CENG 112 Data Structures
Assignments 4: Simple File System Manipulator

This homework will cover the topics given below:

- Stacks
- Queues
- Lists
- File I/O
- Generics
- Binary Search Trees
- N-Ary Trees

You are supposed to implement a program called Simple File System Manipulator in Java. It is a program through which you can manipulate the file system by issuing several commands supported. The commands are:

- *showContent <path>*: Print the content of the file at <path>
- *getHeight <path>*: Print height of the tree beginning from the <path>
- *getNumberOfFiles <path>*: Print the number of all files and sub files inside the <path>
- *removeFile <path>*: Deletes all files and sub files beginning from the <path>
- *traversePreOrderRecursively <path>*: Print all files and subfiles beginning from the <path> in pre order, the implementation must use **recursive** method, not iterative
- *traverseLevelOrder <path>:* Print all files and subfiles beginning from the <path> in level order
- *traversePostOrderIteratively <path>:* Print all files and subfiles beginning from the <path> in post order, the implementation must use **iterative** method, not recursive

When the program begins, first it reads all files in ”fs” directory residing in the current working directory. ”fs” directory includes all files metadata and their contents. Names of the files indicate the unique ids assigned to them, and they may include several lines. First line of each is the metadata about the file and the rest is the content. In case of being a *DIR* type file, the content is the ids of regular files residing in it otherwise it is a *REGFILE* which is just a regular file.

All paths begin with “/” character, so paths of files residing in the root directory is like “/filename”. Let's say we have 2 directories which are *dir1* and *dir2,* and a regular file *file1.* If *file1* is inside *dir2*, and *dir2* is inside *dir1*, and dir1 is in the root directory, so the path of *file1* is “/dir1/dir2/file1”.

Some example commands:

- showContent /j
- getHeight /milton
- *getNumberOfFiles /*
- *removeFile /*
- *traversePostOrderIteratively /milton*
- *traverseLevelOrder /milton/open*

Your Task

Some parts of the program have already been implemented for you such as the classes' methods inside *"fs"* package, so you don't have to deal with reading files or parsing their contents. What you need to do is to give the implementation for all the methods that throw *HaveToBeImplementedException* in **HW4.zip** file. You are free to introduce any **private** fields or **private** methods to the classes of these methods. However, <u>do not</u> remove or change the implemented methods or defined fields. Here below is the methods that you must provide implementations:

- *tree.Tree.getData()*
- *tree.Tree.setData(T data)*
- *tree.Tree.getParent()*
- *tree.Tree.setParent(Tree<T> tree)*
- *tree.Tree.getNumberOfNodes()*
- *tree.Tree.getHeight()*
- *tree.Tree.traverseLevelOrder(Consumer<T> visitingMethod):* While visiting call *visitingMethod.accept*
- *tree.Tree.traversePreOrderRecursively(Consumer<T> visitingMethod):* While visiting call *visitingMethod.accept*
- *tree.Tree.traversePostOrderIteratively(Consumer<T> visitingMethod):* While visiting call *visitingMethod.accept*

- *tree.GenericTree.addChild(Tree<T> child)*
- *tree.GenericTree.removeChild(Tree<T> child)*
- *tree.GenericTree.getChildren()*

- *tree.BinarySearchTree.getKey()*
- *tree.BinarySearchTree.setKey(K key)*
- *tree.BinarySearchTree.add(K key, T data)*
- *tree.BinarySearchTree.remove(K key)*
- *tree.BinarySearchTree.getLeftSubtree()*
- *tree.BinarySearchTree.getRightSubtree()*
- *tree.BinarySearchTree.getChildren()*
- *tree.BinarySearchTree.find(K key)*

Assignment Rules

- This is a <u>2-person group assignment</u>. However, inter-group collaboration <u>is not allowed</u>!
- All assignments are subject to plagiarism detection and the suspected solutions (derived from or inspired by the solution of other groups) will be <u>graded as zero</u>
- You are <u>not allowed</u> to define other than private methods or fields.
- You are <u>allowed</u> to use the following Java Collection classes
    - *java.util.List;*
    - *java.util.ArrayList*
    - *java.util.LinkedList*
    - *java.util.Queue*
    - *java.util.Stack*

- Keep your code clean. Avoid duplication and redundancy. Follow Java Naming Conventions. Use relative paths instead of absolute ones.

Submission Rules

All submissions must:

- be performed via Microsoft Teams by only one of the group members,
- be exported as an Eclipse Project and saved in ZIP format,
- include all necessary data files (if any TXT, CSV, JSON, etc.) in the right directory,
- follow a specific naming convention such that CENG112_HW4_groupID
    - Eclipse Project: CENG112_HW4_G5
    - Exported Archive File: CENG112_HW4_G5.zip

Submissions that do not comply with the rules above are penalized.