CS 1110, LAB 3: STRING EXPRESSIONS

http://www.cs.cornell.edu/courses/cs1110/2017fa/labs/lab3/

First Name: _____ **Last Name**: _____ **NetID**: _____

The purpose of this lab is to help you to better understand strings and user-defined functions. One of your primary tasks in Assignment 1 will be writing functions that take a string as argument and return a string when done. This assignment will go a long way in helping you understand how to work on Assignment 1.

This lab will be similar in style to the previous lab. There are several tables that you will need to fill in the values for. These tables will help you understand string methods and expressions. In addition, we will ask you to create another module. This module will be a bit more involved than the one from last time.

If you have never programmed before, you may find the last part of this lab to be significantly harder than any of the previous labs. If you find yourself struggling, we strongly encourage you to sign up for one of the **One-on-Ones** announced in class.

Getting Credit for the Lab. Once again, you have a choice between getting credit through the online system or your instructor. The online lab is available at the web page

http://www.cs.cornell.edu/courses/cs1110/2017fa/labs/lab3/

The advantage of the online system is that you can do it on your own time and verify that you got credit. The disadvantage is that your answers must be correct. If you want more guided help on this lab, you should use the worksheet instead. Despite the demands of the online system, labs are graded on effort, not correctness.

If you use this worksheet, your answer will include both a sheet of paper (or the sheet provided to you in lab) and file called `lab03.py`. This is a file that you will create frome scratch. When you are finished, you should show both this file and your written answers to your lab instructor, who will record that you did it.

This lab is long enough that you may not finish during class time. If you do not finish, you have **until the beginning of lab next week** to finish it. Over the next week, you may either (1) complete the lab online, (2) show your lab to a consultant during consulting hours, or (3) show your lab to an instructor at the *beginning* of the next lab session. The online version will stop receiving submissions after the last lab on Wednesday.

Course authors: D. Gries, L. Lee, S. Marschner, W. White

1. String Expressions

This section should be very familiar to you by now. The first table challenges you to evaluate an expression. The second table asks you to "fill in the blank", completing an expression to give you the provided value. This time the blanks will all either be a string or number literal.

Throughout this section, pay close attention to spaces and to the different types of quotation marks being used. We use both ' (single quote) and " (double quote). While both work, there are reasons to use one over another.

Expression	Calculated
`'CS '+'is '+'fun'`	
`'CS'+'is'+'fun'`	
`"CS"+"is"+"fun"`	
`"CS"+('is'+"fun")`	
`'Double "'`	
`'Single \''`	
`'Single '''`	
`'' + 'ok'`	
`'' + '4 // 2'`	
`'' + 4 // 2`	
`'' + str(4//2)`	
`'' + repr(4//2)`	

Expression	Calculated	Missing
`'Hello '+` ☐	`'Hello World'`	
`'Hello'+` ☐	`'Hello World'`	
`"Hello "+` ☐	`'Hello World'`	
`"A"+(` ☐ `+"C")`	`'ABC'`	
`'A'+` ☐ `+'B'`	`'A"B'`	
`'A'+` ☐ `+'B'`	`"A'B"`	
`'A'+` ☐ `+'C'`	`"A'B\"C"`	
`''+` ☐	`''`	
`''+` ☐	`'2+2'`	
`'4'+` ☐	`'4 // 2'`	
`str(` ☐ `+ 2)`	`'4'`	
`repr(` ☐ `* 3)`	`'12'`	

2. String Methods

Strings have many handy methods, whose specifications can be found at the following URL:

http://docs.python.org/3/library/stdtypes.html#string-methods

For right now, look at section 4.7.1 "String Methods," and do not worry that about all the unfamiliar terminology. You will understand it all by the end of the semester.

Using a method is a lot like using a function. The difference is that you first start with the string to operate on, follow it with a period, and then use the name of the method as in a function call. For example, the following all work in Python:

```
s.index('a')          # assuming the variable s contains a string
'CS 1110'.index('1')  # you can call methods on a literal value
s.strip().index('a')  # s.strip() returns a string, which takes a method
```

Before starting with the table below, enter the following statement into the Python shell:

```
>>> s = 'Hello World!'
```

Once you have done that, use the string stored in `s` to fill out the tables below, just as before.

Expression	Calculated
`s[1]`	
`s[15]`	
`s[1:5]`	
`s[:5]`	
`s[5:]`	
`'e' in s`	
`'x' in s`	
`s.index('e')`	
`s.index('x')`	
`s.index('l',5)`	
`s.find('e')`	
`s.find('x')`	
`s.count('o')`	

Expression	Calculated	Missing
`s[ ]`	`'W'`	
`[ ][2]`	`'C'`	
`s[4: ]`	`'o Wo'`	
`s[: ]`	`'He'`	
`s[ :]`	`'rld!'`	
`[ ] in 'ABC'`	`True`	
`'e' in [ ]`	`False`	
`s.index( )`	`4`	
`[ ].index('x')`	`2`	
`s.index('o', )`	`7`	
`s.find( )`	`4`	
`[ ].find('e')`	`-1`	
`s.count( )`	`3`	

3. Writing Your First Function

Up until now, we have only been working with the functions provided by Python. Now it is time to create your own. You are going to create a function that will be very useful in Assignment 1.

The purpose of this function is to extract a substring from inside of another using string slicing. The substring that you want will be inside of double quote characters. For example, suppose you have the following string:

```
'The phrase, "Don\'t panic!" is frequently uttered by consultants.'
```

If you provide this string as an argument to the function, it should return the string

```
"Don't panic!"
```

To get started, make a file called `lab03.py`. Like last week, you should start the file with a document string that has four lines.

(1) The first line should say `'The module for first_inside_quotes'`
(2) The second line should be blank
(3) The third line is your name and net-id
(4) The last line is today's date.

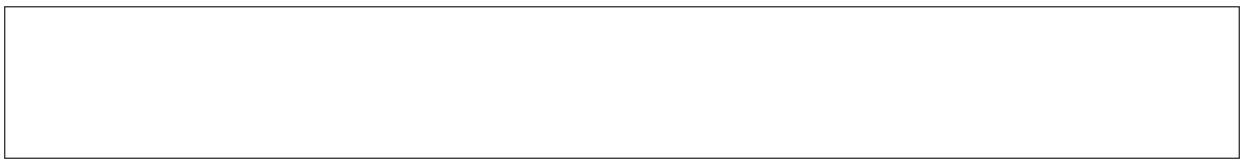
Next, you will create the function stub. A stub is a function definition that has the header but no body. This allows you to call the function, but nothing happens when you call it. Add following the function stub to your file:

```
def first_inside_quotes(s):
    """
    Returns: The first substring of s between two (double) quote characters
    A quote character is one that is inside a string, not one that delimits it.
    We typically use single quotes (') to delimit a string if want to use a double
    quote character (") inside of it.

    Example:  If s is 'A "B C" D', this function returns 'B C'
    Example:  If s is 'A "B C" D "E F" G', this function still returns 'B C'
    because it only picks the first such substring.

    Parameter s: a string to search
    Precondition: s a string with at least two (double) quote characters.
    """
    pass
```

Before you do anything else, let us test that it is working. Put `lab03.py` in a folder, and navigate to that folder, like you did in the last lab. Start the python interactive shell and type the following:

```
>>> import lab03
>>> lab03.first_inside_quotes('A "B C" D')
```

What happens?



You should now replace the word `pass` (which was just a placeholder) with a sequence of assignment statements that slice out the part of the string inside the double quotes. The last line of the body should be a `return` statement with the final value.

If you are unsure of what to do, review the slides for Lecture 5. There are two examples in that lecture that are very similar to this lab. That is all the guidance we are going to give. However, if you are struggling, please call over a staff member to help you.

4. Creating Test Cases

To make sure that the function is working properly, you need to test it. The way you test a function is as follows:

(1) Choose a sample input for the function.
(2) Read the specification (the part in """) to determine what the output should be.
(3) Call the function with your input as argument and look at the output.

If you output you get in (3) is the same as what you guessed in (2) then the test worked correctly. Otherwise, there is something wrong with your function definition and you need to fix it.

Let us start with the example test that we tried before you wrote function body. The input was `'A "B C" D'`. Reading the specification, we see that the function should return `'B C'`. To try out this test, make sure that you are in the same folder as `lab03.py` and start the Python interactive shell. They type

```
>>> import lab03
>>> lab03.first_inside_quotes('A "B C" D')
'B C'
```

If you see the `'B C'` after the function call, then your function is working correctly (so far). If not, call over a staff member to help you fix your function.

As we will see in the next lecture, one test is not enough. Try to come up with three different tests for this function. All you need for a test is an input and an expected output. We will not grade you on how good your tests are, as that is the subject of the next lecture. Right now, we just want you to make some tests. List them below.

Input	**Expected Output**