Tools and Mindset Needed for these Modules

"Tell me and I'll forget; show me and I may remember; involve me and I'll understand.

-Reputably a Chinese proverb

What You Need to Know About Python

Installing and Learning to Use VPython

VPython is a programming language written by scientists specifically to be used to visualize physical phenomena. One of VPython's strengths is that compared to other programming languages it is fairly easy to learn to use and in a short period of time you can be creating some fairly sophisticated 3D simulations and visualizations.

VPython itself is actually a module library that can be imported into the Python programming language. Python itself is a popular and powerful object-oriented programming language[1] used in a variety of fields including math, finance, physics, engineering and computer science.

Downloading and installing VPython on your computer:
Python and VPython are both open source programming languages and thus are available for everyone to freely use. You can download VPython at http://www.vpython.org. When you install these programming languages on your computer, it is very important that you follow closely the instructions. Make sure that you FIRST download AND install Python *BEFORE* you download and install VPython.

Learning to use VPython:
Once you install VPython a program editor called VIDLE should appear on your desktop and/or in your program list. Once you open VIDLE you can immediately write and execute programs. Go ahead and enter the program that follows:

```
from visual import *
scene=display(title="My first program",x=5,y=5,
              width=500,height=300,
              range=vector(10,10,10),
              center=vector(0,0,0))
myfirstbox = box(length=3,height=1,width=2, color=color.red)
```

Note that VPython does not care much about spaces within a command line, but TABS do matter a great deal. Thus, make sure that you follow the indented lines as shown above.

[1] A computer program can be written in a variety of languages. The primary focus of object oriented languages is the creation of an object, its properties and the creation of object classes. Other popular object oriented languages include C++, Java.

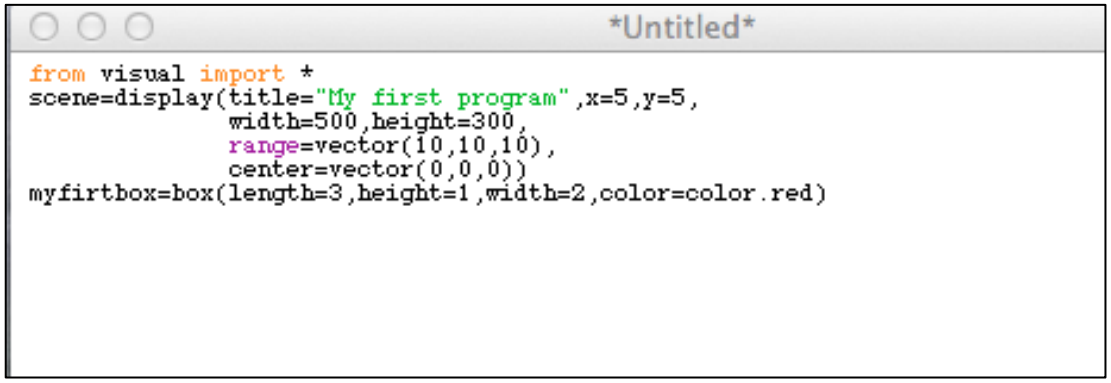


Figure 1 Screen shot (Mac)

Once you enter the above program you can run the program by selecting "Run" and "Run Module" or by pressing the function key F5.

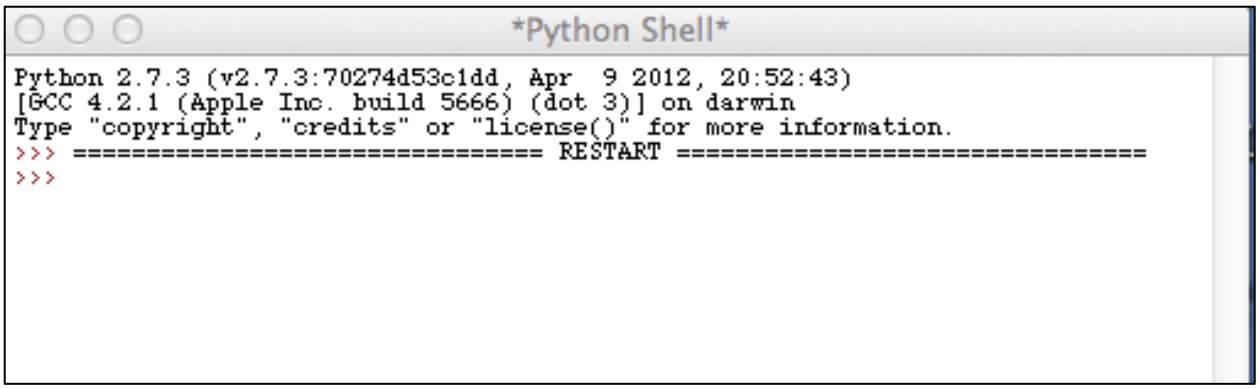
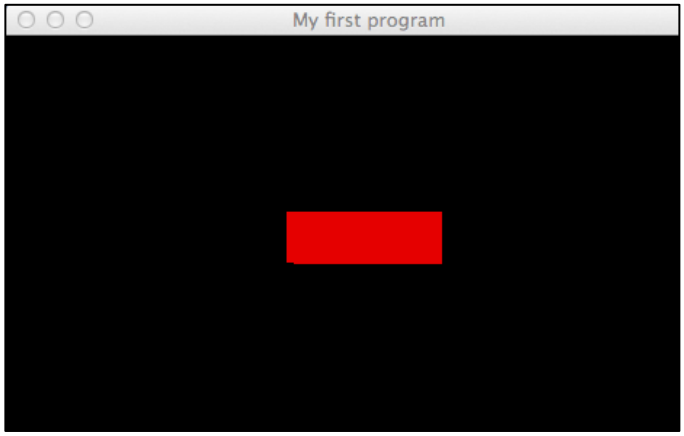


Figure 2 Screen showing running of the program (Mac)

When you run the program, another box should open containing a black background with a red box in the middle of the screen (Figure 3).



Once you select this window, hold down the left+right buttons on a two button mouse (or the option key on Macintosh) and move the mouse. You should see that you are able to zoom into and out of the scene. Next, try holding down the right mouse button (or hold down the Apple command key on Macintosh). You should find that you are able to rotate around the box.

Figure 3 Screen shot (Mac)

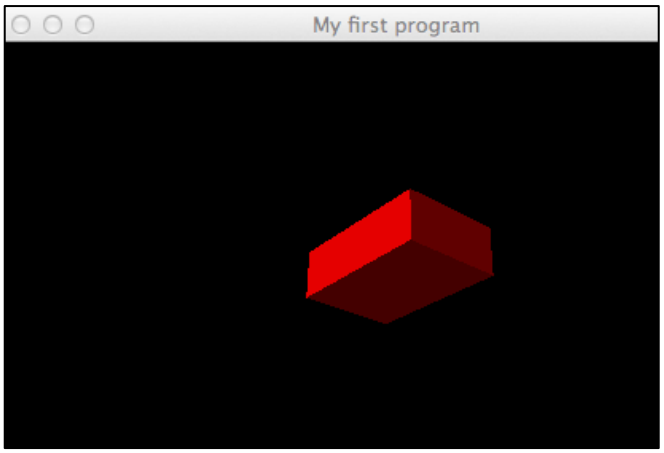


Figure 4 Screen shot (Mac)

To see more example of VPython program you can go to the menu and select "File" and "Open" right after opening VIDLE. You should find yourself in a directory named "examples" where you should be able to find lots of VPython sample programs to run.

More documentation and a tutorial can be found at http://www.vpython.org .

Examples of VPython Objects:

If you want to play some more, check out the "VPython Objects" at http://www.vpython.org/contents/docs/index.html. Below are two examples of numerous 3D objects that you might want to use when making a VPython program.

Here is a very simple Python program `Area.py` that calculates the area of a circle. Note that when a Python program such as `Area.py` is stored in a file, the convention is to have the file name end with the suffix .py so that the computer and you know that this is Python *source code*. The word "source" is used here to distinguish a program written in a language that people can understand from *executable* code, which is the language that computers can understand. When you run this program with IDLE it gets translated into *executable* code.

You should try entering, running and then modifying it.

```
# Area.py: Area of a circle, simple program
from math import pi
radius = 1.
circum = 2. *pi*radius
area = radius * radius *pi
print ('Radius = ', radius)
print ('Circumference = ', circum)
print ('Area = ', area)
```

Use IDLE to modify the program so that you can try different values for the radius.

Here is another simple program `EasyVisual.py`.

Try entering and running it, and see if you get the two graphs in Figure 5. You can modify this file for your own purposes too.

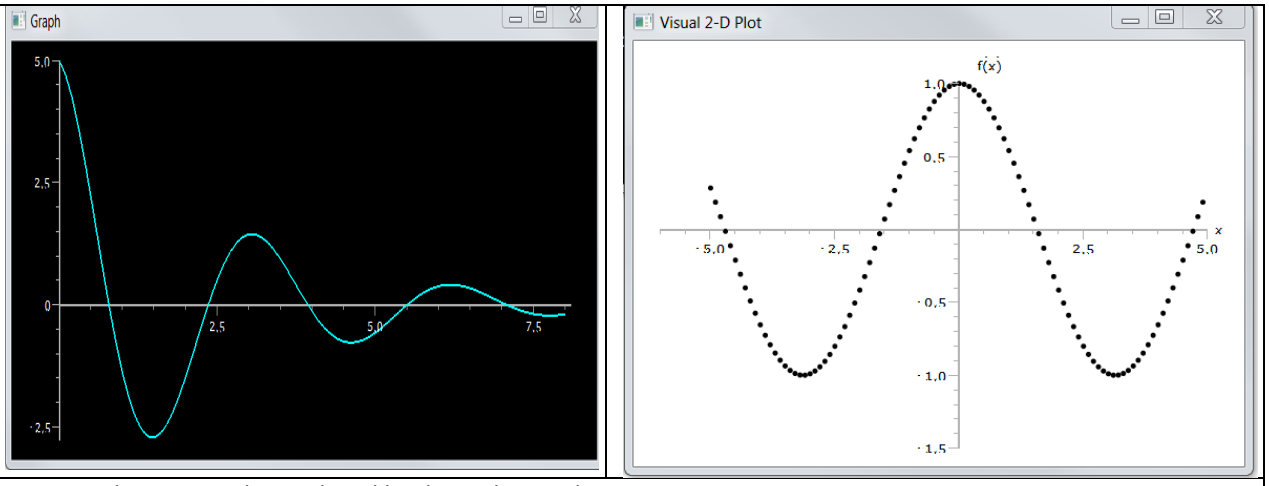


Figure 5. The two graphs produced by the Python code `EasyVisual.py`.

```
# EasyVisual.py:  Simple graph object using Visual package
from visual.graph import *                              # Import Visual
funct1 = gcurve(color = color.cyan)              # Connected curve object
for x in arange(0., 8.1, 0.1):                              # x range
    funct1.plot( pos = (x, 5.*cos(2.*x)*exp(-0.4*x)) )      # Plot points
graph1 =  gdisplay(x = 0, y = 0, width = 600, height = 450, # Second plot
          title = 'Visual 2-D Plot', xtitle = 'x', ytitle = '  f(x)',
          xmax = 5., xmin = - 6., ymax = 1, ymin = - 1.5,
          foreground = color.black, background = color.white)
plotObj = gdots(color = color.black)                              # Dots
for x in arange( -5.,  +5, 0.1 ):
    plotObj.plot(pos = (x, cos(x)))
```

Note that text following the # sign on any line is a comment that is ignored by the computer.

A more complete tutorial on Vpython can be found at http://www.vpython.org/contents/doc.html .
A full tutorial on the Python language can be found at http://docs.python.org/2/tutorial/ .
A free Python textbook for beginners can be found at http://www.greenteapress.com/thinkpython/ .

What You Need to Know About Excel

The Area.py and EasyVisual.py programs have been implemented in Excel for you. Download the Excel implementation file from the Intro module web page. This Excel workbook contains three worksheets: *AreaCircle, EasyVisual* and *Steps*.

A typical workflow of developing a computational model in Excel is:

1. **Generating Data.** Data might come from your experiments, a publication or someone else's model. In Excel, Data is usually entered in columnar format.
    a. It is customary to have a first column with a number of iterations (compare it to Python code `for x in arange(0, 100, 1)`.
    b. Then there is a column for x-axis values which might depend (or not) on the number of iterations.

   c. Then there might be one or more columns (y-axis values) where a dependent variable(s) is calculated. One of these dependent variables is your model (your expectation of how y depends on x).
2. **Visual Representation of Data**. *Steps* worksheet in the Implementation file contains selected screenshots of the workflow during development of the visual representation (Chart) of the data.

Excel Help is a very powerful resource. There are plenty of Excel tutorials and demos on Web. We recommend these two at the moment:

1. TutorialReference from the http://academic.pgcc.edu/~ssinex/excelets/ web-site.
2. http://www.shodor.org/tutorials/excel/IntroToExcel is a great example of creating a computational model in Excel. We recommend that you work your way through it.

Of course you cannot do this unless you have Excel running on your computer to do this properly.

What You Need to Know About Vensim

There is a Vensim tutorial at http://www.shodor.org/tutorials/VensimIntroduction/Preliminaries and we recommend that you work your way through it to learn about Vensim. Of course you cannot do this unless you have Vensim running on your computer to do this properly.