

COMP 171

Scripting Language Practicum

Dr. William L. Honig

Loyola University Chicago

Heavy Strings- Formatting

Overview

Learn to work with text formatting in Python. Explore different formatting options.

Note: formatting works on strings so you can format any string variable, not only things that are printed out.

What you need to do

Follow these instructions fully. You will write a program that reads a file, processes each line of the file and print out the results line by line after formatting them.

1. Inside your script open (for reading) the file InputToFormat.txt provided with the assignment.
2. Read and format each line one at a time, then print out the results
 - a. Use `file.readline()`
 - i. *Don't* use a for loop to read the file – you need to do something different with each line
 - b. Call the first line 0
 - c. After you read each line, use `.strip` to trim off the ending newline character
 - d. Use `.split` to make a list from the line that was read
 - i. There may be different numbers of things on different lines
 - e. Use `.format` to put the input in the correct format (see below)
 - i. Each item will be one of your list elements
 - f. Print out the result after formatting each line
3. Format each line as follows before printing it out (you may need to adjust the type of the items to something other than string to make it work)
 - a. Line zero: center the item in 50 characters
 - b. Line one: the first item without formatting, then four items centered in 12 spaces, the fifth item left adjusted
 - c. Line two: print line number right adjusted in 5 spaces, the first four items as integers centered in 12 spaces with commas in the normal places, the fifth item left adjusted

- d. Line three: print line number right adjusted in 5 spaces, the first four items as integers centered in 12 spaces with commas in the normal places and plus or minus signs, the fifth item left adjusted
 - e. Line four: print line number right adjusted in 5 spaces, the first four items as floats centered in 12 spaces with commas in the normal places, two numbers after the decimal point a minus sign for negatives but no sign for positives, the fifth item left adjusted
 - f. Line five: print line number right adjusted in 5 spaces, the first four items centered in 12 spaces with commas in the normal places, two numbers after the decimal point and with a sign for all numbers, the fifth item left adjusted
 - g. Line six: print line number right adjusted in 5 spaces, the first four items centered in 11 spaces with no commas, two numbers after the decimal point and with a sign for all numbers and leading zeros so that all numbers take the full space (zero padding), the fifth item left adjusted. Be sure the padding zeroes are after the sign. Put a blank space between each column.
 - h. Line seven: print line number right adjusted in 5 spaces. The line contains two numbers and a comment. Print out <first number> is <percentage> of <second number> and then the comment in column 5. Have the percentage displayed with one decimal point and the other two numbers with two decimal points. Make an arrangement that looks good with the other lines in the print out (see the example below).
 - i. Line eight: print line number right adjusted in 5 spaces. The line contains 4 words and a comment. Center each word in the 4 columns after the line number and the comment as usual right adjusted at the end. Make them line up well with the above numbers.
 - j. Line nine: print line number right adjusted in 5 spaces. The line contains 4 words and a comment. Center each word in the 4 columns of 11 characters and fill space with a + character. Leave one blank between filled columns. The comment as usual right adjusted at the end. Make them line up well with the above numbers.
 - k. Line ten: print line number right adjusted in 5 spaces. The line contains 4 words and a comment. Center each word in the 4 columns of 12 characters wide but limit the words to 11 characters at most. The comment as usual right adjusted at the end. Make them line up well with the above numbers.
- 4. Once all the lines are processed and printed out, print out another line saying "I'm done formatting!" centered in 50 characters.
 - 5. Write one or two lines at a time and try them out. Don't try to do this all at once.

6. Do not make any changes to the input file `InputToFormat.txt`. I'll be running your program with my original file.
7. See the `StringFormattingExamples.py` example for formatting to use.
8. See the Python documentation from the links in the class web page for more help. Start with the Python Standard Library, especially sections 6.1.3, 6.1.3.1, and 6.1.3.2
9. Save your final program as a `.py` file

How and what to turn in

Turn in your `.py` file. You must be on time to get points.

Grading

20 points in Assignments for a `.py` file that runs in Python 3 and has all the necessary items. -5 for each missing item; -2 for minor errors in formatting. 0 points if your program has errors at compile or run time.

Example of solution print out:

(see next page)

V2.0 (correct strip and formatting updates) Spring 2012

V1.0 (original) Spring 2012

©W.L.Honig Loyola University Chicago Spring 2012 Comp 171

```
StringFormattingStarter.py - W:\FromSonySync\LUCDocuments\Courses\Comp171Materials\Programs\StringFormattingStarter.py
Python Shell
File Edit Shell Debug Options Windows Help
Python 3.1.2 (r312:79149, Mar 21 2010, 00:41:52) [MSC v.1500 32 bit (Intel)] on win32
Type "copyright", "credits" or "license()" for more information.
>>> ===== RESTART =====
>>>
>>> formatAssignment()
My_Work_With_Python_Formatting
Line# Column1 Column2 Column3 Column4 Comments
2 567,890 -4 23 -54,321 Integers
3 +567,890 -1,000 +23 -54,321 Integers_With_Signs
4 128.00 -128.00 99.99 100.00 Float_Numbers
5 +128.00 -128.00 +99.99 -91.11 Float_Numbers_with_Signs
6 +0000128.00 -0000128.00 +0000099.99 +1234567.89 Float_Numbers_with_Signs_Leading_Zeros
7 253.00 is 24.6% of 1028.00 Percentage_Calculation
8 Twas brillig and slithy Words
9 +++++Did++++ ++format+++ ++ever++++ ++well++++ Filled_Words
10 Aegilops Overnumerou depolarizab Nonsupports Long_Words
I'm done formatting
>>>
def
```

Version 1.0 (original)

©W.L.Honig, for Comp 171 Scripting Practicum, Spring 2012