Lab 5: Using Stacks

In this lab, you'll use Python's lists as stacks to track an intruder's progress through a network to get some practice with the power of stacks.

The Scenario

Intruders have been detected within the network. Fortunately, we were able to log a full sequence of commands used during the intrusion. Here are three such logs: shortIntrusion (labs/shortIntrusion.log), intrusion1 (labs/intrusion1.log), intrusion2 (labs/intrusion2.log).

If you investigate these logs, you'll see that the intruder ssh-ed into a variety of machines, and executed a series of commands. In particular, the intruder used `less` to look at the contents of some files. We'd like to know which files on which machines were read.

For example, in shortIntrusion, the intruder first ssh-ed into a machine called `kelsi`, where he or she looked at the file `grapeful.doc`. The intruder then ssh-ed into a machine called `judi`, where he or she looked at two files, before exiting (putting the intruder back onto kelsi), looking at one more file, before exiting altogether.

For each file looked at by the intruder, we'd like to print out the sequence of machines through which the intruder accessed that file. For example, for shortIntrusion, the output would be:

```
$ python3 fileList.py shortIntrusion.log
kelsi->grapeful.doc
kelsi->judi->empyreumata.txt
kelsi->judi->acidophilous.html
kelsi->mounters.txt
```

As the intruder ssh-es into and out of machines, it might be difficult to keep track of which machine they are on... unless, of course, you have a stack.

The Assignment

Write a program called `fileList.py`which accomplishes the above. Here is a starter file (labs/fileList.txt). It should accept a log file as a command-line argument using sys.argv. **I have written a Stack class for you. You may only use this Stack class for storing data, do NOT use any python lists. You may use multiple stacks if you find it useful.**

Reading from files in Python: As a reminder, Python makes reading from files quite easy using the `with` operator. The below will print out every line of a file called `aFile.txt`:

```
with open('aFile.txt') as someHandle:
  for lineOfText in someHandle:
    print(lineOfText)
```

Extra Credit (110/100)

Display not only the file, but also its absolute path. For example,

```
ssh a
cd aDir
cd bDir
less aFile.txt
cd ..
ssh b
cd cDir
less bFile.txt
exit
less cFile.txt
cd dDir
less dFile.txt
exit
```

would give you...

```
a->~/aDir/bDir/aFile.txt
a->b->~/cDir/bFile.txt
a->~/aDir/cFile.txt
a->~/aDir/dDir/dFile.txt
```

Call this file `fileListExtra.py`.

Here's a test file (labs/pathintrusion.log).