CSE 101: Introduction to Computational and Algorithmic Thinking

Stony Brook University

Lab Assignment #11

Spring 2018

Assignment Due: April 20, 2018 by 11:59 pm

Assignment Objectives

This lab will give you some hands-on practice implementing basic data compression in Python.

Getting Started

Visit Piazza and download the "bare bones" file `lab11.py` onto your computer. Open `lab11.py` in PyCharm and fill in the following information at the top:

1. your first and last name as they appear in Blackboard
2. your Net ID (e.g., jsmith)
3. your Stony Brook ID # (e.g., 111999999)
4. the course number (CSE 101)
5. the assignment name and number (Lab #11)

Submit your final `lab11.py` file to Blackboard by the due date and time. Late work will not be graded. Code that crashes and cannot be graded will earn no credit.

Download Instructions

Download `lab11.zip` from Piazza. In the zip archive, you will find all the normal files as usual (template, driver). You will also find an extra folder called `test_files`. It is important you keep the folder as it is, and place the folder in the same directory with `lab11.py` and `lab11_driver.py`.

Part I: String Compressor (20 points)

In this problem, you will be implementing a basic data compression algorithm.

Write a function `compress_line()`, which takes one argument, `line`, a string that you will need to compress it.

The algorithm works like this: take a string and start reading from the beginning, compress identical **consecutive** characters in the form of *count of that character* directly followed by *that character*. In the end, the original string will be shortened (hopefully!), and the function should return that shortened result.

For example, the string `'AAABBBCCC'` will be shortened to `'3A3B3C'`; the string `'AAC!!?????'` will be shortened to 2A1C2!5?; and the string '   @@@@B ' will be shortened to '3 4@1B2 '. Notice thata space is also considered a character.

Note: You may assume that only valid strings will be given as an input for the function. Also, it is **very** important that you do a .strip('\r\n') on every string. This is because the invisible newline/return characters may cause issues when you do Part II of this assignment.

Examples:

Function Call	**Return Value**
compress_line(' PPPPPPPPggggggggggffff+++n M\r\n')	2 8P10g4f3+1n5 1M
compress_line(' PwwwwwwRRRRRR\r')	3 1P6w6R
compress_line(' hhhhhhhhhhcccccccFFFFFFFFFvvvvdddd ttttttttt\r\n')	3 10h7c9F4v4d8 9t
compress_line(' mmmmmmmm ???? ---------- Uwwww <<<<TTTTTTTTT SSSSSSSS FF \n')	3 8m7 4?1 10-2 1U4w3 4<9T4 8S4 2F7
compress_line(' ######EEEE [[[[[[[S::::::: ')	3 6#4E7 7[1S7:7
compress_line('....xx ;;;;;;;;;999922 }}}}}}}}}cccccKKK 3\r\n')	4.2x4 9;49227 9}5c3K8 13
compress_line('II4*********)))))))vvvvvvvLLLLLLLLLpp]]]]]]]\n')	2I149*7)7v9L2p3 7]
compress_line(' XXX ddssPPPPPP\r')	1 3X7 2d2s6P
compress_line('((((((((((HHHHH ????????.... FFUUUUUUUUUU LLLLLL\n')	10(2 5H2 8?4.3 2F10U7 6L
compress_line(' 8888 yyyyyyyyyy,,,,,,, l2222222\n')	2 485 10y8,7 1l72
compress_line(' gggggggmmmmmm&&&&&&33333 yyyyyyyyiiiiiii\r\n')	1 7g6m6&531 8y7i
compress_line('QQQ000 [[[;;::::CCRRRRRRR%%% \r\n')	3Q302 3[2;4:2C7R3%3
compress_line(' ggggggggggwwwwwwwwJJJJ\r\n')	2 10g8w4J
compress_line('zzzzzFFFFU (( ))))))XXXXXX psssssssss VVVVVVVVV\r')	5z4F1U2 2(5 6)6X6 1p10s8 9V
compress_line(' H w RRRRRRRR\n')	1 1H6 1w3 8R
compress_line(' PPPPPPPPPPjjjjjj IIIIllllllll\r\n')	1 10P6j1 4I8l
compress_line(' +++++$$$$ zzzzzzzl88888888888\r\n')	2 5+4$3 7z1l108
compress_line(' WWWWWWW dddddd !!! ......VVVVVV wwwwww>>>>>>>\n')	3 7W6 6d1 3!3 6.6V6 6w7>
compress_line(' zz\{oo\n')	1 2z1\{2o

Part II: File Compressor (20 points)

Write a function compress_file(), that takes one argument, filename, which is the name of the file you need to compress.

In this part, you should read the file line by line, compress each line as described in Part I, and append each compressed line into a list. In the end, you should return the result list that consists of all the compressed lines, in the correct order. You should call the function you wrote in Part I to compress each individual line for you.

Examples: Due to very long outputs, please consult the lab driver for examples.

Note: If you use MacOS, you may find the format getting messed up when you open one of the test files. Try to open the `.txt` files in Pycharm itself or some kind of text editor like Atom, and it should look correct.

How to Submit Your Work for Grading

To submit your `.py` file for grading:

1. Login to Blackboard and locate the course account for CSE 101.
2. Click on "Assignments" in the left-hand menu and find the link for this assignment.
3. Click on the link for this assignment.
4. Click the "Browse My Computer" button and locate the `.py` file you wish to submit. Submit only that one `.py` file.
5. Click the "Submit" button to submit your work for grading.

Oops, I messed up and I need to resubmit a file!

No worries! Just follow the above directions again. We will grade only your last submission.