Week 3: Assignment

Submit Assignment

Due Jul 21 by 11:59pm **Points** 100 **Submitting** a file upload

Instructions

Please make a python file called week_3_homework.py and put the answers to the problems below in it. All the problems below ask you to write functions. Put ***only*** the functions in the file "week_3_homework.py" do not put any testing code in that file. To test your functions, create another file called week_3_homework_test.py. In this file you can import that functions from your homework file as follows:

```
import week_3_homework as w3h

# then we can test the functions like so:

result = w3h.my_function()
print(result)
```

Please write the following functions. ***Make sure you name the functions exactly as typed below.***

Problem 1

Write a function called **all_the_kwargs**. This function will accept all the keyword arguments that are passed to it and then return the number of keyword arguments that are passed. Displaying the function signature would give the answer away, so I won't do that, but this is straight out of the lectures and is simple - don't overthink it. Please be sure to add a docstring to the function.

You can test the function with the code below.

```
result = all_the_kwargs(my_kwarg='this', second_kwarg='that',
some_number=1)
print(result) # result should be 3.
```

Problem 2

Write a generator function called **almost_fibonacci**. This function will accept one argument, N, and it will produce the first N numbers of a sequence like the fibonacci sequence except that instead of the next number in the sequence being the sum of the previous two numbers, the next number in the sequence will be the sum of the previous *three* numbers. The first three numbers in the sequence are 0, 1, 1. The function signature will be **almost_fibonacci(N)**. Please be sure to add a docstring to the function.

You can test it with the following code:

```
for n in almost_fibonacci(10):
    print(n)

# the following number should print: 0, 1, 1, 2, 4, 7, 13, 24, 44, 81
```

Problem 3

Write a generator function called **first_word_of_each_line**. This function will be very similar to the word count function we went over in lecture, except that, instead of returning a word count for each line, it will just return the first word of each line. The function will take one argument, and this will be the filepath to the file it will read through. Be very careful to not hard code this function to look at a specific path on your computer. The function signature will be **first_word_of_each_line(filepath).** Please be sure to add a docstring to the function.

To test this function, create a plain text file and write several lines in it and then pass the filepath to this file to your function. Loop through the generator, printing the values (just like we do for the problem above) and see if the function is returning the first word of each line in the file.

Submission

Submit the file with the functions, week_3_homework.py, to the homework page on the course website. Each problem is worth 30 points, and you get 10 points for turning something in (that is completely blank). Click on the blue button in the top right corner to submit your assignment.

Click Next (below) to progress through the course.