Short Python Usage Summary

- standard includes
  - from visual import *
  - from visual.graph import *
  - from visual.factorial import *
  - from math import *
- standard math functions
  - x=x**5 $\rightarrow x = x^5$
  - x*=5 $\rightarrow x = x * 5$
  - x+=5 $\rightarrow x = x + 5$
  - factorial(N) = N!
  - combin(a,b) = $\frac{a!}{b!*(a-b)!}$
- vector and functions on vectors
  - V=vector(1,1,1) $\rightarrow \vec{V}$
  - mag(V) $\rightarrow |\vec{V}|$
  - norm(V) $\rightarrow \hat{V}$
  - dot(V1,V2) $\rightarrow \hat{V}1 \cdot \hat{V}2$
  - cross(V1,V2) $\rightarrow \hat{V}1 \times \hat{V}2$
  - V.x $\rightarrow V \cdot \vec{i}$
  - V.y $\rightarrow V \cdot \vec{j}$
  - V.z $\rightarrow V \cdot \vec{k}$
  - V.mag $\rightarrow |V|$
    changing this changes magnitude, not direction
  - rotate(V,angle, axis) $\rightarrow$ axis is a vector
- arrays/list
  - array = []
  - array = $[1, 2, 3, 4]$
  - array = range(2,10,2) $\rightarrow [2, 4, ...]$
  - array = arange(2,10,0.1) $\rightarrow [2, 2.1, ...]$, (allows floating point)
  - array[5] = 6
  - array.append(5)
  - len(array) $\rightarrow$ size of array
- loops
  - for x in array: x= array[1] then array[2] then ...
  - for x in xrange(0,200000): $\rightarrow$ doesn't overflow memory
- defining functions
  - ```
    def function_name(var1,var2,var3):
        [some normal code]
        return [variable or constant]
    ```

VISUAL OBJECTS:

- all visual objects have attributes
  - .visible=1 (1 is visible, 0 is not)
  - .pos=(0,0,0) (position in x,y,z coordinates)
  - .color=(0,1,0) (color as a tuple)
  - NOTE - any object attributes can be passed as parameters on invocation such as sp=sphere(color=(0,1,0))
- sphere
  s=sphere()
  - s.radius
- arrow
  a=arrow()
  - a.axis
  - a.width
- box
  b=box()
  - b.height=1
  - b.width=1
    ```

  - b.length=1
  - b.axis=(0,0,0), specifies orientation of the length of the box
  - b.up=(1,0,0), specifies orientation of height (still perpendicular to length)
- curve
  c=curve()
  - c.pos = $[(0,0,0),(0,0,1)....] \rightarrow$ list of positions on curve
  - c.radius $\leftarrow$ thickness of line
  - c.x, c.y, c.z $\rightarrow$ [0,1,0.5] $\rightarrow$ list of components of pos
  - c.color
  - c.append(pos=(0,0,0)) $\rightarrow$ c.append() c[len(c)].pos=(0,0,0)

GRAPHING

- gdisplay
  - gd=gdisplay()
  - gd.color, gd.x, gd.y, gd.width, gd.height, gd.title, gd.xtitle, gd.ytitle, gd.xmax, gd.ymax, gd.xmin, gd.ymin, gd.forground, gd.background
  - note - if you leave out xmin,ymin,xmax,ymax will autoscale for you. This will belong to whatever gdisplay was last invoked or you can set

    ```
    foo.gdisplay=my_gdisplay
    ```

    to speficy a particular display
- gcurve
  - gc=gcurve()
  - gc.color
  - gc.plot(pos=(x,y),color=foo)
- gvbars
  - gv=gvbars()
  - gv.plot(pos=(x,y))
  - gv.delta
- ghist
  - gh=ghist()
  - gh.accumulate
  - gh.average
  - gh.bins=arange(0,50,0.1)

TIPS

syntactic tips:

- can add properties to objects
- s=sphere()
- s.mass = 5
- 1/2 $\rightarrow$ 0 you want 1.0/2.0 $\rightarrow$ 0.5
- all (a,b,c) type things should be of the form vector(a,b,c) this can't hurt and can only help
- (yes pos=(a,b,c) works, but that's special)

style tips:

- use variables instead of constants for things like diameters or number of particles
- set them at the beginning of your program this way you can adjust them in only ONE place
- don't copy code, make a loop instead