CSC120H Lab 4

1 Objectives

- Practise using strings and string methods
- Practice writing for loops over strings

2 Driver and navigator

As always, you **must** complete this lab **with a partner**, and you and your partner will take on distinct roles:

driver: The person typing at the keyboard.

navigator: The person watching for mistakes, and thinking ahead.

The rest of these instructions call you `s1` and `s2`. Pick which one is which. `s1` should log in, start up Wing, and be the first driver.

3 Strings Warmup

Write the answers to the questions below on this handout, and then try them in the shell. If you get an answer that you didn't expect, don't move on until you figure out your misunderstanding. If you get stuck, ask your TA (or another nearby group) for help.

Consider this code:

```
s = 'a'
t = 'b'
```

1. What does the expression `s + t` evaluate to?
   (a) `'st'` (b) `'s + t'` (c) `'a + b'` (d) `'ab'` (e) None of the above
2. What does the expression `s * t` evaluate to?
   (a) `'st'` (b) `'s * t'` (c) `'a * b'` (d) `'ab'` (e) None of the above
3. What does the expression `s * 6` evaluate to?
   (a) `'s6'` (b) `'ssssss'` (c) `'6s'` (d) `'aaaaaa'` (e) None of the above
4. Which value does `s[0]` refer to? _____
5. Consider this code:

   ```
   s = 'm'
   s = 'c'
   ```

   After the code above is executed, which value does `s` refer to? _____
6. Consider this code:

   ```
   s = 'csc120'
   t = s[2:5]
   ```

   After the code above is executed, which value does `t` refer to? _____

7. Consider this code:

   ```
   s = ''
   ```

   What does `len(s)` return? _____

8. Consider this code:

   ```
   s = 'heffalump'
   ```

   What does `len(s)` return? _____

   What does `s[4]` refer to? _____

   What does `s[len(s) - 1]` refer to? _____

9. What does the code below print?

   ```
   vowels = 'aeiou'
   t = 'abcdefghIJKLMNO'
   for c in t:
       if c in vowels:
           print(c)
   ```

4 Writing `for` Loops over strings

This part of the lab asks you to write the code for a set of functions that use strings, and in some cases, `for` loops that iterate over strings.

1. Download and save the file `lab4_part4.py` on the Labs page of the course website.
2. You will fill out the function bodies as described in the docstrings. Some examples calls may be given, but you should fill in your own examples where asked to in the docstrings.
3. After writing your function bodies, test your docstring examples in the shell.
4. After (and outside) each function, `print` out the function calls of at least three of **your own** test cases (not the ones in the docstring examples).
5. Make sure your test cases give back the correct value.
6. Repeat steps 2-5 for every function.

Switch driver and navigator roles for <u>each</u> function.

5 Using `str` Methods

Methods act much like functions. A method is a function that belongs to a specific data type and is designed to make use of the specific value on which it is called. For example, `'abc'.upper()` uses the value `'abc'` to return its uppercase equivalent, `'ABC'`. The methods available for a specific value depend on the value's type: `'abc'` has a method `upper()` because `'abc'` is a `str`.
The last page of the lab contains a list of `str` methods and their `help` descriptions. Use the appropriate `str` methods from that list to complete the tasks below.
Create strings in the Python shell and do the following tasks (alternate driving and navigating). Each can be done with a single method call. Write your answers on the next page. The first one is done for you as an example.

1. Check whether or not a string ends with the letter `"h"`.

```
s1 = 'abc'
s2 = 'fgh'
s1.endswith('h')
s2.endswith('h')
```

2. Check whether or not a string starts with the letter `"w"`.

3. Check whether or not a string contains only numbers.

4. Replace every instance of the character `"l"` (*ell*) in a string with a `1` (*the number one*).

5. Check whether or not a string contains only lowercase letters.

6. Remove all leading zeroes (any zeroes at the beginning of the string) from a string composed only of numbers.

7. Remove all trailing ones (any `1`s at the end of the string) from a string composed only of numbers.

Short Python help descriptions for `str` methods:

```
str:
  x in s --> bool
    Produce True if and only if string x is in string s.
  str(x: object) -> str
    Convert an object into its string representation, if possible.
  S.count(sub: str[, start: int[, end: int]]) -> int
    Return the number of non-overlapping occurrences of substring sub in
    string S[start:end].  Optional arguments start and end are interpreted
    as in slice notation.
  S.find(sub: str[, i: int]) -> int
    Return the lowest index in S (starting at S[i], if i is given) where the
    string sub is found or -1 if sub does not occur in S.
  S.index(sub: str) -> int
    Like find but raises an exception if sub does not occur in S.
  S.isalnum() -> bool
    Return True if and only if all characters in S are alphanumeric
    and there is at least one character in S.
  S.isalpha() -> bool
    Return True if and only if all characters in S are alphabetic
    and there is at least one character in S.
  S.isdigit() -> bool
    Return True if and only if all characters in S are digits
    and there is at least one character in S.
  S.islower() -> bool
    Return True if and only if all cased characters in S are lowercase
    and there is at least one cased character in S.
  S.isupper() -> bool
    Return True if and only if all cased characters in S are uppercase
    and there is at least one cased character in S.
  S.lower() -> str
    Return a copy of the string S converted to lowercase.
  S.lstrip([chars: str]) -> str
    Return a copy of the string S with leading whitespace removed.
    If chars is given and not None, remove characters in chars instead.
  S.replace(old: str, new: str) -> str
    Return a copy of string S with all occurrences of the string old replaced
    with the string new.
  S.rstrip([chars: str]) -> str
    Return a copy of the string S with trailing whitespace removed.
    If chars is given and not None, remove characters in chars instead.
  S.split([sep: str]) -> list of str
    Return a list of the words in S, using string sep as the separator and
    any whitespace string if sep is not specified.
  S.strip([chars: str]) -> str
    Return a copy of S with leading and trailing whitespace removed.
    If chars is given and not None, remove characters in chars instead.
  S.swapcase() -> str
    Return a copy of S with uppercase characters converted to lowercase
    and vice versa.
  S.upper() -> str
    Return a copy of the string S converted to uppercase.
```