CS 3450 – Design Patterns

**Program 6**

“Directory Traversal”

**Date Due**: See Syllabus

Write a program that simulates the directory listing commands of popular operating command shells. You will support the following commands:

* **list**
  + lists the entries in the current directory horizontally
* **listall**
  + prints a hierarchical listing of the current directory subtree (starting from the current node)
* **chdir <entry>**
  + changes directory to the named, adjacent subdirectory
* **up**
  + moves up the tree to the parent (like **cd ..**)
* **count**
  + prints the number of *files* (not directories) in the current directory
* **countall**
  + prints the number of files (not directories) in the subtree starting from the current node
* **q**
  + quit the program

You will begin by reading in a file that contains a directory tree. Here is a sample:

top:

file1

middle:

file2

file3

bottom:

file4

file5

file6

file7

another:

file8

Directories end with a colon. Print a prompt with the current directory name. Here is a sample session using the sample tree:

top> listall

top:

file1

middle:

file2

file3

bottom:

file4

file5

file6

file7

another:

file8

top> count

2

top> countall

8

top> chdir file1

no such directory

top> chdir middle

middle> list

file2 file3 bottom file6

middle> listall

middle:

file2

file3

bottom:

file4

file5

file6

middle> count

3

middle> up

top> list

file1 middle file7 another

top> chdir another

another> list

file8

another> up

top> chdir middle

middle> chdir bottom

bottom> list

file4 file5

bottom> ip

invalid command

bottom> up

middle> up

top> q

Needless to say, you’ll be reading into a composite structure.

Here’s a sample main program (but you don’t have to have an Explorer class):

int main() {

ifstream in("directory.dat");

Explorer exp(DirectoryFactory::createDirTree(in));

exp.process(cin, cout);

}

Explorer is a wrapper for the composite and interprets user commands and calls the right component methods, while keeping track of the current directory. Enjoy! (C++ programmers: beware memory leaks!)

**Design Note:**

1. This is obviously an exercise in using the Composite pattern and internal iteration. A good way to lose at least 15 points is to write or use a method that identifies the subtype of an object, such as isLeaf().
2. Do NOT use external iteration. (Don’t write an Iterator.) Unless you want to make it much harder than it has to be…
3. In the input file, subdirectories are indicated by series of three spaces.
4. Listall: when you are in a subdirectory, and you do a listall, start the current level at the left side.
5. Note that the data file I will use to test your program is MUCH more complicated than the sample given. Be sure to test your program thoroughly!!!