Lab 12: Welcome To Python

Instructions:

This worksheet serves as a guide and set of instructions to complete the lab.

- You must use the starter file, found here, to get credit for the lab.
- Additionally, here is the workbook that you can read through for further context and additional (non-required) material.
- All material was sourced from the CS10 version of The Beauty and Joy of Computing course.

Notes:

As we begin working with Python, it's totally normal to have questions about getting your code up and running. We know it can take some time to get comfortable, and you might get stuck or need help. Before asking a logistical question, PLEASE check the workbook first. It includes step-by-step instructions on using the terminal, opening a shell, running Python, and defining functions. You can also find guidance here for setting up Python on your computer.

Submitting:

You will need to complete Exercises 1-4 and submit the file to Gradescope.

- To receive full credit, you will need to complete the required functions, and the required functions must pass all tests from the autograder in Gradescope.

Please note, you must use the starter file, and you must NOT edit the name of any of the required blocks. Failing to do either for these will result in the autograder failing.

Objectives:

We have learned many topics during our weeks together such as iteration, algorithms, recursion, conditionals, and more. We have practiced and implemented these ideas primarily through Snap. In this lab, you will apply the more simple topics you have learned so far in the course in another programming language, Python. By the end of lab you will:

- Know how to start/open a file in a text editor
- Know how to run your python file through terminal
- Apply pre-existing knowledge of topics in Python
- Create working functions in Python
- Understand the basic syntax of Python

Required Functions:

- Function 1: sum_all_numbers(x, y)
- Function 2: exponent(num, power)
- Function 3: palindrome(string)

- Function 4: reverse_string(string)

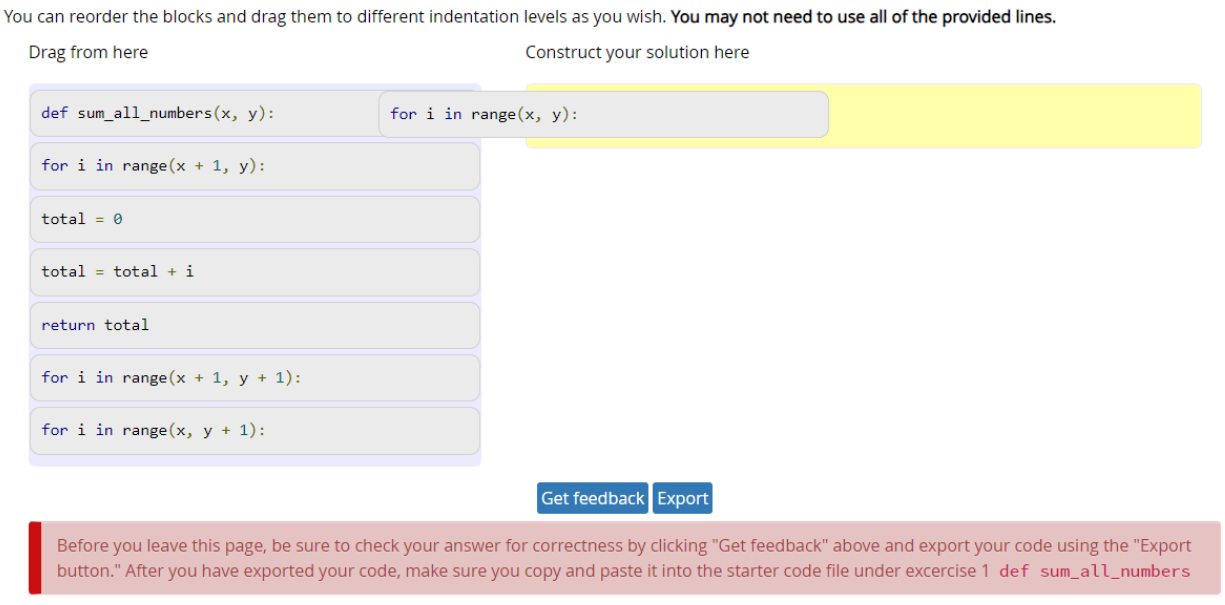
Important Topics mentioned in the Workbook:

For better understanding of the lab we highly recommend going through these workbook pages!

- Introduction to the Terminal
- Variables and Primitive Expressions
- **Run a Python Script**
- Defining A Simple Python Function
- Print vs Return
- While Loops
- Basic String Manipulation
- Conditionals

Function 1: sum_all_numbers(x, y)

- Objective:
  - Create a function that adds all the numbers between (and including) the two inputs, x and y.
- For this exercise, IT IS REQUIRED that you create the correct function on the Workbook and either export/copy/retype the exact solution into your file



  - *Please refresh the page if you can not see these blocks/instructions*
- Notes:
  - Use of sum() function is prohibited
- Inputs:
  - x = any number
  - y = any number
    - y > x
- Output:

    - a number
    - Reports the sum of all numbers between (and including) x and y.
- Examples:
    - Included in starter file
        - We call these “doctests”

Function 2: exponent(num, power)

- Objective:
    - Create a function that calculates the input num with an exponent of the input power
    - Calculate num^power
    - The following method has to be used as above. Click, drag, and rearrange the code in the workbook
- Notes:
    - Use of pow() or ** function is *prohibited*
- Inputs:
    - Num = any number
    - Power = any positive number
        - Power >= 0
- Output:
    - a number
    - Reports Num to the power of Power
        - num^power

- Examples:
    - Included in starter file
        - We call these “doctests”
    -

```
>>> exponent(10, 0)
1
>>> exponent(5, 3)
125
>>> exponent(2, 10)
1024
```

Function 3: palindrome(string)

- Objective:
    - Create a function that confirms or denies if a string is a palindrome
        - The same code blocks can be found on the workbook

Drag from here

```
elif string[0] != string[-1]:
return False
return True
if len(string) < 2
elif string[0] != string[1]:
if len(string) < 2:
return palindrome(string[1:-1])
def palindrome(string):
return palindrome(string[0:-2])
return palindrome(string[1:-2])
```

- Notes:
  - A palindrome is a word, phrase, or sequence that reads the same backward as forward
    - E.g. tacocat, racecar, noon
  - *You must use recursion to write this block. Loops and list comprehension are not allowed.*
- Inputs:
  - String = any string
    - Usually a word
- Output:
  - boolean
  - Reports whether the string is a palindrome

- Examples:
  - Included in starter file
    - We call these “doctests”

Function 4: reverse_string(string)

- Objective:
  - Create a function that reverses the string you input
  - For this problem, instead of dragging and dropping code in the browser, you will write code free form using a text editor.
    - *To do so, open up the starter code and proceed to exercise 4 def reverse_string(string). If you need help, refer to this page or ask a staff member for help!*

- Inputs:

        - String = any string
            - Usually a word
- Output:
    - a string
    - Reports the reversed version of your input string

- Examples:
    - Included in starter file
        - We call these “doctests”

To use the autograder type: python3 -m doctest lab12-starter-code.py in your terminal
-alternatively- **python3 -m doctest <labfilename>.py**

```
PS C:\Users\train\downloads> python3 -m doctest lab12-starter-code.py
```

Note you must be in the correct parent file! + You must save.

Failing tests like this:

```
**********************************************************************
4 items had failures:
   5 of   5 in lab12-starter-code.exponent
   6 of   7 in lab12-starter-code.palindrome
   4 of   5 in lab12-starter-code.reverse_string
   7 of   7 in lab12-starter-code.sum_all_numbers
***Test Failed*** 22 failures.
```

With specific doctests showing where your error lies. If the code runs and enters a new line with no failures, then you have passed the autograder.

Remember to submit on Gradescope!