CAS CS 112. Assignment 5

Due 11:59pm on Monday, November 7, 2011

Problem 1. (30 points) In this problem you will create code to manage a directory tree and provide the user with a basic subset of commands for doing so. We will model this assignment on the Unix command line[1].

There are two things to keep track of: the directory tree structure and the current directory that the user is in. Initially, there is only the root directory, which has the empty string for its name. The user may create subdirectories of the current directory (i.e., children) by issuing "mkdir" command; delete subdirectories by issuing "rmdir" command; and change the current directory to a subdirectory by issuing a "cd" command. In this assignment, these commands take a single argument, which is the name of the subdirectory. (In a real shell, they are more sophisticated.) In addition, "cd .." changes the current directory to the parent directory.

The user may also see all the children of the current directory by issuing the "ls" command, see the subtree rooted at the current directory by issuing the "tree" command, and see the path from the root to the current directory by issuing the "pwd" command.

Here is an example interaction:

```
> pwd
/
> ls
> mkdir cs
> mkdir math
> cd math
> mkdir calculus
> mkdir algebra
> mkdir geometry
> ls
calculus
algebra
geometry
> tree
math
  calculus
  algebra
  geometry
> cd ..
> pwd
/
> mkdir cs
Directory already exists
> cd cs
```

[1] If you are familiar with Linux or Mac OS command line, then you already know how to use these commands; if you are familiar with a Windows command line, then the commands are the same, except "pwd" on Unix is the same as "cd" without any arguments on Windows, and "ls" on Unix is the same as "dir" on Windows.

```
> mkdir cryptography
> mkdir graphics
> mkdir networking
> cd databases
No such directory
> mkdir databases
> ls
cryptography
graphics
networking
databases
> cd cryptography
> ls
> mkdir public-key
> mkdir symmetric-key
> tree
cryptography
  public-key
  symmetric-key
> cd ..
> tree
cs
  cryptography
    public-key
    symmetric-key
  graphics
  networking
  databases
> mkdir badLanguages
> cd badLanguages
> mkdir Basic
> cd ..
> tree
cs
  cryptography
    public-key
    symmetric-key
  graphics
  networking
  databases
  badLanguages
    Basic
> rmdir badLanguages
> tree
cs
  cryptography
```

```
    public-key
    symmetric-key
  graphics
  networking
  databases
> cd ..
> tree

  cs
    cryptography
      public-key
      symmetric-key
    graphics
    networking
    databases
  math
    calculus
    algebra
    geometry
> cd ..
No such directory
> exit
Goodbye.
```

Make sure your output matches *verbatim*. We provide you with user-interface code in SimpleShell.java. Implement and submit DirectoryTree.java.

Hints: modify the code we developed in class; in particular, include a parent pointer in each Node and a pointer to the current directory in the DirectoryTree class. Be careful with pwd: getting things to match verbatim is a bit tricky.

Problem 2. (70 points) In this problem, you will build the data structures necessary for analyzing words appearing in a certain text. The user interface (which you should not modify) is already provided for you in the classes `TextAnalyzer` and `TextAnalyzerGUI`. Be sure to understand it before proceeding. You may work with any text file for testing your code; will we eventually be testing it on Charlotte Bronte's Jane Eyre. Download `Jane Eyre.txt`, and be sure to put it into the appropriate directory so that it is easy to open. The two files you need to modify are `CountingTree.java` and `BSTWithDuplicates.java`. What to implement and how, and how much what is worth, is directly in those files. Study the the files before doing any coding.