Assignment 5

Start Assignment

Due No Due Date **Points** 10 **Submitting** a file upload **File Types** zip

This week's assignment is a continuation of last week's work, you will be practicing using different UI elements in HTML with a JavaScript program to perform some operations on user input. You'll create a cryptogram generator, which takes any message as input and outputs a cipher message in a few different possible outputs.

You are provided with a **starter.zip** (https://csus.instructure.com/courses/97428/files/14970118/download?download_frd=1) with starting `HTML` for you to use and build upon. There is some CSS, but you can add your own.

Part I: Write the start of `encrypt-it.js`

Write JavaScript testing code that logs a message to the console. This is just a test to make sure that your browser is running your JavaScript file.

- The starter zip file includes a starter JS file
- Include the line of code into the file: `console.log("Window loaded!");` such that the message is logged to the console *when the page is loaded*
- Link your `HTML` page to your JavaScript file using a `script` tag
- Refresh your page in the browser. Do you see the `console` message?

Part II: Testing a Button

Now let's set up a very basic JS event handler. Modify your JS code and so that the "Button clicked!" `console` message won't output until the user clicks the "Encrypt-It!" button.

Strategy:

- First, make sure you understand why the event listener for `window`'s `load` event is needed?
- Move your `console` statement inside a new function `handleClick`.
- Add an event listener to the the "Encrypt-It!" button such that when clicked, your `handleClick` function is called.
- Refresh your page in the browser. Click the button. Do you see the `console.log` message? If so, move on. Otherwise, double-check the syntax and for both of your event listeners (window load and button click), or ask a TA for help

Part III: Implementing a Basic Shift-Cipher

Modify your JS code so that when the user clicks "Encrypt-It!", the text in the input text area will be encrypted using a basic shift-cipher, and output into the page's paragraph element with the id of `output`.

Details:

- To get text from the textarea, you'll need to make sure you can access it from JS. You can use `document.getElementById` or `document.querySelector` to access a DOM element in JS.
- Modify (and appropriately rename) your `handleClick` function so that when called, it now retrieves the textarea's text value and generates a shift cipher (algorithm discussed below; a solution for this cipher function is provided at the very bottom if you'd like to skip the algorithm part). This generated cipher will be output as text in the `#result` paragraph.

The rules of a shift cipher are fairly straightforward. Let the English alphabet we all know and love(?) be called *A*. Let the shift-encrypted alphabet be called *C*. For simplicity, we will shift letters in our encryption function by 1 letter. Then *C* is defined as mapping each letters in *A* to the letter alphabetically next. For example, 'a' is mapped to 'b', 'b' is mapped to 'c', ... and 'z' is mapped to 'a' (creating a cycle of 26 letters). In this exercise, we will consider uppercase letters and lowercase letters equivalent to one another (ie, 'a' is considered equal to 'A').

Visually, the cipher can be represented as the following:

```
input letter     a b c d e f g h i j k l m n o p q r s t u v w x y z
                 | | | | | | | | | | | | | | | | | | | | | | | | | |
                 v v v v v v v v v v v v v v v v v v v v v v v v v v
output letter    b c d e f g h i j k l m n o p q r s t u v w x y z a
```

Your task in this part is to convert the text in the input text area from alphabet *A* to alphabet *C*. This is all you need to know to implement the cipher in this lab, but if you would like additional hints, here are some provided:

Part III: Hints

Note that the value you get from the textarea is just a long string. So your goal is to build up a new string that is the result of applying the cipher to each letter in the input text, in order, and adding it to your result string.

There are a few ways to go about this, but note that one of the most intuitive approaches would be to use a `for` loop through the input string and add 1 to each letter (letters are actually represented by

numerical values, so this is a natural operation). To handle the z -> a shift, you can add a special case for each letter, or use mod arithmetic to avoid this extra case.

You may find `charCodeAt(index)` (https://www.w3schools.com/jsref/jsref_charcodeat.asp) and `fromCharCode(asciiNum)` (https://www.w3schools.com/jsref/jsref_fromcharcode.asp) helpful for this problem.

Here is a sample solution:

```
/**
 * Returns an encrypted version of the given text, where
 * each letter is shifted alphabetically ahead by 1 letter,
 * and 'z' is shifted to 'a' (creating an alphabetical cycle).
 */
function shiftCipher(text) {
  text = text.toLowerCase();
  let result = "";
  for (let i = 0; i < text.length; i++) {
    if (text[i] < 'a' || text[i] > 'z') {
      result += text[i];
    } else if (text[i] == 'z') {
      result += 'a';
    } else { // letter is between 'a' and 'y'
      let letter = text.charCodeAt(i);
      let resultLetter = String.fromCharCode(letter + 1);
      result += resultLetter;
    }
  }
  return result;
}
```

Submit your zip file with any changes made to the starter zip.