CS-1004, Introduction to Programming
for Non-Majors, A-Term 2018

Setting up Python 3.6.5, matplotlib, and numpy on your Macintosh computer

Hugh C. Lauer©
Teaching Professor
Worcester Polytechnic Institute

Programming assignments in CS-1004 will be carried out using the programming language *Python* — specifically, version 3.5 of *Python*. In addition, you will need several *Python* packages, including one called *numpy* (meaning "Numerical Python") and one called *matplotlib*, a *Python* version of the popular *Matlab* system. This document provides instructions for installing *Python 3.5* and packages such as *numpy* and *matplotlib* on Macintosh platforms.[1]

In general, it is expected that assignments will be compatible among Windows and Macintosh, provided that they all use compatible versions of *Python* and *numpy*. Unlike previous years, we have no prepackaged Linux installation of Python 3.6.5 this year.

> **Note:** There are two different, incompatible sets of versions of *Python* in general use around the world — *Python 2.7* and *Python 3.5*. Significant changes to the *Python* language were made between *Python 2.x* and *Python 3.y* (for all values of *x* and *y*). The *Python 3* language is cleaner, more self-consistent, and more user-friendly. Programs written for versions of *Python 2* will not necessarily run on *Python 3* installations; if they do run, they may get **different answers** to the same problems.

That being said, a lot of legacy *Python 2* code is still in use, and new *Python 2.7* code is still being written and distributed by organizations that have not yet upgraded to *Python 3*. Not all *Python 2* packages have been ported to *Python 3*.

> **Note 2:** All versions of Mac OS X come with Python 2.7 already installed for their own needs. That version of Python must not be deleted or disturbed when installing Python 3.6.5 and its packages. Please follow these instructions exactly to avoid corrupting Python 2.7.

[1] If you have a Windows computer or laptop, please refer to this documents instead:– docx, pdf.

Installing Python 3.6.5 on Macintosh[2, 3]

Installing *Python 3.6.5* on Mac OS X is somewhat simpler than in previous years.[4] However, packages such as *numpy* and *matplotlib* need to be installed from a networked source, just as with *Windows* installations.

First, create a folder on your desktop — for example, *Python-files* — and download the following installer to that folder by clicking on this link:– python-3.6.5-macosx10.9.pkg. Alternatively, you may browse to

http://www.cs.wpi.edu/~cs1004/a18/Resources-A18

and download the file from there.

Open the **`python-3.6.5-macosx10.9.pkg`** file that you just downloaded. This should bring up a dialog like Figure 1 below.

Figure 1

The message indicates that this installer applies to macOS version 10.9 or later. *If you have an earlier version of MacOS* (including a 32-bit version), *please see the Professor for an older version of the Python installation program for the Macintosh.*

If the installer complains that there is already a version of *Python 3* installed on your computer, please seek help or contact the Professor. You must have *Python 3.6.5* for this course. Earlier versions should be upgraded. Click *Continue* to bring up another dialog:–

---

2 It is useful to print out the relevant section of this document. If you read it on-screen, the dialog boxes of the installation tend to obscure the instructions of the document, just when you need them the most!

3 These instructions have been tested on MacOS X 10.13.3 High Sierra. There may be slight differences with other versions of MacOS.

4 In particular, an additional package named *Tcl/Tk* is now built into the *Python* installation package and therefore does not have to be installed separately.

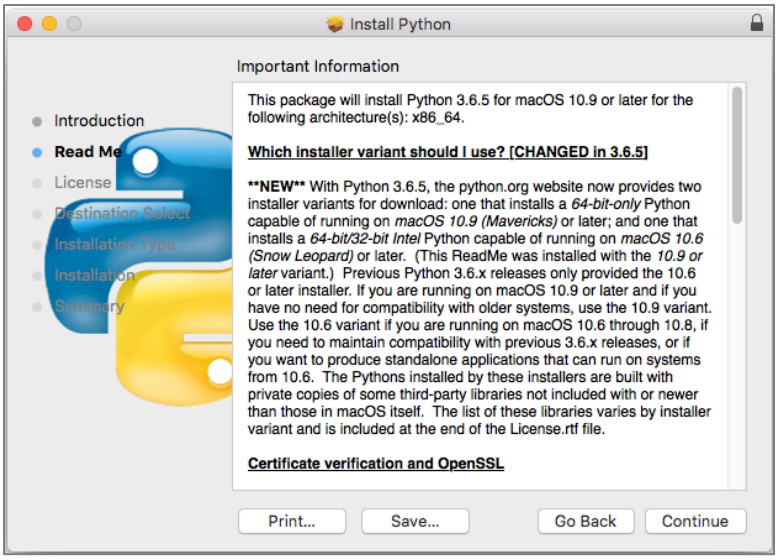

Figure 2

This dialog box is scrollable and contains information about what is new in this release. Click *Continue* to bring up one or more additional dialog boxes pertaining to the history of *Python* and its license. Continue clicking until the dialog in Figure 3 appears.

Figure 3

Click *Agree* to continue the installation. The installation itself will take several minutes and may show one or more dialog boxes. It will finish with a panel resembling Figure 4.

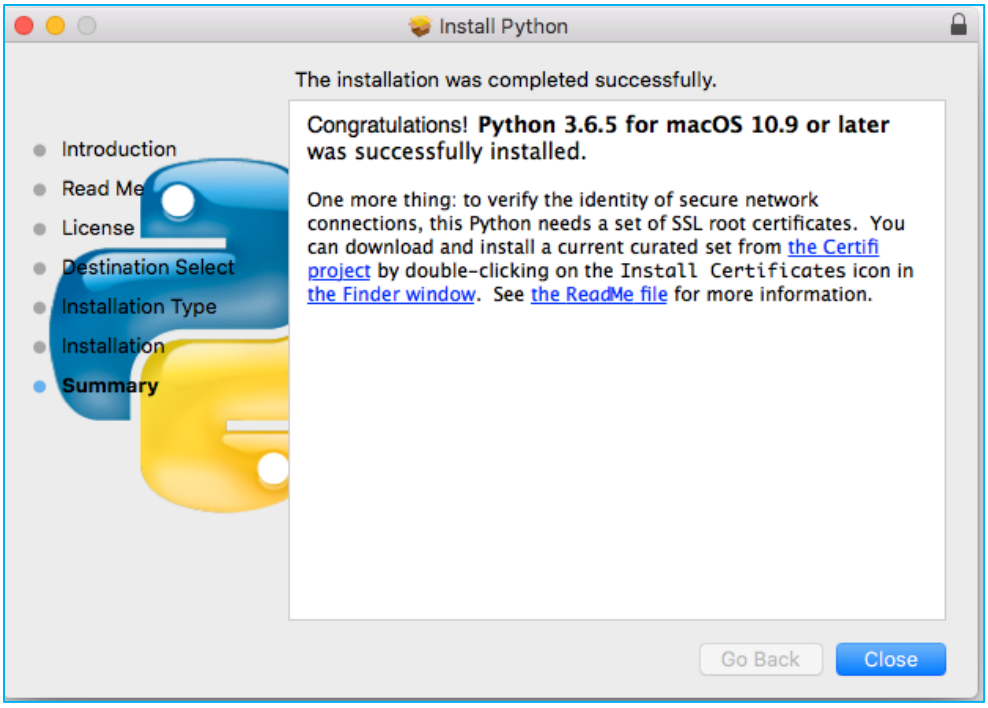

Figure 4

After it has completed, you should find a folder named *Python 3.6* in your *Applications* folder, as shown circled in dotted black below.

Figure 5

Open the *Python 3.6* folder to show Figure 6 below.

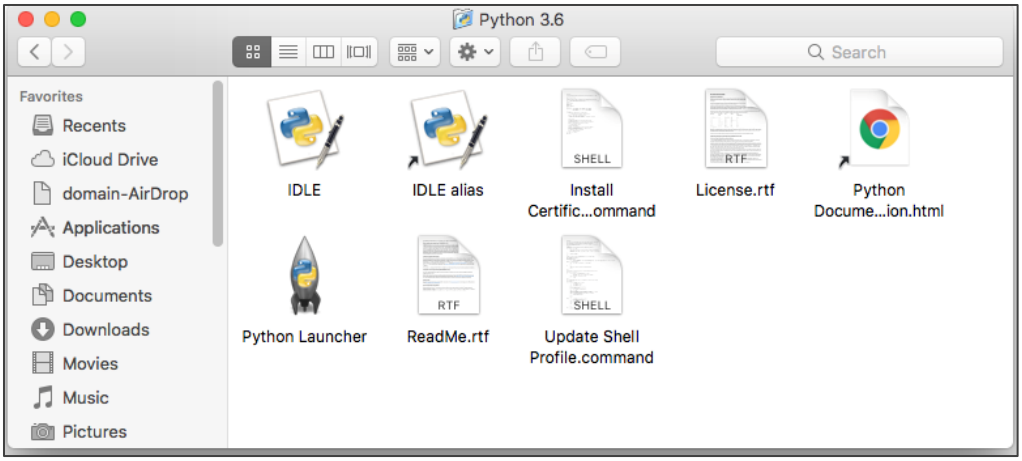

Figure 6

Drag the *IDLE Alias* in the first row to a convenient place so that you can access it easily — for example, on your desktop. Double-click this alias to bring up the following window:–

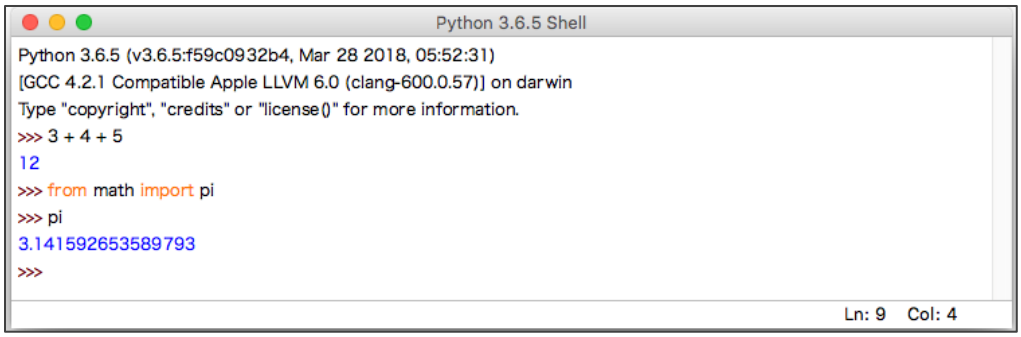

Figure 7

This is *IDLE*, the *Python* command prompt and graphical user interface. *This is where we will start all programs and projects in this course.*

For now, simply type any *Python* statement or expression after the ">>>" prompt. For example, Figure 7 shows the expression *3 + 4 + 5* as typed, followed by *Python's* response with the value *12*.

After the next two ">>>" prompts, type the command

```
from math import pi
```

and the expression

```
pi
```

Python responds by printing the value of *pi* to 15 decimal places, also shown in Figure 7

Continue testing by typing out the code on pages 10-11 of the textbook, just to make sure that your installation works as expected. Note that if you mistype something, you will probably get some unexpected behavior. Simply try again. If you get really lost, type CTRL-C to get back to the IDLE prompt.

Note: IDLE is essentially the same across *Windows*, *Macintosh*, and *Linux* platforms. It will be used the same way in all three for this course.

Installing graphics.py

Graphics.py is a simple drawing package that we will use a lot in this course. It was written in *Python 3* and created by the textbook author for making simple drawings. To install it, click on this link — graphics.py — and download the file *to the folder where you keep your Python programs*. Follow the instructions in sections 4.3-4.8 of the textbook (3rd edition).

Installing *matplotlib*, *numpy*, and other packages

These steps require you to have a working internet connection.

One of the many benefits of *Python* is the vast number of third-party packages that can be downloaded and used by your *Python* programs. Many of these are open-source and free. For this course, we will use at least the following:–

- *matplotlib* (a package for creating 2D plots and graphs similar to *Matlab*),
- *numpy* (meaning "Numerical Python," a package for efficient handling of large arrays of numerical data), and

Installing *matplotlib*, *numpy*, and other packages on a Macintosh requires you to type Linux-like commands in a *Terminal* window.

In the *Applications* folder of Figure 5, scroll down until to you see the *Utilities* folder, as shown in Figure 8. Open this *Utilities* folder, as shown in Figure 9 on the next page.

Figure 8

Figure 9

Scroll down the *Utilities* folder until you an icon labeled *Terminal*, as shown on the right side of Figure 9.

Double-click (or open) this icon to bring up a *terminal window* (also known as a *command shell* in Unix and Linux terminology). An example terminal window is shown in Figure 10. In this window, the system prints a *prompt* starting at the beginning of a line and ending in '$'. After the prompt, you type a *command*, consisting of a command name followed by zero or more *operands*, which control what the command does. When you terminate the command with the *Enter* key, the system performs the command.

A *command* may work silently and then type out its results in the same window, or it may engage in a textual conversation with you, requiring you to respond, or it may open its own window with its own graphical user interface. When the command has completed, the system prints a new prompt for the next command.

Installing *matplotlib*, and *nose*

In your terminal window, type the following command lines, one at a time, exactly as shown:--

```
pip install -U pip
pip install -U matplotlib
pip install -U nose
pip list
```

The first command will update *pip* (the *Python Installation Program*) if necessary. For many students in this course, it is likely that *pip* is already up-to-date; in this case, *pip* will indicate this fact.

The second command downloads and installs *matplotlib*, along with all its dependencies, including *numpy*. The third command downloads *nose*, the *numpy* test program. The final command lists the installed Python packages. It should produce a result resembling the following figure:--

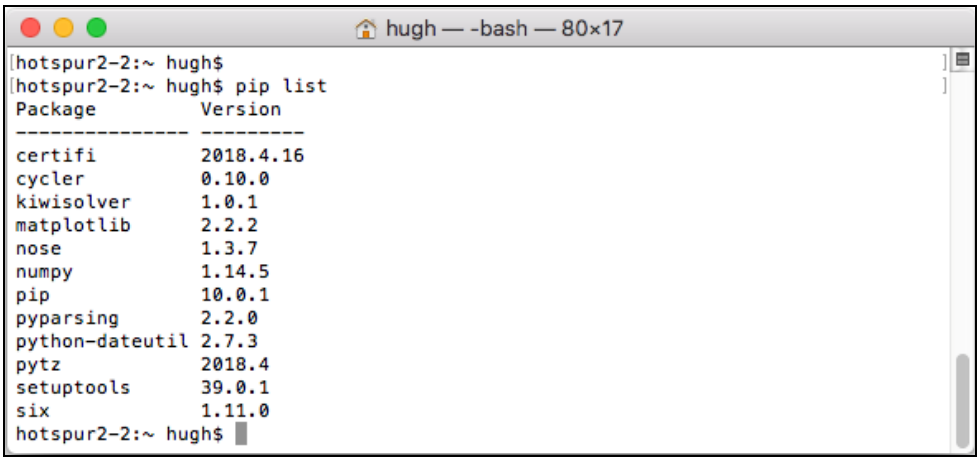

Figure 10

You can test your installation of *numpy* by opening an *IDLE* window, as in Figure 11. Type or paste the following commands into IDLE, one line at a time, *exactly* as written.

> **Note:** In these lines, the word **`version`** is preceded by *two* underscore characters and followed by two more underscore characters.
>
> The word **`arange`** is spelled with one "r" — it is a variation of the **`range`** function that we will learn about early in the course.

```
import numpy as np

np.__version__

a = np.arange(10)
a

b = np.arange(1, 9, 2)
b

c = np.eye(3)
c

d = np.diag(np.array([1, 2, 3, 4]))
d
```

After each line **`a`**, **`b`**, **`c`**, and **`d`**, *Python* will print the values of these variables. The result should resemble Figure 11 below:–

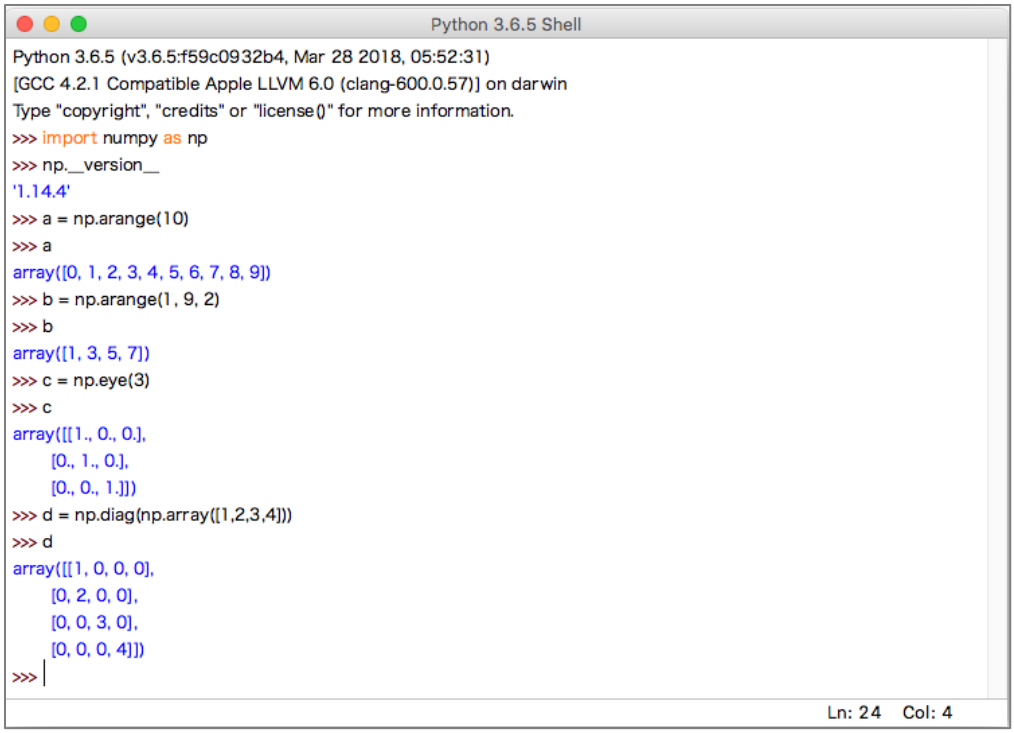

Figure 11

Congratulations! You have successfully installed and tested *numpy 1.14.4*. However, we will run another test below.

To test your *matplotlib* installation, type or paste the following commands into IDLE, *one line at a time*, exactly as written:–

```
from matplotlib import pyplot
pyplot.plot([1, 2, 3, 4], [1, 4, 9, 16])
pyplot.show()
```

The IDLE window should look something like the following:–

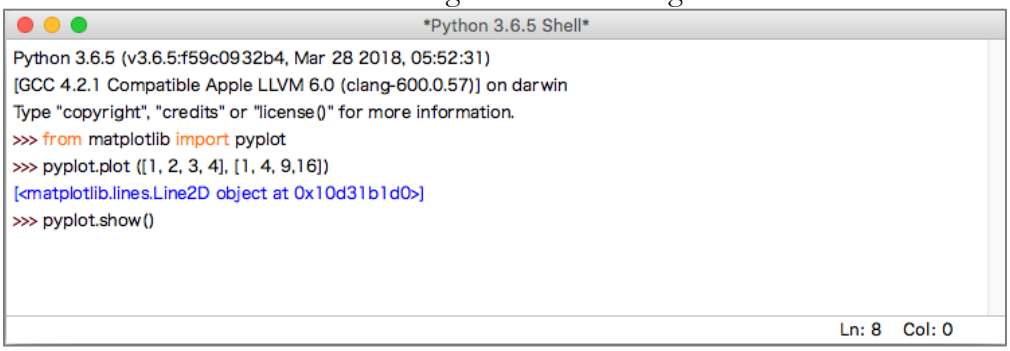

Figure 12

After you type the **ENTER** key following the last line, the following window should appear:–

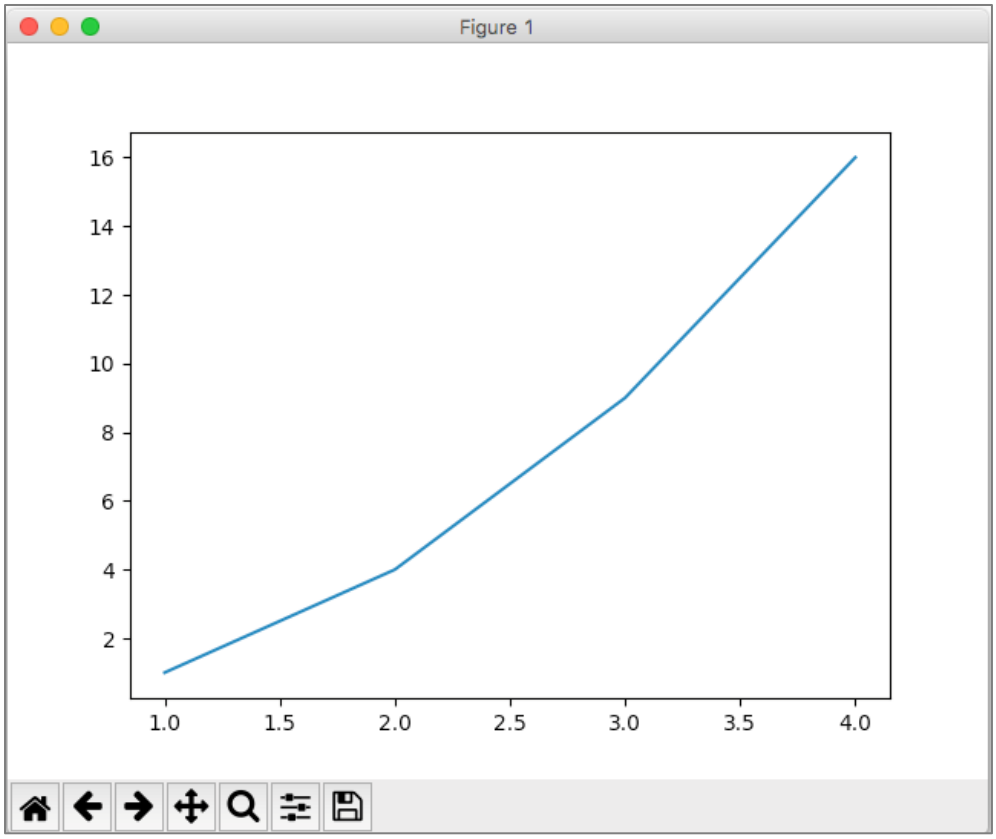

Figure 13

To close this window, click on the Macintosh "close" button in the upper left corner.

For a more interesting test, download the following file to a convenient folder and save it as a **`.py`** file:–

TestMatplotlib2.py

Then, use the *File* menu in the *IDLE* window to open this file in its own window, as shown in Figure 14:–

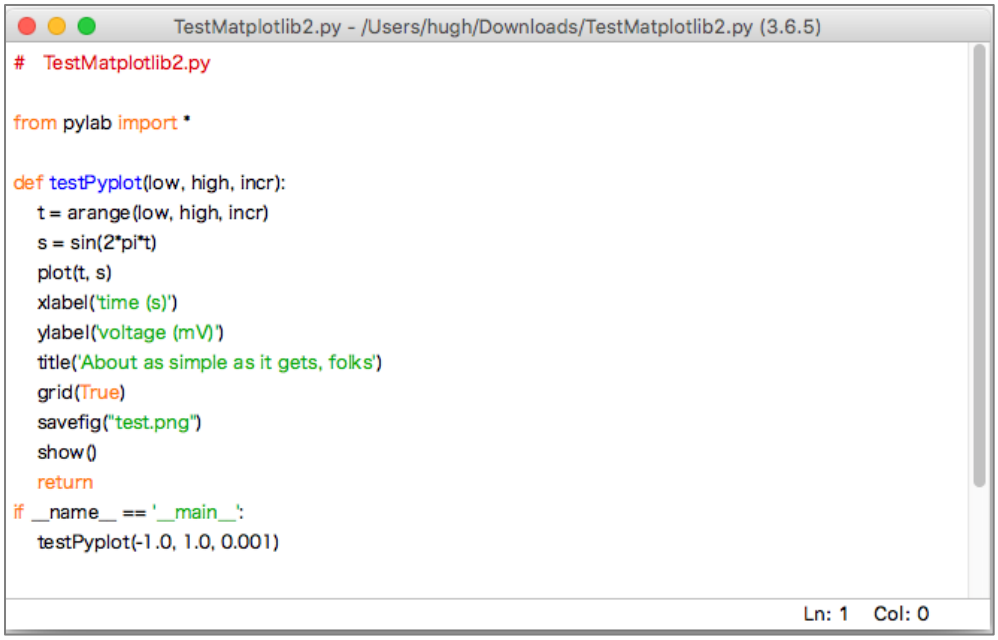

Figure 14

Click on this window to make it the active window, and then select the *Run* > *Run Module* command in the IDLE menu at the top of the Macintosh screen. This will cause the *Python* program to run and to produce the window of Figure 15.

This tests not only *matplotlib* but also *numpy* (in the background). The result should be a window like this:–

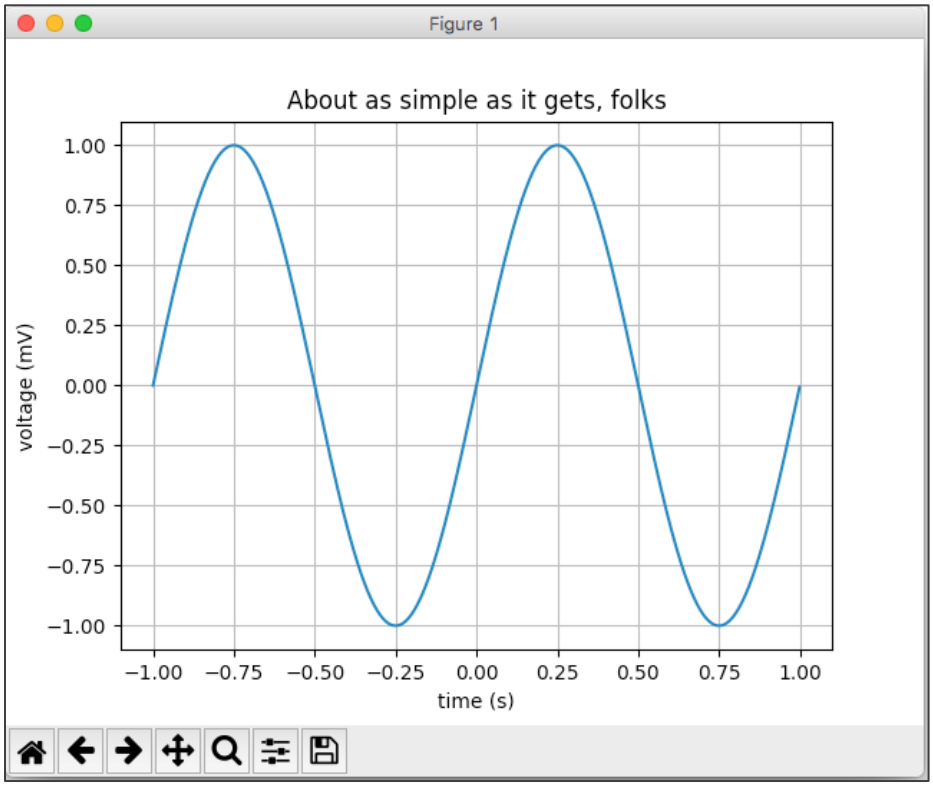

Figure 15

Congratulations! You now have a working version of *matplotlib* installed. As part of this test, a copy of the picture was saved under the name **`test.png`** in the current directory.

Note also that *matplotlib* used *numpy* in the background to build up the array of points making up the graph.

For a final test, type the following into an *IDLE* window:–

```
import numpy as np
np.test()
```

This will run a complete set of tests of the *numpy* package for several minutes, producing a lot of output. The output begins with the lines in Figure 16 and continues for some time. At times, it may seem like it has paused or stopped. Please be patient.

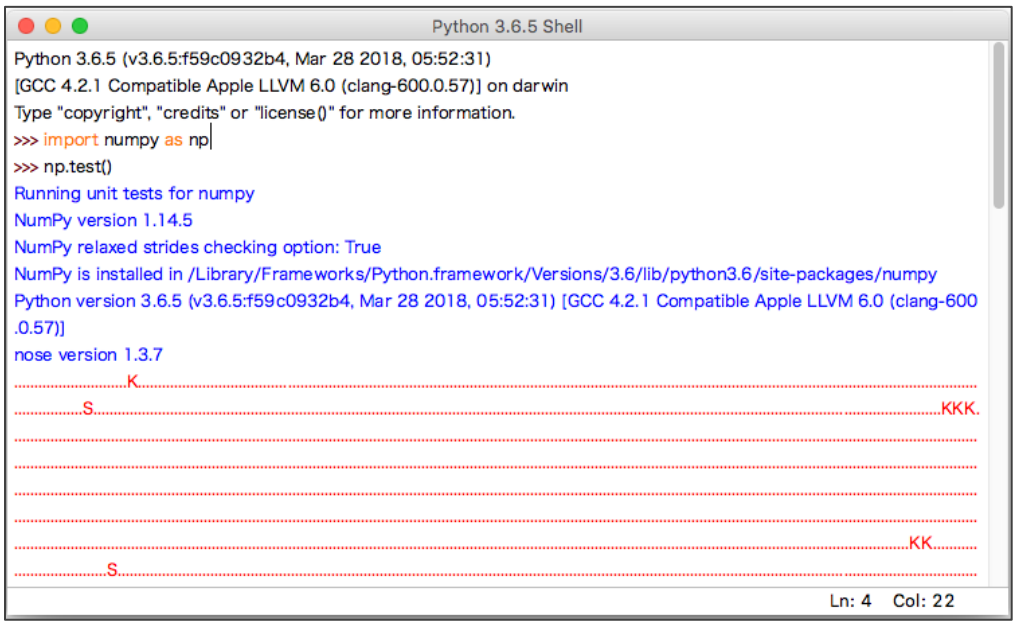

Figure 16

Eventually, it will finish with some "apparent" errors as shown in Figure 17 below. You may ignore these errors.

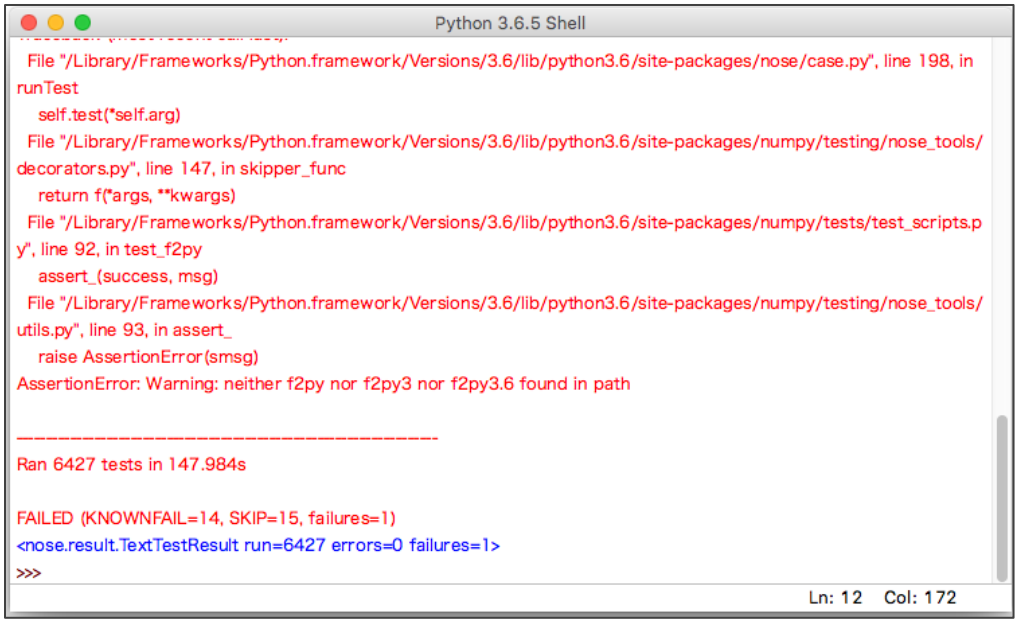

Figure 17

The last line shows that 6427 tests were run with one failure. This is considered success.

Congratulations! You now have both *matplotlib 2.2.2* and *numpy 1.14.4* working.