Dept. of Computer Science
Trent University

COIS 2020H: Data Structures and Algorithms

Assignment 3

Due Sunday, December 11, 2022 at 11:59pm.
No late assignments will be accepted.

Background

The common computer file system is like a tree where each node represents a directory that contains 0 or more files, and 0 or more links to subdirectories.

To implement such a file system, a data structure called the **leftmost-child, right-sibling** is often used. The leftmost child refers to the first subdirectory of a node, and the rightmost child refers to the next directory on the same level as the node (see Figure 1). In this case, the root directory has three files (P, Q and R) and three subdirectories (A, B and C). Note that directory names can be repeated if they do not appear as siblings. Similarly, file names can be repeated if they do not appear in the same directory.

Requirements

The primary task of this assignment is to design, implement, and test a file system using the leftmost-child, right-sibling as the data structure and where the behaviour of the file system is defined by the methods in Figure 2 (next page). Each method in FileSystem is worth 7 marks, except the constructor which is worth 3 marks.

Figure 1

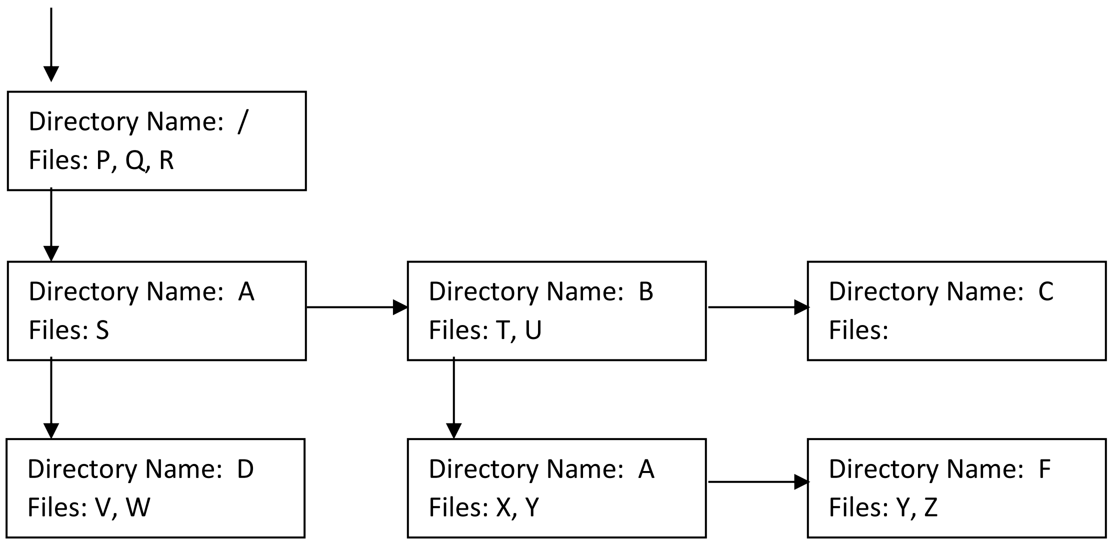

Figure 2

```
public class FileSystem
{
   private class Node
   {
       public string directory;
       public List<string> file;
       public Node leftMostChild;
       public Node rightSibling;
       ...
   }

    private Node root;

    // Creates a file system with a root directory where the name of the root directory is "/".
    public FileSystem( ) { ... }

    // Adds a file at the given address
    // Returns false if the file already exists at that address or the path is undefined; true otherwise
    public bool AddFile(string address) { ... }

    // Removes the file at the given address
    // Returns false if the file is not found at that address or the path is undefined; true otherwise
    public bool RemoveFile(string address) { ... }

    // Adds a directory at the given address
    // Returns false if the directory already exists or the path is undefined; true otherwise
    public bool AddDirectory(string address) { ... }

    // Removes the directory (and its subdirectories) at the given address
    // Returns false if the directory is not found or the path is undefined; true otherwise
    public bool RemoveDirectory(string address) { ... }

    // Returns the number of files in the file system (Do not add a count as a data member)
    public int NumberFiles( ) { ... }

    // Prints the directories in a pre-order fashion along with their files
    public void PrintFileSystem( ) { ... }
}
```

For each of the methods above, the address for a directory or file is given in absolute terms starting at the root node. For instance, the address of a directory or file is stated as:

/ADirectory/BDirectory/CDirectory
/ADirectory/BDirectory/CDirectory/AFile

The FileSystem class will also need to parse the given address to determine which path to follow in the file system.

Submission

Zip and submit your project (all files including the executable) as well as test results online at Blackboard (myLearningSystem). Only one partner needs to submit the assignment but do make sure that all names are on the submission.

Mark Breakdown

Node class	5
Parsing the address	10
Methods of FileSystem	45
Main Program	15
Testing	15
Inline documentation	10

Partners

The assignment is to be done in groups of 2 or 3. A discussion board has been set up to help you find team members. No solo assignments will be accepted.