1 Objectives

- Practice using string methods.
- Practice using for loops over strings.

2 Driver and navigator

As always, in this lab you may work on your own or with a partner. You and your partner will take on distinct roles: **driver** (the person typing at the keyboard) and **navigator** (the person watching for mistakes, and thinking ahead). You should switch roles after every question below.

3 String Operators and Methods

Complete the tasks below using Python’s operators or `str` methods in the shell. Use `dir(str)` and then `help`, either on `str` or on a specific string method, to get help on the various string methods that exist.

Please do not submit any code you write in the Shell on MarkUs. This is for your practice and as a warm-up for the sections that follow. If you are unsure about anything in this section, please ask your lab TA.

1. Calculate the length of the string `'supercalifragilisticexpialidocious'`.
2. Check if string `'or'` is a substring of string `'Hello world!'`.
3. Find the number of occurrences of the letter `'a'` in `'banana'`.
4. Convert `'welcome to csc120'` to upper case.
5. What does `welcome to csc120 .rstrip()` do?
6. What does `'123'.isdigit()` return?
7. What does `'ABC123'.isalpha()` return?
8. What does `' ' in "october"` evaluate to?
9. What does the expression `table > 'task'` evaluates to?
10. How do you return a new string with all occurrences of 'fall' switched with 'spring' in "The leaves fall in fall."
11. What do you print the following message, including the double quotes: She said: "You can make it!"
12. Print the following using a single call to function `print`:

   1st line

   4th line

   Note the two empty lines between the first and second phrase.
4 Basic Iteration

Switch roles, so \texttt{s2} is the driver and \texttt{s1} is the navigator.

In this section, you will practice writing simple for loops in the Shell, and using function \texttt{print}. Please note, that you will not be submitting this code on MarkUs either. If you have any questions, ask your TA in the lab.

Working in the Python shell, begin by assigning the string “superconductivity” to a variable \texttt{s}.

Now do these exercises:

1. Write a loop that prints each character in \texttt{s} on a separate line.
2. Write a loop that prints each character in \texttt{s} on the same line with a space after each character. (Hint: \texttt{help(print)}).
3. Write a loop that prints each character in \texttt{s} on the same line, but with a comma and a space after each character: \texttt{s, u, p, e, r, c,} etc. Notice that there is a comma and a space after the last character. We won’t worry about this for now.

5 Functions using for loops over strings

If you are working with a lab partner, Switch roles, so \texttt{s1} is the driver and \texttt{s2} is the navigator. In this section, you should switch roles in between each function that you write.

This section asks you to write the code for the three functions specified below. There are some important aspects to these exercises:

- Each of these functions should include a \texttt{for} loop and an \texttt{if} statement.
- For each of these functions, you should write a docstring first, including some examples (at least two). Run \texttt{doctest} for each function you write, once you’ve implemented it. Recall that to do so you’d need to \texttt{import doctest} at the top of your file. Then at the bottom you can type:

  ```python
  if __name__ == '__main__':
      doctest.testmod(verbosity=True)
  ```

The description of each function is listed below. All parameters in the functions below are of type \texttt{str}.

- \texttt{count_digits(word)}:
  - Description: Return the number of digits present in word.
  - Example: \texttt{count_digits('CSC120-Fall17')} should return the int 5.
  - Hint: You will need to use a numerical accumulator as we did in class.

- \texttt{first_digit(word)}:
  - Description: Return the first digit present in word, or the empty string if no digit is present.
  - Example: \texttt{first_digit('CSC120')} would return a string ‘1’.
  - Clarification: The return type of this function should always be \texttt{str}. 


• `find_uppercase(word):`
  – Description: Return a string containing all uppercase characters in word (and in the same order).
  – Example: `find_uppercase('PYthon3 is fuN!')` would return the string 'PYN'.
  – Hint1: There is a string method that helps you find out if a cased character is uppercase or not.
  – Hint2: You would need to use a variable as a string accumulator. This variable should initially refer to the empty string.

### 6 Declaring Lab Partnership on MarkUs and Submitting

As in the previous lab, you will need to declare, on MarkUs, whether you are working alone or with a partner before you are allowed to submit. Note that you will need to repeat this process for every lab; this is to allow you to switch lab partners or work alone if you choose to.

See Lab2 handout about the two-step process involved in declaring a partnership.

Once your partnership has been declared, you are ready to submit your work. Although only one partner should submit the assignment, because you declared your partnership, we will know that both of you should get credit for the work – no matter who submits it.

For this lab, you need to submit all the code you wrote for Section 5 in a file named `lab4.py` to MarkUs by Friday October 6th at 9:59pm. Your function names must be exactly as written above. Recall that Python is a case sensitive language. Your functions should not contain any print statements. You should have a single main clause with a call to doctest. All your docstring examples should pass doctest.