CSE 101 - Introduction to Programming
Tutorial 3

Topics: Modules and Functions, Importing and using functions from modules, Defining and calling functions

0. Define a function that takes in coefficients of quadratic equation as input and prints the roots of the equation.

1. Define a module named **`string_module`**. Include docstrings for module and functions describing what they do. Import this module in a script and call these functions. The module should contain following functions:

- **`rotate`**: takes a string, rotation direction and count as parameters and returns a string rotated in the defined direction count times
  - `rotate('Hello', 'left', 2)` should return `'lloHe'`
  - `rotate('Hello', 'right', 3)` should return `'lloHe'`

- **`fitin`**: takes two parameters: a tag string of length 4, such as "{{}}", and a word. The function should return a new string where the word is in the middle of the tag string, e.g. "{{word}}".
  - `fitin('(())', 'Hello')` returns `'((Hello))'`
  - `fitin('<<>>', 'Yikes')` returns `'<<Yikes>>'`
  - `fitin('[[]]', 'Black')` returns `'[[Black]]'`

2. Write a Python program to swap the values of two variables with and without using a temporary variable. Use a function call for doing so. Also notice how it affects the actual arguments passed.

Answer 0:

```
def solve_quadratic(a, b, c):
      D = b**2 - 4*a*c
      if D < 0:
            x1 = (-b + (-D)**0.5)/(2*a)
            x2 = (-b - (-D)**0.5)/(2*a)
            print("x1 = " + str(x1) + "i\nx2 = " + str(x2) + "i\n")
      else:
            x1 = (-b + D**0.5)/(2*a)
            x2 = (-b - D**0.5)/(2*a)
            print("x1 =", x1, "\nx2 =", x2)
```

Answer 1:

string_module.py

```
"""
This module provides string formatting functions.
"""

def rotate(s, direction, k):

      """
      This function takes a string, rotation direction and count
      as parameters and returns a string rotated in the defined
      direction count times.
      """

      k = k%len(s)

      if direction == 'right':
            r = s[-k:] + s[:len(s)-k]
      elif direction == 'left':
            r = s[k:] + s[:k]
      else:
            r = ""
            print("Invalid direction")

      return r
```

```
def fitin(tag, s):

      """
      This function returns a new string where the word is
      in the middle of the tag string.
      """

      return tag[:2] + s + tag[2:]
```

script_that_uses_string_module.py

```
import string_module

s1 = string_module.rotate('HelloWorld', 'left', 11)
s2 = string_module.rotate('Thisistutorialthree', 'right', 3)

print(s1, s2)

s3 = string_module.fitin('<<>>', 'Yikes')
s4 = string_module.fitin('[[]]', 'Black')

print(s3, s4)
```

Answer 2:

```
def swap(x, y):
      x = x + y
      y = x - y
      x = x - y
      print(x, y)
      print("x is", x, "and y is", y)

x = 10
y = 20

print("x is", x, "and y is", y)
swap(x, y)
print("x is", x, "and y is", y)
```

Were the values of x and y really swapped, what really happened?

Refer:
https://robertheaton.com/2014/02/09/pythons-pass-by-object-reference-as-explained-by-philip-k-dick/
https://www.python-course.eu/passing_arguments.php