Assignment 02

Your program should

- read a csv file (provided by the user as a required argument) into 2-dimensional array. The csv file contains only numeric data. The first column is the y-value (dependent variable) and the last two columns are x-values (independent variables),
- compute the regression coefficients using arrays or matrices only (as we did in class) and not any specialized functions,
- the program output needs to be either brief (-b), verbose (-v) or the default
  - the **verbose** output contains
    - the description of the equation
    - the R^2 value
    - actual data values with column titles
  - the **brief** output contains the coefficients of the regression equation
  - the **default** output contains the coefficients of the regression equation and R^2 value
- the program should have the option to plot (-p) the regression – actual values as an XY-plot and the estimated values as a hyperplane
- the program should have the option (-o) to send the output (**verbose** only) to a file with the name of the file provided by the user. The argpase syntax to do that is provided below:

```
parser.add_argument('-o', dest='outfile', action='store',
                    help='output file')
```

You are required to use <u>at least three functions</u> for input, processing and output. They are

```
fileInput(fileName)
       """
       Input = name of csv text file with comma separated numbers
       Output = numpy array
       """

def regress(myData):
       """
       Input = numpy array with three columns
       Column 1 is the dependent variable
       Columns 2 and 3 are the independent variables
       Returns = a column vector with the b coefficients
       """

def myPlot(myData, b):
       """
       Input = numpy array with three columns
       Column 1 is the dependent variable
       Columns 2 and 3 are the independent variables
       and
       a column vector with the b coefficients
       Returns = Nothing
       Output = 3D plot of the actual data and
       the surface plot of the linear model
       """
```

I have provided the **`myPlot`** function. You are free to create more functions if you feel the need for them. All these functions should be placed in a module named **`A02Module_Gwid.py`**.

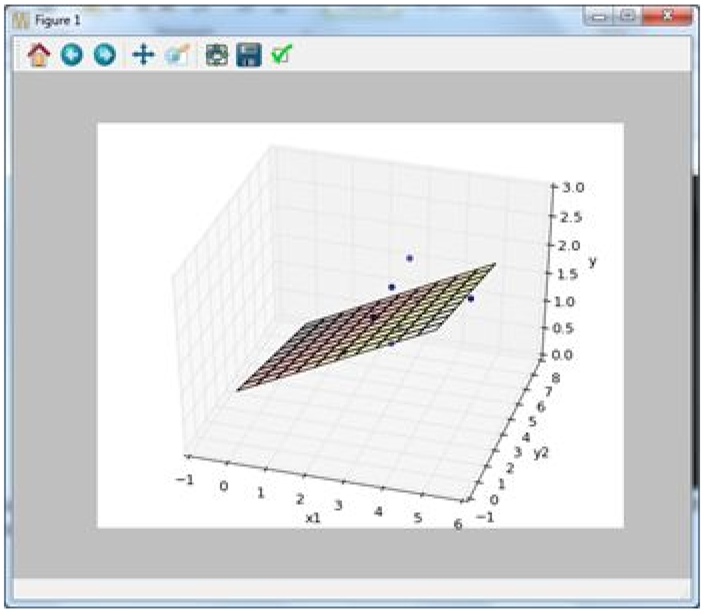
Sample output

```
$ python a02_gwid.py mycsv.txt
b0 = 0.92
b1 = 0.32
b2 = -0.10
R-Square = 0.31


$ python a02_gwid.py mycsv.txt -b
b0 = 0.92
b1 = 0.32
b2 = -0.10


$ python a02_gwid.py mycsv.txt -v
The equation is:
y = 0.92 + 0.32x1 + -0.10x2
The R-square value is:  0.31
The formatted input data is shown below:
       y      x1       x2
=========================
   1.00      2.00     3.00
   2.00      3.00     4.00
   1.00      3.00     4.00
   2.00      3.00     2.00
   2.00      3.00     6.00
   1.00      3.00     5.00
   2.00      5.00     4.00
   1.00      3.00     7.00


$ python a02_gwid.py mycsv.txt -p -o myout.txt
b0 = 0.92
b1 = 0.32
b2 = -0.10
R-Square = 0.31
Output being sent to myout.txt
```



How to submit your assignment

1. You need to submit two Python files to Blackboard to the Assignment 02 link.
2. The program should be commented well enough so that the TA or I should not have to struggle with understanding variable names and codes and what statements or code blocks do.
3. The grading rubric is shown on the last page.
4. Name the files as `A02_Gwid.py and A02Module_Gwid.py` . So if your GWID is G19860011 then you should name your files as `A02_G19860011.py` and `A02Module_G19860011.py` .
5. Your program headers for each program should look something like

```
# -*- coding: utf-8 -*-
"""
Created on Fri Sep 04 09:09:54 2016

@author: kanungo
GWID: G19860011

A brief description of the program not exceeding two lines
"""
```

Rubric for Grading the Programming Assignment

	Unsatisfactory	**Satisfactory**	**Good**	**Excellent**
Delivery	· Completed less than 70% of the requirements. · Not delivered on time or not in correct format (Blackboard or git)	· Completed between 70-80% of the requirements. · Delivered on time, and in correct format (Blackboard or git)	· Completed between 80-90% of the requirements. · Delivered on time, and in correct format (Blackboard or git)	· Completed between 90-100% of the requirements. · Delivered on time, and in correct format (Blackboard or git)
Coding Standards	· No name, date, or assignment title included · Poor use of white space (indentation, blank lines). · Disorganized and messy · Poor use of variables (many global variables, ambiguous naming).	· Includes name, date, and assignment title. · White space makes program fairly easy to read. · Organized work. · Good use of variables (few global variables, unambiguous naming).	· Includes name, date, and assignment title. · Good use of white space. · Organized work. · Good use of variables (no global variables, unambiguous naming)	· Includes name, date, and assignment title. · Excellent use of white space. · Creatively organized work. · Excellent use of variables (no global variables, unambiguous naming).
Documentation	· No documentation included.	· Basic documentation has been completed including descriptions of all variables. · Purpose is noted for each function.	· Clearly documented including descriptions of all variables. · Specific purpose is noted for each function and control structure.	· Clearly and effectively documented including descriptions of all variables. · Specific purpose is noted for each function, control structure, input requirements, and output results.
Runtime	· Does not execute due to errors. · User prompts are misleading or non-existent. · No testing has been completed.	· Executes without errors. · User prompts contain little information, poor design. · Some testing has been completed.	· Executes without errors. · User prompts are understandable, minimum use of symbols or spacing in output. · Thorough testing has been completed	· Executes without errors excellent user prompts, good use of symbols, spacing in output. · Thorough and organized testing has been completed and output from test cases is included.
Efficiency	· A difficult and inefficient solution.	· A logical solution that is easy to follow but it is not the most efficient.	· Solution is efficient and easy to follow (i.e. no confusing tricks).	· Solution is efficient, easy to understand, and maintain.