversal()** methods provide different search algorithms to find specific files within the tree.

BFS(): uses a breadth-first search algorithm, which starts searching from the root of the tree and explores all the child nodes at each level before moving to the next level. It prints each visited node during the search process and returns true if the file is found, or false otherwise.

DFS(): uses a depth-first search algorithm, which starts searching from the root and explores as far as possible along each branch before backtracking. It prints each visited node during the search process and returns true if the file is found, or false otherwise.

postOrderTraversal(): uses a post-order traversal algorithm, which visits the child nodes of a parent node before visiting the parent node itself. It prints each visited node during the search process and returns true if the file is found, or false otherwise.

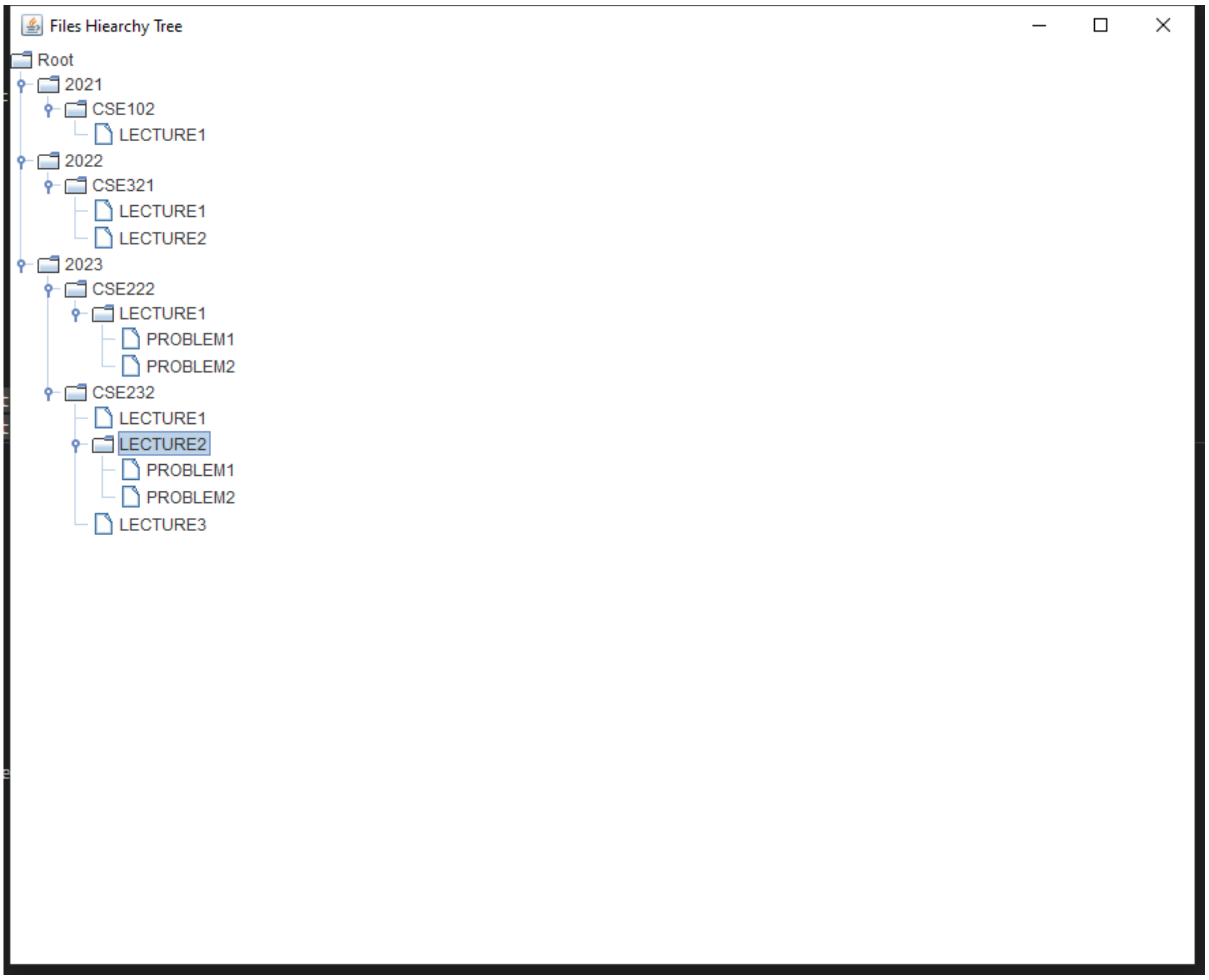
These search methods provide a systematic way to track the search process and locate specific files within the tree.

In summary, the **FileHierarchy** class provides a solution to organize file paths into a hierarchical tree structure and offers methods to search for specific files. It follows a clear sequence of steps to read the file, convert the data into a tree structure, visualize the tree, and perform efficient searches.

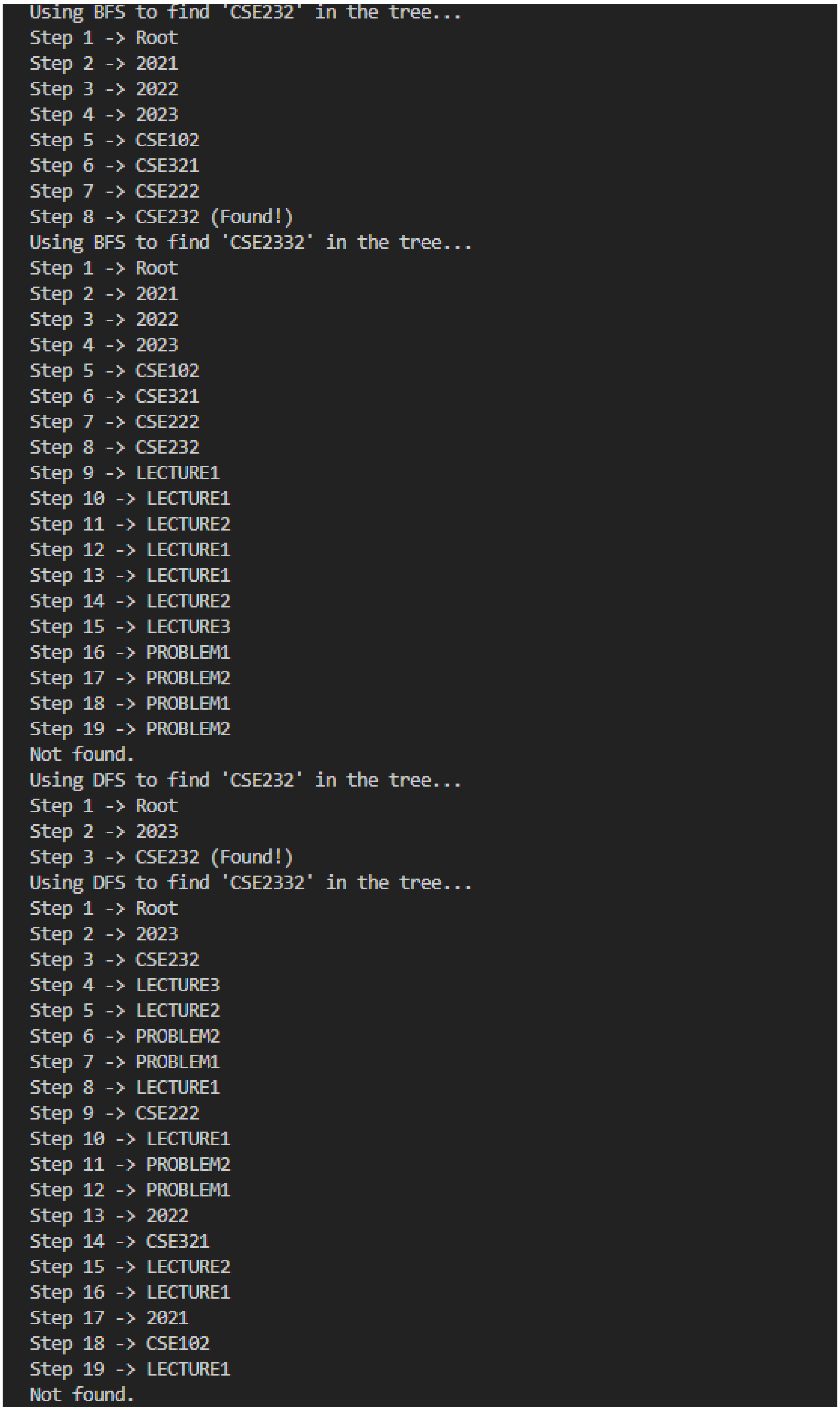
3 Running commands and Outputs

Running commands:

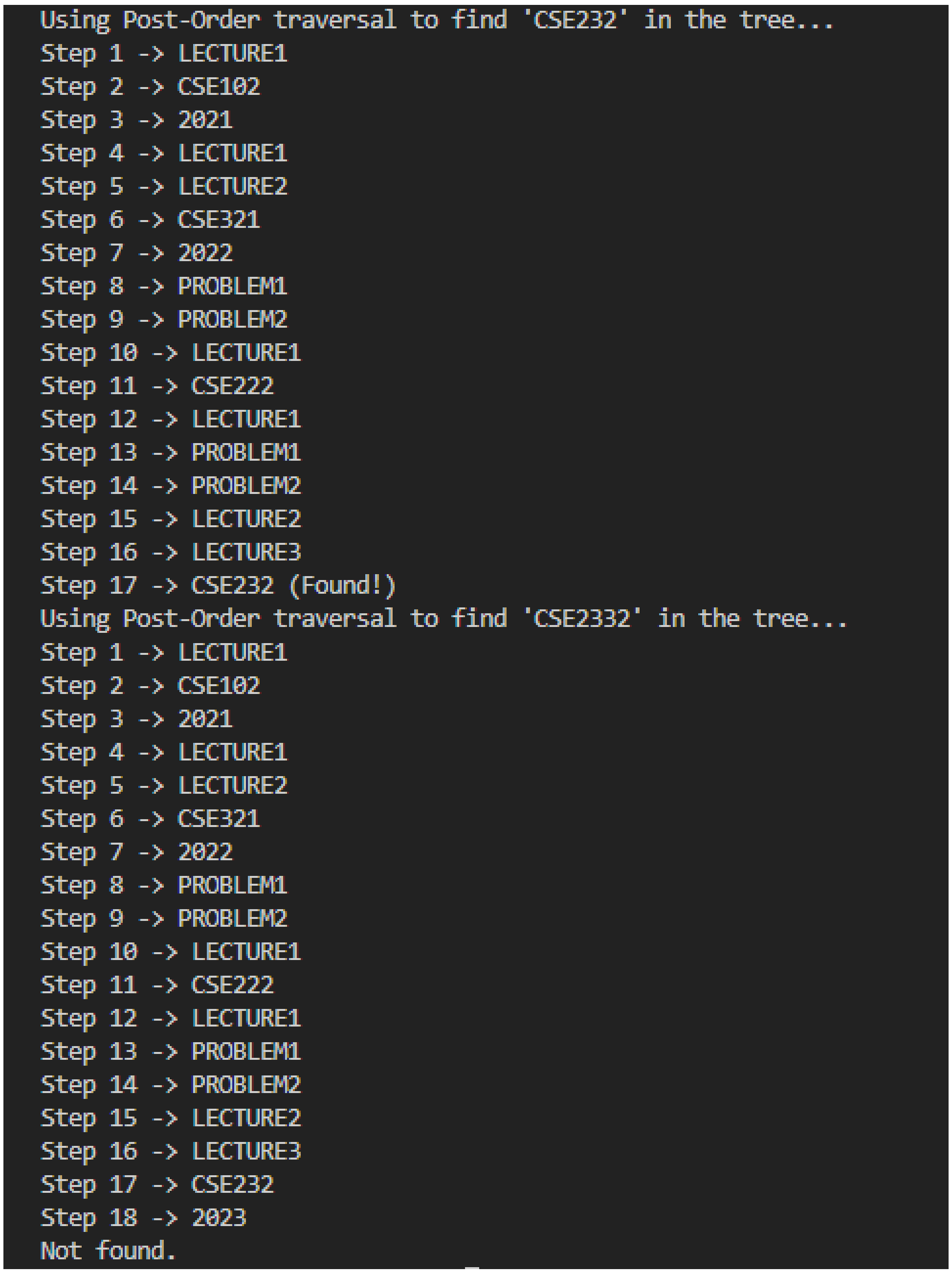
```
resitaga@resitaga:~/Desktop/HW5$ javac FileHierarchy.java
resitaga@resitaga:~/Desktop/HW5$ javac Test.java
resitaga@resitaga:~/Desktop/HW5$ java Test
```



```
Files Hiearchy Tree
Root
├ 2021
│  └ CSE102
│     └ LECTURE1
├ 2022
│  └ CSE321
│     ├ LECTURE1
│     └ LECTURE2
└ 2023
   ├ CSE222
   │  └ LECTURE1
   │     ├ PROBLEM1
   │     └ PROBLEM2
   └ CSE232
      ├ LECTURE1
      ├ LECTURE2
      │  ├ PROBLEM1
      │  └ PROBLEM2
      └ LECTURE3
```



```
Using BFS to find 'CSE232' in the tree...
Step 1 -> Root
Step 2 -> 2021
Step 3 -> 2022
Step 4 -> 2023
Step 5 -> CSE102
Step 6 -> CSE321
Step 7 -> CSE222
Step 8 -> CSE232 (Found!)
Using BFS to find 'CSE2332' in the tree...
Step 1 -> Root
Step 2 -> 2021
Step 3 -> 2022
Step 4 -> 2023
Step 5 -> CSE102
Step 6 -> CSE321
Step 7 -> CSE222
Step 8 -> CSE232
Step 9 -> LECTURE1
Step 10 -> LECTURE1
Step 11 -> LECTURE2
Step 12 -> LECTURE1
Step 13 -> LECTURE1
Step 14 -> LECTURE2
Step 15 -> LECTURE3
Step 16 -> PROBLEM1
Step 17 -> PROBLEM2
Step 18 -> PROBLEM1
Step 19 -> PROBLEM2
Not found.
Using DFS to find 'CSE232' in the tree...
Step 1 -> Root
Step 2 -> 2023
Step 3 -> CSE232 (Found!)
Using DFS to find 'CSE2332' in the tree...
Step 1 -> Root
Step 2 -> 2023
Step 3 -> CSE232
Step 4 -> LECTURE3
Step 5 -> LECTURE2
Step 6 -> PROBLEM2
Step 7 -> PROBLEM1
Step 8 -> LECTURE1
Step 9 -> CSE222
Step 10 -> LECTURE1
Step 11 -> PROBLEM2
Step 12 -> PROBLEM1
Step 13 -> 2022
Step 14 -> CSE321
Step 15 -> LECTURE2
Step 16 -> LECTURE1
Step 17 -> 2021
Step 18 -> CSE102
Step 19 -> LECTURE1
Not found.
```


```
Using Post-Order traversal to find 'CSE232' in the tree...
Step 1 -> LECTURE1
Step 2 -> CSE102
Step 3 -> 2021
Step 4 -> LECTURE1
Step 5 -> LECTURE2
Step 6 -> CSE321
Step 7 -> 2022
Step 8 -> PROBLEM1
Step 9 -> PROBLEM2
Step 10 -> LECTURE1
Step 11 -> CSE222
Step 12 -> LECTURE1
Step 13 -> PROBLEM1
Step 14 -> PROBLEM2
Step 15 -> LECTURE2
Step 16 -> LECTURE3
Step 17 -> CSE232 (Found!)
Using Post-Order traversal to find 'CSE2332' in the tree...
Step 1 -> LECTURE1
Step 2 -> CSE102
Step 3 -> 2021
Step 4 -> LECTURE1
Step 5 -> LECTURE2
Step 6 -> CSE321
Step 7 -> 2022
Step 8 -> PROBLEM1
Step 9 -> PROBLEM2
Step 10 -> LECTURE1
Step 11 -> CSE222
Step 12 -> LECTURE1
Step 13 -> PROBLEM1
Step 14 -> PROBLEM2
Step 15 -> LECTURE2
Step 16 -> LECTURE3
Step 17 -> CSE232
Step 18 -> 2023
Not found.
```