Welcome to BST 281 Lab 6

8 Mar, 2018

Mike MacArthur

macarthur@g.harvard.edu

Office Hours: Fridays 2-3p

Kresge Student Lounge

Assignment 3 due on Friday March 16th by 11:59pm

Last office hours before due date are tomorrow (3/9)!

sys.argv vs stdin/stdout

sys.argv give a list of arguments passed with the script from the command line

The names of files passed through sys.argv are just names

To actually work with these files you'll need to open a file handle

stdin is a file object (or file-like object)

You can directly parse files passed via stdin

Examples

If we generate the following script and provide a sample input.txt file as the first argument and name a sample output.txt file as the second argument then we will parse the lines of the input file and write them to the output file

`python lab7.py sampleIn.txt sampleOut.txt`

```
import sys

print(len(sys.argv))
for arg in sys.argv:
    print(arg)

inFile = sys.argv[1]
outFile = sys.argv[2]

with open(inFile, 'r') as i:
    lines = i.readlines()

with open(outFile, 'w') as j:
  j.write(str(lines))
```

Notice that we need to open a file handle to access the contents of the file names in `sys.argv[1]`

Let's run the same workflow using stdin and stdout instead

Note that Windows users will have to run the following command directly in cmd, it will not work in the Atom Powershell

`python lab7_2.py < sampleIn.txt > sampleOut.txt`

```
import sys

for strLine in sys.stdin:
    print(strLine)
```

Another option is like this

```
import sys

lines = sys.stdin.readlines()
sys.stdout.write(str(lines))
```

What do you notice is different between the output files from the three different techniques?

Using stdin and stdout to manipulate CSV files

Let's use stdin and stdout to read and write a small example

`python lab7_3.py < sampleIn.csv > sampleOut.csv`

```
import sys
import csv

reader = csv.reader( sys.stdin )
writer = csv.writer( sys.stdout )

for row in reader:
    writer.writerow(row)
```