CS-UY 1114: Lab 6

Strings

You must get checked out by your lab CA **prior to leaving early.** If you leave without being checked out, you will receive 0 credits for the lab.

Restrictions

The Python structures that you use in this lab should be restricted to those you have learned in lecture so far. Please check with your teaching assistants in case you are unsure whether something is or is not allowed!

Create a new python file for each of the following problems.

Your files should be named *lab[num]_q[num].py* **similar to homework naming conventions.**

Problem 1: *History with Strings*

Solve this problem by hand on paper

1. Given the assignment statement below, write expressions that evaluate to the specified strings. For example, given the assignment statement below, the Python expression name[0] evaluates to "G".

   ```
   name = "Grace Hopper"
   ```

   - Write an expression whose value is "r".
   - Write an expression using the function len whose value is "r".
   - Write an expression whose value is "Grace". Use the slicing operator.

2. Given the assignment statement below, write expressions that evaluate to the specified strings. Use the slicing operator.

   ```
   name = "alan turing"
   ```

   - "Alan"
   - "Turing"

3. Given the following use a combination of slicing and concatenation to write expressions that evaluate to the specified strings.

   ```
   S = "Computer Science"
   ```

   - "Cmue cec"
   - "Comp Sci"

4. Given the following use a combination of slicing and concatenation to write expressions that evaluate to the specified strings.

   ```
   place = "Bletchley Park, England"
   ```

- "England"
- "yelhctelB"

Problem 2: *Reverse it*

Note: You may not reverse the string for this problem through the `.reverse()` method or slicing.

Write a program that will take an input from the user and reverse it. This is a quick and straightforward problem that focuses on indexing with strings.

Here are some sample executions of the program:

```
Enter your phrase: hello
olleh
```

```
Enter your phrase: CS 1114
4111 SC
```

Problem 3: *Hamming Distance*

In computer science, it is useful to be able to quantify the difference between two strings. One approach is called the Hamming distance. To find the Hamming distance, count the number of positions at which the corresponding characters differ between two strings. Hamming distance requires that the strings be of equal length.

For example: The Hamming distance between `"cat" and "cat" is 0` because the strings contain the same characters at all index positions.

The Hamming distance between `"cat" and "car" is 1` because there is one index position at which the strings differ (index position 2). At index positions 0 and 1, the strings contain the same characters.

Here are a couple more examples:

```
"cat" and "pot" Hamming distance is 2
"cat" and "dog" Hamming distance is 3
```

Write a program which the Hamming distance between str1 and str2 both entered by the user. You may assume that the user enters strings of equal length.

Hint: You need to find a way to compare the strings character by character.

Problem 4: *It's a secret...*

Write program that takes as input a line of text and outputs a new line of text with the digits of the integer numbers replaced by the character x.

Below is a sample output of the program:

```
Enter a phrase: My userID is john17 and my 4 digit pin in 1234 which is
secret
My userID is john17 and my x digit pin in xxxx which is secret
```

Notice that the digits in the username `john17` were **not** changed to x. If the digit is part of a word, it should **NOT** be changed. You may assume that the text entered by the user will contain only letters, digits and may also assume that there is exactly one space between each word in the line of text.

Hint: One approach to solving this problem is to use the slicing operator to isolate each word in the line. In order to do this, you will need to know the index position at which the word starts and ends. You can use the space character `" "` as an indicator of separation between words. You will need to iterate over the string and check if you encounter the space character `" "`. You will also need to store in the indices of the a variable the previous space that you encountered so that when you encounter a new space, you will know the space that comes before the word and the space that comes after the word. Note that you will need to account for the first and last word in a string differently.

Once the words are isolated the problem is more straightforward consider using the `isdigit()` method to your advantage.