Problem Solving Methodology in IT (COMP1001)

Assignment Five
(Due at noon on 25 October 2018)

(Instructions: see the instructions in Blackboard)

Rocky K. C. Chang

16 October 2018

1. [Weight = 3] (String reversing) Write a Python program that will accept a string consisting of multiple words and output the reverse of the words in the string with only one space between adjacent words. You can keep the punctuations in the string except for the full stop at the end of the string. The full stop, if exist, will stay at the end of the output string.

```
Function reverseString(inStr)
    Input: inStr, a string generally consisting of multiple words
    separated by at least one space.
    Output: return a string with the words arranged in an opposite
    order, and the adjacent words are separated by only one space. The
    punctuations, if any, can be kept with the words, except for the
    last full stop, if any, which will stay at the end of the output
    string.
```

   Include the following statements in your .py file.

   - `print("The correct output is:", reverseString("Here I am. Send me."))`
   - `print("The correct output is:", reverseString("Here I am. Send me!"))`
   - `print("The correct output is:", reverseString("All hard work brings a profit, but mere talk leads only to poverty."))`
   - `print("The correct output is:", reverseString("The beginning of wisdom is this: Get wisdom, and whatever you get, get insight."))`

   The expected print out should be

```
The correct output is: me Send am. I Here.
The correct output is: me! Send am. I Here
The correct output is: poverty to only leads talk mere but profit, a brings work hard All.
The correct output is: insight get get, you whatever and wisdom, Get this: is wisdom of beginning The.
```

2. [Weight = 6] (Tic Tac Toe) Visit https://inventwithpython.com/chapter10.html for a Python implementation of the famous tic tac toe game. Please download it and run it a few times to familiarize yourself with the features of this program. Read the code to understand how it works. You should be able to understand this program, because we have covered everything used in the code. In this question, you are asked to make a few enhancements to this code.
   a. [Weight = 1] The nine places on the board are not labeled. To aid the human player to make his move, the first enhancement is to show the numbers of the 9 places on the board. The expected output is

```
   |   |
 7 | 8 | 9
   |   |
-----------
   |   |
 4 | 5 | 6
   |   |
-----------
   |   |
 1 | 2 | 3
   |   |
```

   b. [Weight = 1] Currently there is no way for the human player to stop the current game once it is started. The second enhancement is to add an option for the human player to stop the game when he enters -1. The expected output is

```
What is your next move? (1-9). Enter -1 if you want to terminate this game. -1
Do you want to play again? (yes or no)
```

   c. [Weight = 1] When the human player enters a wrong number, say 10, or the space is already taken, the program will prompt the user for another input. However, the message prompt is the same for both cases. The third enhancement is therefore to give different messages for these two cases. The expected output is

```
What is your next move? (1-9). Enter -1 if you want to terminate this game. 10
Please enter a valid number. Enter -1 if you want to terminate this game. 12
Please enter a valid number. Enter -1 if you want to terminate this game. 1
Please enter a valid number. Enter -1 if you want to terminate this game. 9
```

   d. [Weight = 3] After the human player enters his number, the board will be updated with the human player's move as well as the computer player's next move. Often it is hard to see both changes at the same time. In the fourth enhancement, the program will wait for the human player to enter anything for the program to display the computer player's move. For example,

```
Enter anything to see computer's move.
   |   |
 O | 8 | X
   |   |
-----------
   |   |
 4 | 5 | 6
   |   |
-----------
   |   |
 O | X | O
   |   |
```