Attach the assignment to your HW/Lab ENGR 102 Section **Lab 12a**
Date: _____ **DUE DATE: see website**
You name _____Team # (table)_____

ENGR 102 Sect Lab 12a team

17pts = 100 points

Reading assignment:

Lecture Slides	**L12**
zyBook chapter	**Chapter 12**
Handout	**Matplotlib for Python Developers**

Attention!!

You may work with others, but it is Individual submission.

Deliverables:
There are 2 deliverables for this team assignment. Please submit the following files to zyBooks:

- `numpy_example.py 12.4 6pts Autograded = 50%`
- `matplotlib_example.py 12.15 11pts = [2 auto]+9manual=50%`

This week's lab and individual assignments are meant to familiarize you with two of the most commonly used engineering packages in Python, `numpy` and `matplotlib`. It will also begin to give you more experience in making function calls and using a module. To do this, **as a team** you will work some of the basic tutorial/intro material for each one, and ensure that each member of the team knows how it works.

`Numpy` is a package for performing scientific computing. In particular, it offers functions and types that will allow you to create vectors and matrices, and perform linear algebra operations more easily. The website for `numpy` is: http://www.numpy.org/

`Matplotlib` is a library for creating plots, graphs, and charts. It supports a wide range of plots and graphs, and is often used for visualizing data. The website for `matplotlib` is: https://matplotlib.org/

Activity #0: Check for package installation – to do individually.
Both of these should already be installed on your system when you downloaded the Anaconda distribution of Python (or standalone Spyder), and you should have checked this in Week 1. If you did not get the Anaconda distribution, you may need to download these on your own. You should first check to ensure they are installed on your system.

1. *Create a program with two lines of code:*
```
import numpy
import matplotlib
```
and try to run this program. If the program runs without an issue (exit code 0), then you have both packages installed. If you get an error, then at least one is not installed. You may need to ask for help in getting this to run.

Activity #1: `numpy` – team
The point of this activity is to familiarize your team with the basic matrix and vector structures and operations in `numpy`. It will also begin to give you more experience in making function calls and using a

module. You will do this by going through the `numpy` tutorial as a team, and making some small modifications along the way.

a) First go to the `numpy` home page (link is above), then click on "Learn" (at the top). The "NumPy Quickstart Tutorial" shows several commands in the context of an interactive python interpreter. Your team should have one team member share their screen and try entering and modifying the code described below for everyone to see. Each team member may want to read the documentation as you work through the examples together.
b) As a team, work through parts of the tutorial (listed in 3. below). For each part, your team should do the following:
    a. If there is text, read it, and check to make sure that everyone understands it. For any code in a section, do parts b-e below.
    b. Write the code in your own IDE and execute it. You will probably need to put in print statements to print out values of variables that are shown automatically in the interactive view.
    c. Verify that the output is what you expect – that the arrays are created correctly and the values are computed.
    d. Make some modification to the code (e.g. to the entries of a matrix) and rerun to make sure you see how the various functions behave.
    e. Ask everyone in the group if they follow what has happened in that section. Do not go on until each person states that they understand.
        i. If some team members do not understand, others should help explain the topics.
        ii. If all of you are stuck, or need help, ask a member of the teaching team.
    f. Then, go on to the next section.
c) The parts of the tutorial to work through (there is a "Table of Contents" on the right side of the page) are: https://numpy.org/devdocs/user/quickstart.html
    a. The Basics – An example
    b. The Basics – Array Creation
    c. The Basics – Printing Arrays
    d. The Basics – Basic Operations
    e. The Basics – Universal Functions
    f. Shape Manipulation – Changing the shape of an array
    g. (Optional) Check out NumPy: the absolute basics for beginners (in the left menu) for a more detailed explanation of concepts
d) At the end, each person on the team should understand how to create arrays of various sizes, how to perform basic linear algebra operations (calculate dot product or do matrix multiplication), and how to use mathematical functions on an array.
e) Finally, to demonstrate that your team understands how this process works, create a program named numpy_example.py that does the following:
    a. Start your program with the following statement in comments after the header:
```
# As a team, we have gone through all required sections of the
# tutorial, and each team member understands the material
```
    b. Create 3 matrices, A, B, and C, of size 3x4, 4x2, and 2x3, respectively. For each matrix, have the first element be zero (0), and increase each subsequent element by one (1). Print each of these matrices with a single blank line in between.

c. Compute the product D = ABC and print the resulting matrix and a blank line. (Note: this is matrix multiplication not simple elementwise multiplication.)
d. Print the transpose of D and a blank line.
e. Calculate and print E = √(D) / 2.

Functions and Methods overview (with links)

Array Creation

arange, **array**, **copy**, **empty**, **empty_like**, **eye**, **fromfile**, **fromfunction**, **identity**, **linspace**, **logspace**, **mgrid**, **ogrid**, **ones**, **ones_like**, **r_**, **zeros**, **zeros_like**

Conversions

ndarray.astype, **atleast_1d**, **atleast_2d**, **atleast_3d**, **mat**

Manipulations

array_split, **column_stack**, **concatenate**, **diagonal**, **dsplit**, **dstack**, **hsplit**, **hstack**, **ndarray.item**, **newaxis**, **ravel**, **repeat**, **reshape**, **resize**, **squeeze**, **swapaxes**, **take**, **transpose**, **vsplit**, **vstack**

Questions

all, **any**, **nonzero**, **where**

Ordering

argmax, **argmin**, **argsort**, **max**, **min**, **ptp**, **searchsorted**, **sort**

Operations

choose, **compress**, **cumprod**, **cumsum**, **inner**, **ndarray.fill**, **imag**, **prod**, **put**, **putmask**, **real**, **sum**

Basic Statistics

cov, **mean**, **std**, **var**

Basic Linear Algebra

cross, **dot**, **outer**, **linalg.svd**, **vdot**

Example output:

```
A = [[ 0  1  2  3]
 [ 4  5  6  7]
 [ 8  9 10 11]]

B = [[0 1]
 [2 3]
 [4 5]
 [6 7]]

C = [[0 1 2]
 [3 4 5]]
```

```
D = [[ 102  164  226]
 [ 294  468  642]
 [ 486  772 1058]]

D^T = [[ 102  294  486]
 [ 164  468  772]
 [ 226  642 1058]]

E =  [[ 5.04975247  6.40312424  7.51664819]
 [ 8.5732141  10.81665383 12.66885946]
 [11.02270384 13.89244399 16.26345597]]
```

Activity #2: `matplotlib` – team

The purpose of this activity is for your team to review some of the capabilities of `matplotlib`, get practice learning how to find the parameter settings for various graphs, and make a few plots. As a team, create three (3) different plots described below. The expectation is that you will all work together to ensure everyone knows how to handle different aspects of the process.

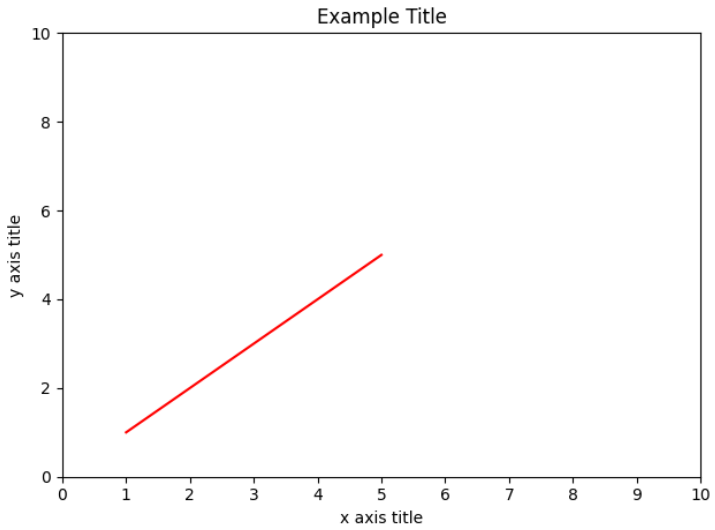
- First go to the `matplotlib` website (https://matplotlib.org/.). Similar to what was done with Activity #1, you should work through the code provided on this page for training.
    1. Click on "Tutorials" (at the top) and then select "Pyplot tutorial". Your team should go through the tutorial, as was done for `numpy` in Activity #1.
    2. You can also check out the "Examples" page to see a range of examples of plots that can be produced.
    3. Do not do more until everyone has followed the tutorial, and agreed that they understand it, and that the group has briefly looked at several examples.
- There are several sections that are a bit beyond what we need to learn. For this week, you should learn how to accomplish the following tasks:
    1. Create a figure.
    2. Plot numerical data (this includes plotting multiple sets of data on the same figure).
    3. At a minimum, label the figure with a title, x-axis label, and y-axis label.
    4. Control the x- and y-axis scale values.
    5. Create a legend.
    6. Control the width and color of a plotted line.
    7. Control the width, color, and shape of plotted points.
    8. Create subplots within a single figure.

Example

```
import matplotlib.pyplot as plt

# lets create some values for our x and y axes
x = [1, 2, 3, 4, 5]
y = [1, 2, 3, 4, 5]

# lets create our figure
fig, ax = plt.subplots()

# changing the grpah title
ax.set_title('Example Title')

# changing the x and y axis labels
ax.set_ylabel('y axis title')
```

```
ax.set_xlabel('x axis title')

# set limits for x and  y axes
ax.set_xlim(0, 10)
ax.set_ylim(0, 10)

# set the tick marks
plt.gca().xaxis.set_major_locator(plt.MultipleLocator(1))
plt.gca().yaxis.set_major_locator(plt.MultipleLocator(2))

# to change the color of the line to red say color='r' in the plot function
ax.plot(x, y, color='r', label='example label')

# add a legend to our plot
plt.legend()

plt.show()
```

This example code outputs this graph:



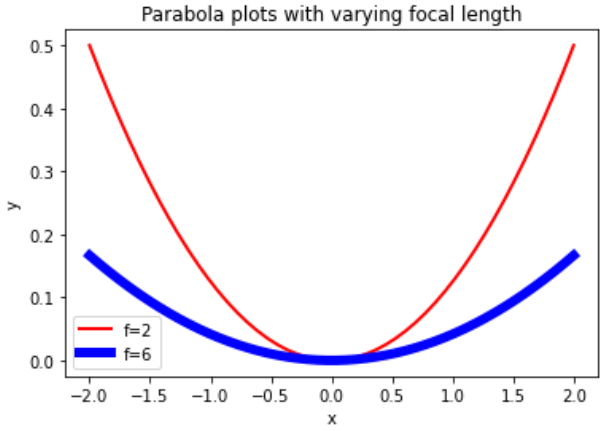
Assignment

- As a team, create a program named matplotlib_example.py that produces the three (3) plots described below. Include the same statement in comments as you did in Activity #1. Each plot should have the plotted data, a meaningful title to describe the data, labels for the x- and y-axes, and a legend when plotting two (2) or more things. Feel free to change the colors, markers, and line styles, unless otherwise noted. This activity will be manually graded. Useful hint: to establish an array of x-values, check out `numpy.linspace()`.

    1. One equation for a parabola is given below in terms of its focal length (f):

    $$y = \frac{1}{4f}x^2$$

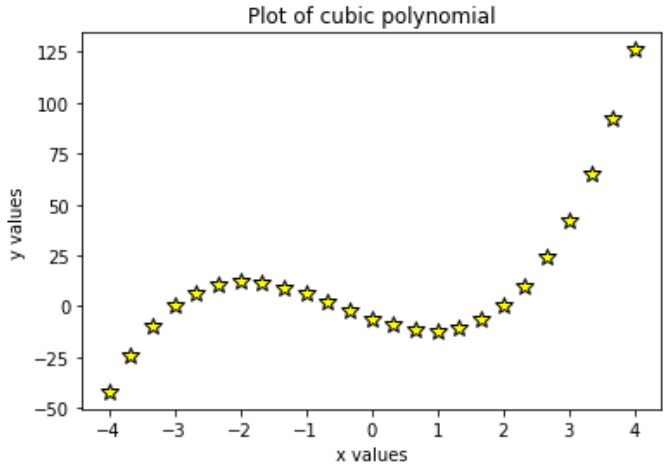
    Write the Python code to plot two parabolas as lines on the same plot for the domain $-2.0 \leq x \leq 2.0$ for $f = 2$ and $f = 6$. Choose different colors for each line. Make the line for $f = 6$ width of 6.0 and the line for $f = 2$ width of 2.0.



    2. Write the Python code to plot (x, y) points for the cubic polynomial

    $$y = 2x^3 + 3x^2 - 11x - 6$$

    for the domain $-4.0 \leq x \leq 4.0$. Twenty-five (25) data points were used for the plot below.



3. Write the Python code to plot $\sin(x)$ and $\cos(x)$ using two subplots with different color lines for the domain $-2\pi \leq x \leq 2\pi$. Display a grid on both plots.