CSC 458 Data Mining & Predictive Analytics I, Spring 2021

Dr. Dale E. Parson, Assignment 1, Python data cleaning & regular expressions.

DUE By 11:59 PM on Saturday February 13, 2021 via make turnitin on mcgonagall or acad. The standard 10% per day deduction for late assignments applies.

This assignment uses a new dataset that is an aggregation of HTML files from several directories under https://faculty.kutztown.edu/parson/, along with some metadata tagged by a bash shell script.

From acad it will be necessary to **ssh mcgonagall** from acad in order to **make test**. You can also download and run the VPN and then putty or ssh into mcgonagall.kutztown.edu directly, although KU's VPN locks up some machines periodically. Testing must occur on mcgonagall because of increasing CPU load in these projects. If you are on campus, you can log directly into mcgonagall.kutztown.edu without going through acad. Also, you can perform file editing via acad, since acad and mcgonagall share the same student and faculty networked file systems. You can **make turnitin** at the end of the project from either machine. It is essential to use multiples of 4 space characters, not TAB characters, to indent your Python code. Python uses levels of indentation to denote blocks of code, and since you will be integrating your code into mine that uses 4 spaces, your code must conform. The course page contains instructions for setting up 4-space tabbing when you hit the TAB key for Notepad++ and vim.

Perform the following steps to set up for this project. Start out in your login directory.

```
cd  $HOME
mkdir  DataMine  # This may already be there.
cd  ./DataMine
cp  ~parson/DataMine/csc458spring2021py1.problem.zip  csc458spring2021py1.problem.zip
unzip  csc458spring2021py1.problem.zip
cd  ./csc458spring2021py1
```

This is the directory from which you must run **make turnitin** by the project deadline to avoid a 10% per day late penalty. **Please do not change the name of this directory**, since my test scripts depend on it.

You will see the following files in this **csc458spring2021py1** directory:

makecsc458spring2021assn1data.sh	Bash shell script used to aggregate raw data file.
htmlSpring2021rawdata.txt	Raw data file with some non-ASCII characters.
stripNonAscii.c	C program to clean out non-ASCII characters.
scanhtml.py	Python program you must complete.
FileLink.csv.ref	Correct output CSV file from Parson's supplied code.
HttpLink.csv.ref	Your code in scanhtml.py creates this CSV file.
makefile and makelib	Used for **make test** and **make turnitin**.

Running **make test** creates some additional files.

stripNonAscii	Compiled stripNonAscii.c.
clean_htmlSpring2021rawdata.txt	ASCII clean copy of htmlSpring2021rawdata.txt.
FileLink.csv	Parson's correct output CSV.
FileLink.csv.dif	Empty **diff FileLink.csv FileLink.csv.ref** file.

Here is what **make test** prints for the handout code.

$ **make test**
/bin/rm -f *.o *.class .jar core *.exe *.obj *.pyc __pycache__/*.pyc
/bin/rm -f junk* *.pyc stripNonAscii clean_htmlSpring2021rawdata.txt
/bin/rm -f *.tmp *.o *.dif *.out *.csv __pycache__/*
cc -o stripNonAscii stripNonAscii.c
./stripNonAscii -v < htmlSpring2021rawdata.txt > clean_htmlSpring2021rawdata.txt
OK, Filtering out non-ascii char at line 124 char offset 74
OK, Filtering out non-ascii char at line 126 char offset 72
OK, Filtering out non-ascii char at line 887 char offset 30
OK, Filtering out non-ascii char at line 1700 char offset 40
OK, Filtering out non-ascii char at line 4298 char offset 54
OK, Filtering out non-ascii char at line 5028 char offset 30
OK, Filtering out non-ascii char at line 10089 char offset 33
OK, Filtering out non-ascii char at line 10539 char offset 42
OK, Filtering out non-ascii char at line 10544 char offset 25
OK, Filtering out non-ascii char at line 10545 char offset 28
OK, Filtering out non-ascii char at line 11841 char offset 30
OK, Filtering out non-ascii char at line 15835 char offset 30
OK, Filtering out non-ascii char at line 15912 char offset 26
OK, Filtering out non-ascii char at line 15916 char offset 54
OK, Filtering out non-ascii char at line 15917 char offset 26
OK, Filtering out non-ascii char at line 16070 char offset 40
OK, Filtering out non-ascii char at line 16070 char offset 58
OK, Filtering out non-ascii char at line 19061 char offset 26
/bin/bash -c "PYTHONPATH=.:.. /usr/local/bin/python3.7 scanhtml.py clean_htmlSpring2021rawdata.txt FileLink.csv HttpLink.csv"
diff FileLink.csv FileLink.csv.ref > FileLink.csv.dif
diff HttpLink.csv HttpLink.csv.ref > HttpLink.csv.dif
diff: HttpLink.csv: No such file or directory
make: *** [test] Error 2

The completed PARSON code in scanhtml.py matches **<title>…</title>** and **href="file:** lines in clean_htmlSpring2021rawdata.txt like this:

6 <title>CSC 220CPVL, Object Oriented Multimedia Programming, Fall
7 2020</title>
141 href="file:////Users/parson/Library/Group%20Containers/UBF8T346G9.Office/TemporaryItems/msohtmlclip/clip_filelist.xml">

and creates entries in FileLink.csv like this, with one instance output line per **href="file:** input line:

datafile,title,titleLineNumInScan,titleLineNumInHtml,fileURL,fileLineNumInScan,fileLineNumInHtml
clean_htmlSpring2021rawdata.txt,"CSC 220CPVL, Object Oriented Multimedia Programming, Fall 2020",7,6,////Users/parson/Library/Group%20Containers/UBF8T346G9.Office/TemporaryItems/msohtmlclip/clip_filelist.xml,142,141

…

with the following attributes:

datafile	name of the aggregate data file being processed
title	string between the most recent <title> and </title> delimiters in the file
titleLineNumInScan	line number of most recent <title> in the data file being processed
titleLineNumInHtml	line number of most recent <title> in the originating HTML file
fileURL	URL of the href="file:, starting with the character after : to the closing ".
fileLineNumInScan	line number of href="file: being extracted in the data file being processed
fileLineNumInHtml	line number of href="file: being extracted in the originating HTML file

We will go over my supplied, working scanhtml.py code for extracting this data from the input file and writing it to CSV file FileLink.csv in class on January 26 and 28. Please take notes during class.

Your completed STUDENT code in scanhtml.py will match **LOCATION:** and **href="http:** OR **href="https:** lines in clean_htmlSpring2021rawdata.txt like this:

```
LOCATION: /Users/parson/kuweb/fall2020/CSC220Fall2020.html     8250 words
   61  href="https://learningtechnologysupport.kutztown.edu/support/solutions/folders/9000185752">Zoom
```

and create entries in HttpLink.csv like this, with one instance output line per **href="http:** or **href="https:** input line:

```
datafile,locationPath,locLineNumInScan,locNumberOfWords,httpURL,isSecure,httpLineNumInScan,http
LineNumInHtml
clean_htmlSpring2021rawdata.txt,/Users/parson/kuweb/fall2020/CSC220Fall2020.html,1,8250,//learning
technologysupport.kutztown.edu/support/solutions/folders/9000185752,1,62,61
…
```

with the following attributes:

datafile	name of the aggregate data file being processed
locationPath	HTML file path following LOCATION:
locLineNumInScan	line number of most recent LOCATION: in the data file being processed
locNumberOfWords	number of words per **wc -w** in this LOCATION's HTML file
httpURL	URL of the href="http: OR href="https:, starting with the character after : to the closing ".
isSecure	1 if https: else 0 for http:
httpLineNumInScan	line number of this href="http: or href="https: in the data file being processed
httpLineNumInHtml	line number of this href="http: or href="https: in the originating HTML file

STUDENT N % comments in scanhtml.py give your steps & grading rubrics. Make sure to read all STUDENT comments.

$ **grep 'STUDENT [0-9].*%' scanhtml.py**

```
# STUDENT 1 5%: Complete the above template 5 percent of assignment.
# STUDENT 2 30%: Create your patterns for these example lines here,
   # STUDENT 3 10%: You need similar variables to keep track of LOCATION's
   # STUDENT 4 15%: Create variables similar to fileLinkOutFile through
         # STUDENT 5 15%: if your LOCATION: pattern matches, retrieve its
         # STUDENT 6 15%: if your href="http: or https: matches, retrieve
   # STUDENT 7 10%: Close your output CSV file.
```

If you get an error with a .dif difference file like this on **make test**:

```
diff HttpLink.csv HttpLink.csv.ref > HttpLink.csv.dif
make: *** [test] Error 1
```

compare lines with the < arrow, which are your output, to lines with the > arrow, which are the correct output lines, to find the difference.

$ **grep '<' HttpLink.csv.dif | head -3**

```
< clean_htmlSpring2021rawdata.txt,/Users/parson/kuweb/fall2020/CSC220Fall2020.html,1,8250,//learningtechnologysupport.kutztown.edu/support/solutions/folders/9000185752,1,lineno,61
< clean_htmlSpring2021rawdata.txt,/Users/parson/kuweb/fall2020/CSC220Fall2020.html,1,8250,//kutztown.zoom.us/,1,lineno,131
< clean_htmlSpring2021rawdata.txt,/Users/parson/kuweb/fall2020/CSC220Fall2020.html,1,8250,//faculty.kutztown.edu/parson/index.html,0,lineno,874
```

$ **grep '>' HttpLink.csv.dif | head -3**

```
> clean_htmlSpring2021rawdata.txt,/Users/parson/kuweb/fall2020/CSC220Fall2020.html,1,8250,//learningtechnologysupport.kutztown.edu/support/solutions/folders/9000185752,1,62,61
> clean_htmlSpring2021rawdata.txt,/Users/parson/kuweb/fall2020/CSC220Fall2020.html,1,8250,//kutztown.zoom.us/,1,132,131
> clean_htmlSpring2021rawdata.txt,/Users/parson/kuweb/fall2020/CSC220Fall2020.html,1,8250,//faculty.kutztown.edu/parson/index.html,0,875,874
```

The diffs usually point you to the errors, and unless there are missing lines of output, there is a one-to-one correspondence between erroneous < lines in the program output and correct > lines in the reference file.

When you have completed coding and **make test** passes, make sure to insert your name and other project documentation at the top of scanhtml.py. Run **make test** a final time, then **make turnitin** and follow the prompt to turn in this project by the due date.

You can use my solution to FileLink.csv output as a template for many of your steps in this project.