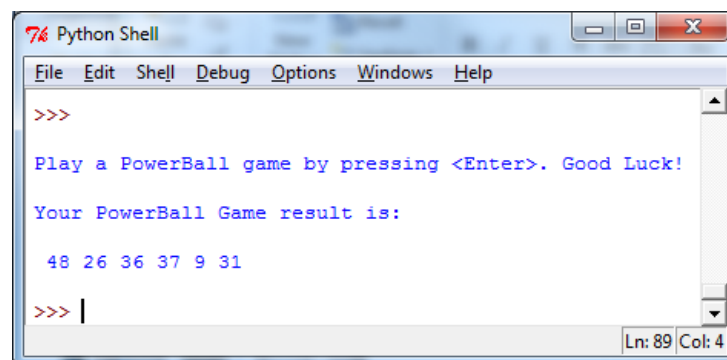


## STSCI 4060 HW2

(Assigned on: 2/21/2019, Due: 11:59 PM, 3/2/2019)

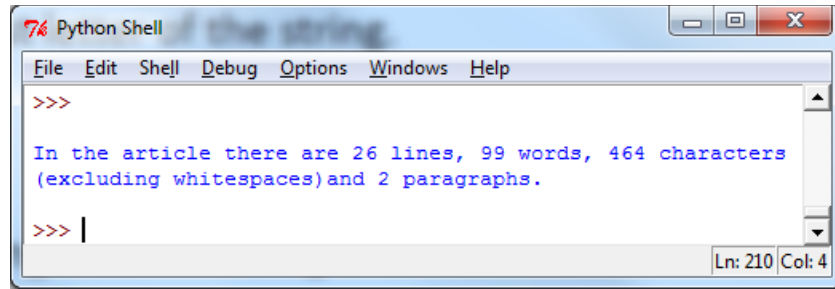
### General Instruction:

- **Do your own work. Cornell academic integrity rules are enforced. Copying code from the Internet, fellow students or elsewhere is prohibited.**
  - **You must include some comments to show the question number and what your program does for each Python program. If this part is missing, up to 5 points will be deducted.**
  - **What to submit:**
    1. Python files (with a .py extension), one single file for each question.
    2. An MS Word (or PDF) file, containing screenshots of the outputs from all the questions.
  - **How to submit:** Upload a single zipped file containing all the above components to the course website on Blackboard.
1. (10 points) Develop your own Powerball lottery game. In the game you randomly select six numbers. The first five numbers are selected from a group of white balls numbered from 1 through 59 (without replacement) and the sixth number is selected from another group of balls (called power balls, colored red) numbered from 1 through 35. Write a Python program to accomplish this. You should output results like the following. Note that the spacing and layout of the output and that there should be no square brackets enclosing the numbers.



```
Python Shell
File Edit Shell Debug Options Windows Help
>>>
Play a PowerBall game by pressing <Enter>. Good Luck!
Your PowerBall Game result is:
48 26 36 37 9 31
>>> |
Ln: 89 Col: 4
```

2. (20 points) You are given an article (AboutCornell.txt), which can be downloaded from the course website. Write a Python program to find out the number of lines of the text (blank lines are counted), the number of words, the number of characters (excluding whitespaces) and the number of paragraphs in the article. An output example (not a real result of this question) is shown below.



```
Python Shell
File Edit Shell Debug Options Windows Help
>>>
In the article there are 26 lines, 99 words, 464 characters
(excluding whitespaces) and 2 paragraphs.
>>> |
```

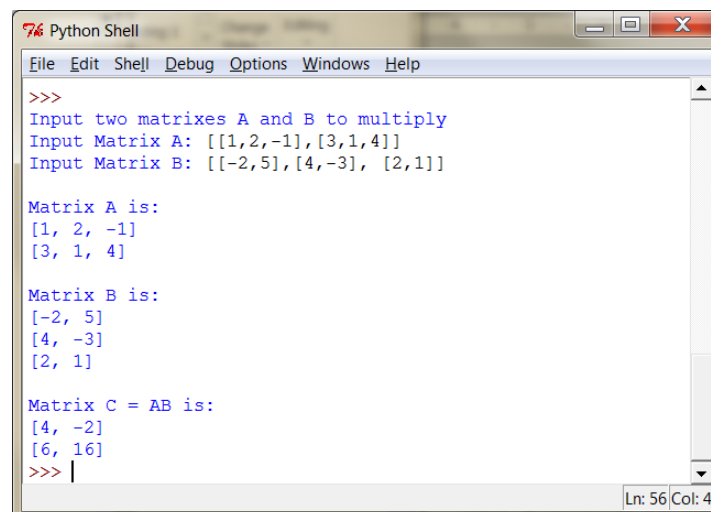
3. (40 points) Given two matrixes A and B of arbitrary dimensions, you are required to write a Python program by defining a function called `matrixMultiplication()` to do matrix multiplications, which should take matrixes A and B as its two arguments and defining a main function that uses the `matrixMultiplication()` function. Your program must be suitable for all situations, i.e., it is a general-purpose program. First, you judge if you can do a multiplication on these two matrixes. If not, output a message saying "Error, these two matrixes cannot be multiplied since their dimensions do not match." And then prompt the user to enter two different matrixes again. If the dimensions of the two matrixes meet the requirement, you implement the process based on the formula of matrix multiplication (i.e., you are not allowed to import any pre-defined modules to do the multiplication). You input matrixes A and B in your main function, call the `matrixMultiplication()`, and display matrixes A and B and the result matrix called C. Your program must first produce the same output as the screenshot below, and then you show:

- 1) Your multiplication result of another pair of matrixes of your choice.
- 2) The output where the two given matrixes cannot be multiplied.

Use the `range()` function and nested for loops, etc. You need to initialize matrix C with all the entries set to zero. You can use the following method to define and initialize a list L:

$L = [\text{expression for variable in sequence}]$

where, *expression* is the value zero in this case; *variable* can be any Python variable name; and *sequence* is a list with a length of the number of columns of matrix C. Think about how you initialize the whole matrix with this method.



```
Python Shell
File Edit Shell Debug Options Windows Help
>>>
Input two matrixes A and B to multiply
Input Matrix A: [[1,2,-1],[3,1,4]]
Input Matrix B: [[-2,5],[4,-3],[2,1]]

Matrix A is:
[1, 2, -1]
[3, 1, 4]

Matrix B is:
[-2, 5]
[4, -3]
[2, 1]

Matrix C = AB is:
[4, -2]
[6, 16]
>>> |
```

4. (30 points) From the data file **stock.txt**, you are required to create a dictionary, **myDict**, containing four keys: **stockNames**, **dates**, **openPrices** and **closePrices**, and then from **myDict** find out the stock names that increased their prices on that day. The values of these dictionary keys are lists named **names**, **dates**, **openPrices** and **closePrices**, respectively. Each line of the text of **stock.txt** represents the values for stock name, date, stock open price, stock high price, stock low price, stock close price, stock volume and stock price adjacent to close, respectively. Extract text from the first line, line by line, to populate these four lists so that the list **names** contains the stock names, the list **dates** contains the dates of trading, the list **openPrices** contains stock open prices, and the list **closePrices** contains the stock close prices. Display the four lists and leave a blank line between your outputs, in the form similar to the following (same for the rest unless otherwise specified):

The list names is:

```
['ABC', 'DEF', 'GHI']
```

Use Python **built-in functions** to create **myDict** with a single statement. Display **myDict**. After that, create a list, **increasedStocks**, to hold the stock names that increased their prices on that day (hint: use a **for loop** to populate the list). Lastly, display the content of **increasedStocks**, with the following format:

The stocks that increased their prices are ABC, DEF, ..., XYZ.