Timed Lab 3 – Address Verifier

This is a Timed Lab; this Timed Lab is worth 25 **Exam & Timed Lab** points.

For this Timed Lab, you *may* use	However, you *may not*
• Course notes • Homeworks • Recitation assignments • Other course material • Any material you may find on the Internet that don't involve communicating "live" with other people.	• Communicate with other people/students in real-time via any means. This means no Facebook, email, Piazza, IM, IRC, cell phones, Google Talk, smoke signals, etc. • Share code with other students. • Look at other students work.

The TAs will be available to answer clarifying questions about the problem, but they are not permitted to give assistance with debugging code, program logic, etc. You will have an entire recitation period to work on this assignment; this time begins *exactly* when your recitation begins and ends *exactly* when your recitation ends: **No extra time will be given if you arrive late**, except in highly extenuating circumstances that must be approved by Dr. Summet or the Head TA.

T-Square will not permit any late submissions; ensure that you submit your code to T-Square several times to prevent earning a zero due to you being unable to submit. Your TAs will give a verbal warning 10 and 5 minutes before the end of the recitation period; you should submit at these times.

In your collaboration statement, if you use code from somewhere that is *not* a class resource (i.e. not listed on the course calendar), please list where this code came from. Ensure that you fill out the header at the top of the file.

Note that you must check out with your TA before you leave the recitation room. If you do not check out with your TA or you modify your submission after you leave the recitation room, **you will receive a grade of zero on the timed lab.** No submissions will be accepted after T-Square closes the assignment (i.e. it will not let you submit).

Problem Description:

You have been hired to analyze some data about properties in the area, based on information collected in random internet surveying. However, because this came from the internet, some people decided to enter bad data! Your job is to find the number of valid addresses in the webpage you have been given. Additionally, you have been asked to report your findings in a simple GUI so your management can use your program more easily. The website you have been given is http://www.cc.gatech.edu/classes/AY2013/cs2316_fall/hw/hopeulikit-031312.html. For all function headers below, if you choose to build your GUI in a class, you may modify the function headers as appropriate to make the functions work with your class.

getHTML

The purpose of the getHTML function is to take in one parameter, which is the URL of the website to download data from. You should connect to the webpage and download the data. You should return the contents of the website as a string.

getAddresses

This function should take in the string contents of the webpage as a parameter and return an integer representing the number of valid addresses on the webpage. Valid addresses are defined by the following pattern:
2-5 numbers followed by 2 or more words (Letters and spaces only) followed by a City, then a 2 letter state abbreviation, then a zip code. A zip code is exactly 5 digits 0-9. Some examples of a complete address:

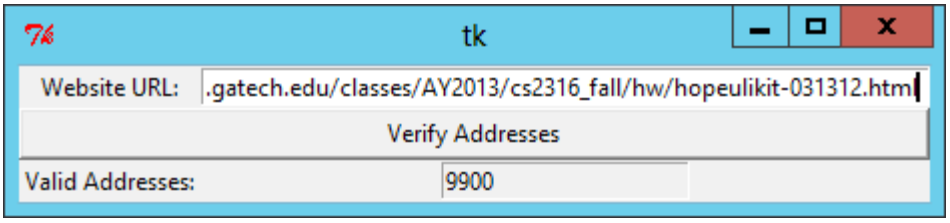
580 Turner Place NW
Atlanta GA 30332
1144 Bedford Rd
Hopeulikit GA 30461

HINT: In the HTML, there is a
 tag in addition to the **newline** (found directly before the
 tag) between the end of the street address and the beginning of the city and state. You may assume that this will be present in all addresses. Additionally, if you choose to use regular expressions to solve this problem, you would be well advised to copy and paste the sourcecode of the webpage into Regexr (http://www.gskinner.com/RegExr/) and use that to help you graphically build your regex. You may also want to think about how we matched everything that wasn't an HTML tag to help you deal with the newline character appropriately.

The GUI:

You should create a GUI that looks like the following when it is first launched:

The user will type in the address of the webpage containing the data, push the button, and receive a GUI that looks like the following:



(5844 is the correct number of addresses on the given website, 9900 above in the image is incorrect, but your code should work with any website with the same format as the given website). Note that the valid addresses entry box is in the read only state, and the website url entry box is set to width 60.

Grading:

getHTML

+2 – Correct function header.
+2 – Imports the correct module.
+2 – Downloads data from the correct source.
+1 – Correct return value

getAddresses

+2 – Correct function header
+1 – Has some code which attempts to match addresses.
+1 – Correctly identifies the house number
+1 – Correctly identifies the street name (Ferst Dr, Cherry St, etc.)
+1 – Correctly identifies city and state
+1 – Correctly identifies zipcode
+1 – Correctly identifies the
 and newline character
+1 – Finds the correct number of valid addresses.
+1 – Returns the number of valid addresses (correct or incorrect)

CS 2316 Fall 2012

GUI

+4 – GUI initially looks like the first screenshot
+4 – GUI looks like the second screenshot after the button is clicked.