CSE 231 **Spring 2014**

Computer Project #5

Assignment Overview

This assignment focuses on the design, implementation and testing of a library of Python functions to manipulate character strings, as described below.

It is worth 40 points (4% of course grade) and must be completed no later than 11:59 PM on Monday, February 17.

Assignment Deliverables

The deliverables for this assignment are the following files:

`proj05library.py` – the source code for your library module
`proj05test.py` – the source code for your program to test your library module

Be sure to use the specified file names and to submit them for grading via the **handin system** before the project deadline.

Assignment Specifications

1. The library module will contain three constants and eight function definitions.

a) It will include the following constants:

```
ASCII_LOWERCASE = "abcdefghijklmnopqrstuvwxyz"
ASCII_UPPERCASE = "ABCDEFGHIJKLMNOPQRSTUVWXYZ"
DECIMAL_DIGITS  = "0123456789"
```

b) It will include the following functions:

```
is_alpha( str ) → bool
is_digit( str) → bool
to_lower( str ) → str
to_upper( str ) → str
find_chr( str, str ) → int
find_str( str, str ) → int
replace_chr( str, str, str ) → str
replace_str( str, str, str ) → str
```

The notation above gives the name of each function, the number and type of its argument(s), and the type of its return value.

c) The function names will be spelled exactly as shown (for example, `is_digit`).

d) Function `is_alpha` has one parameter (a string). It returns True if all of the characters in the string are upper case or lower case ASCII letters (it returns False otherwise).

e) Function `is_digit` has one parameter (a string). It returns True if all of the characters in the string are ASCII decimal digits (it returns False otherwise).

f) Function `to_lower` has one parameter (a string). It returns the string which is a copy of the parameter, but where all of the upper case ASCII letters have been converted to lower case ASCII letters.

g) Function `to_upper` has one parameter (a string). It returns the string which is a copy of the parameter, but where all of the lower case ASCII letters have been converted to upper case ASCII letters.

h) Function `find_chr` has two parameters (both strings), where the second parameter must be of length 1. It returns the lowest index where the second parameter is found within the first parameter (it returns -1 if the second parameter is not of length 1 or is not found within the first parameter).

i) Function `find_str` has two parameters (both strings). It returns the lowest index where the second parameter is found within the first parameter (it returns -1 if the second parameter is not found within the first parameter).

j) Function `replace_chr` has three parameters (all strings), where the second and third parameters must be of length 1. It returns the string which is a copy of the first parameter, but where all occurrences of the second parameter have been replaced by the third parameter (it returns the empty string if the second or third parameter are not of length 1).

k) Function `replace_str` has three parameters (all strings). It returns the string which is a copy of the first parameter, but where all occurrences of the second parameter have been replaced by the third parameter. If there are no occurrences of the second parameter in the first, it returns a copy of the first parameter. If the second parameter is the empty string, it returns the string which is a copy of the first parameter, but with the third parameter inserted before the first character, between each character, and after the last character.

l) The library module will not use any of the string methods listed in Section 4.7.1 of the Python Standard Library:

http://docs.python.org/3.3/library/stdtypes.html#string-methods

m) The library module will not contain any `import` statements.

n) The library module will not perform any input or output operations. For example, it will not call function `input` or function `print`.

2. The program to test your library module will demonstrate that each of the three constants and eight functions is implemented correctly.

a) It will contain one of the following `import` statements:

```
import proj05library
from proj05library import *
```

b) It may contain additional `import` statements.

c) It may use any of the string methods listed in Section 4.7.1 of the Python Standard Library. For example, it may use the string method `lower` as a means of determining whether or not the value returned from function `to_lower` is correct.

d) The program will not perform any input operations (it will not call function `input`).

e) For each test case, the program will display the appropriate information so that someone reading the output can understand the purpose of the test and judge if the result is correct.

Assignment Notes

1. Approximately 25% of the total points will be allocated to the test program, so be sure that it includes sufficient test cases to demonstrate to your TA that you have thoroughly tested your implementation of each of the constants and functions which are part of the library module.

2. The test program may not perform any input operations, so all of your test cases must be embedded in the source code. That is, your program may not prompt the user to enter test cases.

3. The library module may not perform any input or output operations, so all "communication" between the test program and the functions in the library module must be done by passing parameters and receiving return values.

4. The following files in the project directory are a simple example of a library module which contains one constant and one function definition, and a program to test the library module:

`project05library_ex.py` – the source code for the library module
`project05test_ex.py` – the source code for the test program

You may wish to examine those files and experiment with them.

5. The following functions from the built-in library may be useful when you implement your library module:

```
ord( str ) → int      # return code of ASCII character
chr( int ) → str      # return ASCII character of code
```

Suggested Procedure

- *Solve the problem using pencil and paper first.* You cannot write a program until you have figured out how to solve the problem. This first step is best done collaboratively with another student. However, once the discussion turns to Python specifics and the subsequent writing of Python statements, you must work on your own.

- Write a simple version of the library module and of the program. The library module should contain one function definition (perhaps function `is_digit`) and the program should contain your test cases to thoroughly test that function.

- Run the program and track down any errors.

- Use the **handin system** to turn in the first version of your solution.

- Cycle through the steps to incrementally develop your program:
    - Edit your library module and program to add new capabilities.
    - Run the program and fix any errors.
    - Use the **handin system** to submit the current version of your solution.

- Use the **handin system** to submit the final version of your solution.

- You would be wise to back up your files on your H: drive, also.

- Be sure to log out when you leave the room, if you're working in a public lab.