

DM2212 Programming Physics



FALLING APPEES
MINDPHASE

ML: Dioselin Gonzalez
2007 S1



Python and VPython

Based on tutorial and examples from

Ralph Noack. "[An Incremental Introduction to Python](#)"

Michael Williams. "[Handbook of the Physics Computing Course](#)"

Image by chocoloco.deviantart.com

What and why Python?

- General purpose, object oriented programming language
- Interpreted
- Open source
- Used in many domains and applications, including
 - Autodesk® Maya®, MotionBuilder™
 - Production pipeline at ILM
 - CCP Games' EVE Online
 - FIRAXIS Games' Civilization IV

C++ vs Python

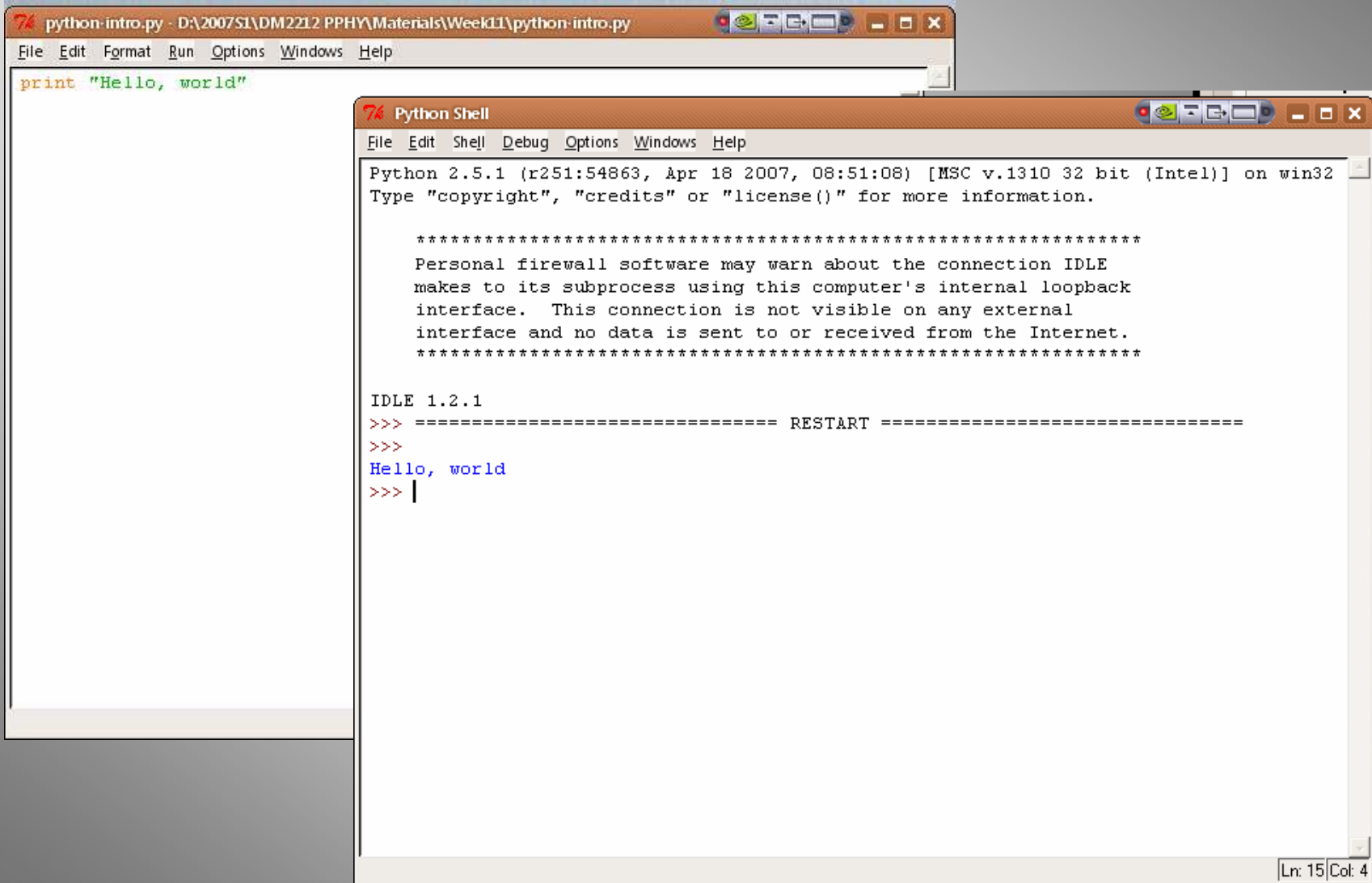
- Hello.cpp

```
#include <iostream>
using namespace std;
void main() {
    cout<<"Hello, world"<<endl;
}
```

- Hello.py

```
print "Hello, world"
```

IDLE



The image shows two windows from the IDLE Python IDE. The top window, titled 'python-intro.py', contains the code `print "Hello, world"`. The bottom window, titled 'Python Shell', shows the output of running the code. It displays the Python version (2.5.1), a warning about a firewall connection, and the output 'Hello, world'.

```
python-intro.py - D:\2007S1\DM2212 PPHY\Materials\Week1\python-intro.py
File Edit Format Run Options Windows Help

print "Hello, world"

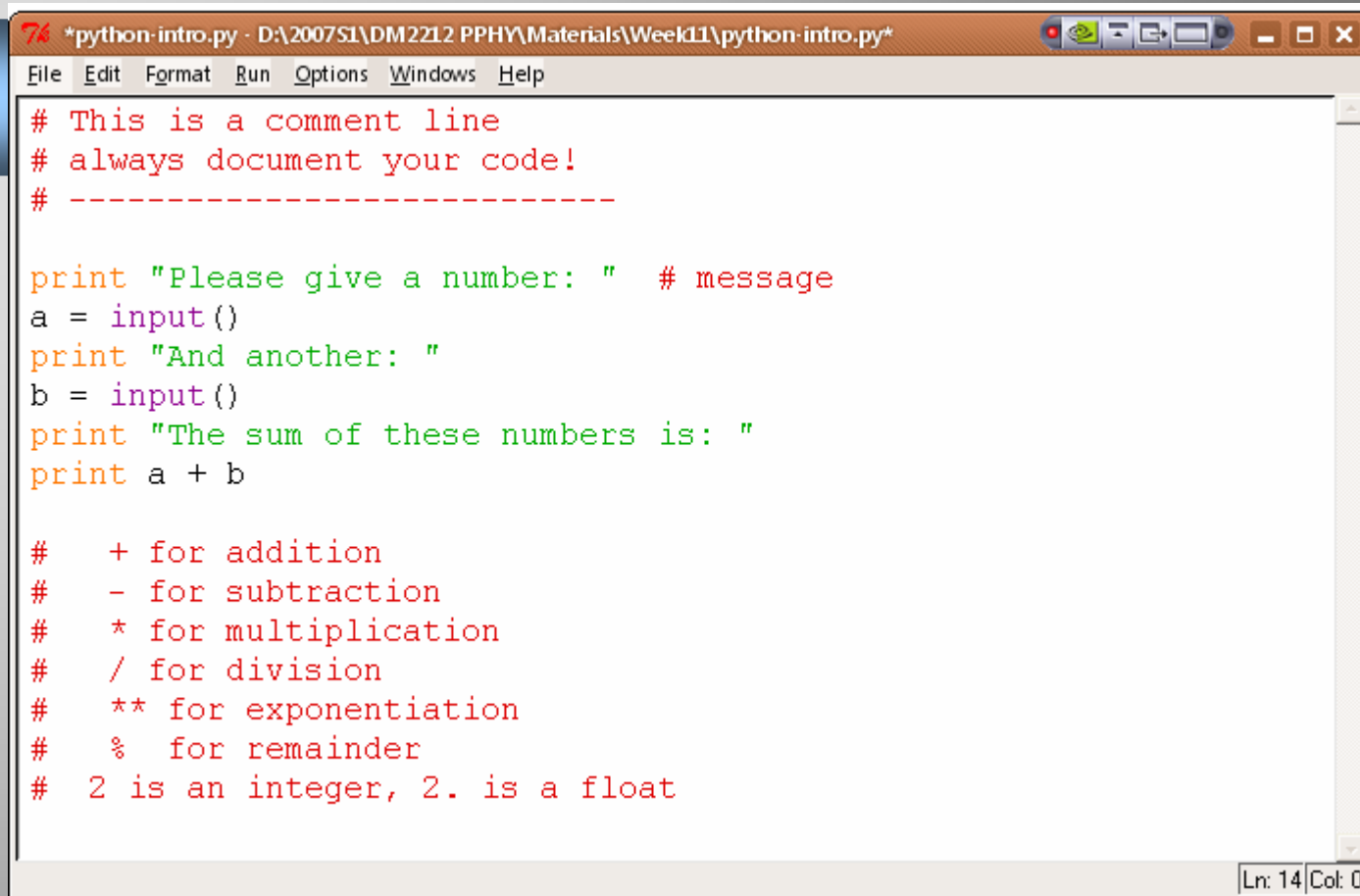
Python Shell
File Edit Shell Debug Options Windows Help

Python 2.5.1 (r251:54863, Apr 18 2007, 08:51:08) [MSC v.1310 32 bit (Intel)] on win32
Type "copyright", "credits" or "license()" for more information.

*****
Personal firewall software may warn about the connection IDLE
makes to its subprocess using this computer's internal loopback
interface. This connection is not visible on any external
interface and no data is sent to or received from the Internet.
*****

IDLE 1.2.1
>>> ===== RESTART =====
>>>
Hello, world
>>> |
```

Ln: 15 | Col: 4



```
*python-intro.py - D:\2007S1\DM2212 PPHY\Materials\Week11\python-intro.py*
File Edit Format Run Options Windows Help
# This is a comment line
# always document your code!
# -----

print "Please give a number: " # message
a = input()
print "And another: "
b = input()
print "The sum of these numbers is: "
print a + b

# + for addition
# - for subtraction
# * for multiplication
# / for division
# ** for exponentiation
# % for remainder
# 2 is an integer, 2. is a float
Ln: 14 Col: 0
```

1. Type and run this program
2. What happens if you enter “hola” and “:-)” as parameters?

Variables

- Dynamic typing: variables need not be declared (see 'duck-typing' in python's glossary)

- Created when first assigned

```
>>> a="hey"  
>>> print a  
hey
```

- cAsE SEnsItiVE

Variables

- Integers and floats

```
>>> myint=3
```

```
>>> myfloat=3.0
```

- Strings

```
>>> mystr="hola"
```

```
>>> myotherstr='hey'
```

```
>>> print mystr[0]
```

```
>>> print len(mystr)
```

```
>>> mystr[0:2]
```

Variables

- Strings

```
>>> mystr="123456"
```

```
>>> mystr[:1]
```

```
>>> mystr[3:]
```

```
>>> mystr[-1]
```

```
>>> mystr[-6:-1]
```

```
>>> "an integer %d a floating point number %f" % (1,2.0)
```

Variables

- Lists
 - comma separated items between brackets
 - Items don't have to be the same type
- Exercise: Create and print a list with integer, float and string items

Variables

- Lists are mutable
 - Can assign a new value to an item
 - Append, insert, delete and more operations:
 - `mylist.append("new item at the end")`
 - `mylist.insert(2, "new item at pos 2")`
 - `del mylist[1]`
- Tuples
 - Immutable

Variables

- Dictionaries
 - Also called hash table
 - Indexed by a key

Key	Value
"Venezuela"	26
"Singapore"	4
"China"	1321

```
>>>myD = {} #an empty dictionary
>>>myD["Singapore"] = 4
>>>myOtherD={"China":1321, "Venezuela":26}
>>>print myOtherD.keys()
```

Blocks of code

- Indentation
- New block is preceded by a statement ending with a colon character
- C++

```
if (a==b) {  
    //code;  
    //more code;  
}
```

- Python

```
if (a==b) :  
    #indented code (4 is the std width!)  
    #more code
```

If statement

```
if <conditional>:  
    <block of code>  
elif <conditional>:  
    <block of code>  
else:  
    <block of code>
```

```
#conditionals: a == b, a!=b, a<b, a<=b, a>b, a>=b
```

Exercise: Write a program to ask for the distance traveled and time taken for a journey. If they went faster than some suitably dangerous speed, warn them to go slower next time.

While statement

```
while <conditional>:  
    <block of code>  
else:  
    <block of code>
```

'break' statement will cause the loop to terminate and the else block is not executed

'continue' will transfer execution back to the top

For statement

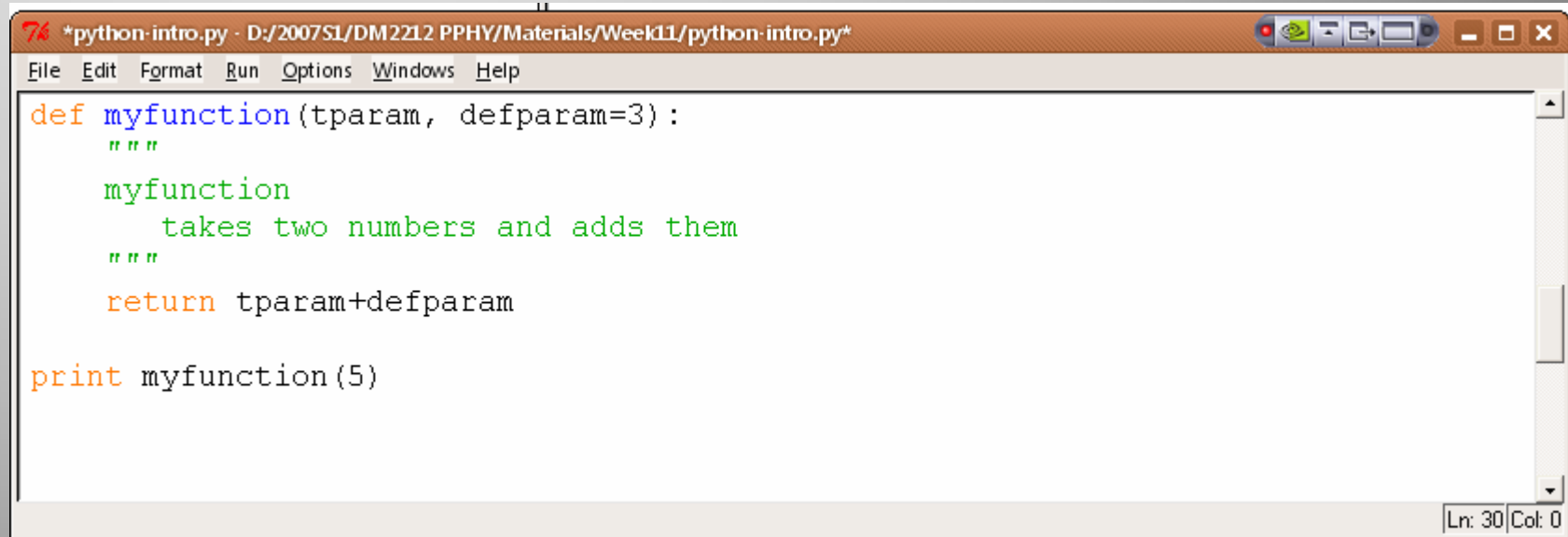
```
for <target> in <sequence>:  
    <block of code>  
else:  
    <block of code>
```

#range() function commonly used

#'break' statement will cause the loop to terminate and the else block is not executed

#'continue' will transfer execution back to the top

Functions



```
*python-intro.py - D:/2007S1/DM2212 PPHY/Materials/Week11/python-intro.py*
File Edit Format Run Options Windows Help
def myfunction(tparam, defparam=3):
    """
    myfunction
    takes two numbers and adds them
    """
    return tparam+defparam

print myfunction(5)
Ln: 30 Col: 0
```

Exercise: Write a function that takes a list or tuple as parameter, adds all elements in the list and returns the final sum

Importing modules

```
*python-intro1.py - D:/2007S1/DM2212 PPHY/Materials/Wee
File Edit Format Run Options Windows Help
import myPPHYlib

if __name__ == "main":
    print myPPHYlib.myfunction(5)
```

```
*python-intro1.py - D:/2007S1/DM2212 PPHY/Materials/Wee
File Edit Format Run Options Windows Help
from myPPHYlib import *

if __name__ == "main":
    print myfunction(5)
Ln: 8 | Col: 0
```

Classes

```
*python-intro1.py - D:/2007S1/DM2212 PPHY/Materials/Week11/python-intro1.py*
File Edit Format Run Options Windows Help

class myPPHYclass:
    def __init__(self, mass, color="blue"): #the "constructor"
        self.mass = mass
        self.color = color
        self.pos = [0,0,0]
    def move(self, x, y, z): # a method
        self.pos[0] += x
        self.pos[1] += y
        self.pos[2] += z

mobj = myPPHYclass(30)
mobj.move(2,1,3)
print self.pos

Ln: 17 Col: 0
```

VPython

```
76 bounce2.py - C:\Python25\Lib\site-packages\visual\examples\bounce2.py
File Edit Format Run Options Windows Help
from visual import *

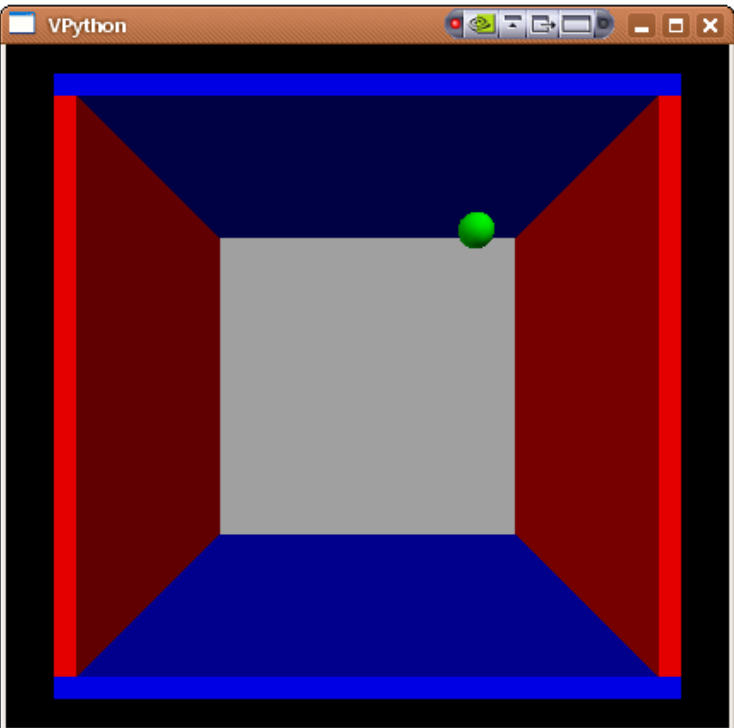
print """
Right button drag to rotate "camera" to view scene.
Middle button to drag up or down to zoom in or out.
    On a two-button mouse, middle is left + right.
    On a one-button mouse, middle is CTRL + mouse.
"""

side = 4.0
thk = 0.3
s2 = 2*side - thk
s3 = 2*side + thk
wallR = box (pos=( side, 0, 0), length=thk, height=thk, color=red)
wallL = box (pos=(-side, 0, 0), length=thk, height=thk, color=red)
wallB = box (pos=(0, -side, 0), length=s3, height=thk, color=blue)
wallT = box (pos=(0, side, 0), length=s3, height=thk, color=blue)
wallBK = box(pos=(0, 0, -side), length=s2, height=thk, color=gray)

ball = sphere (color = color.green, radius = 0.4)
ball.mass = 1.0
ball.p = vector (-0.15, -0.23, +0.27)

side = side - thk*0.5 - ball.radius

dt = 0.5
```

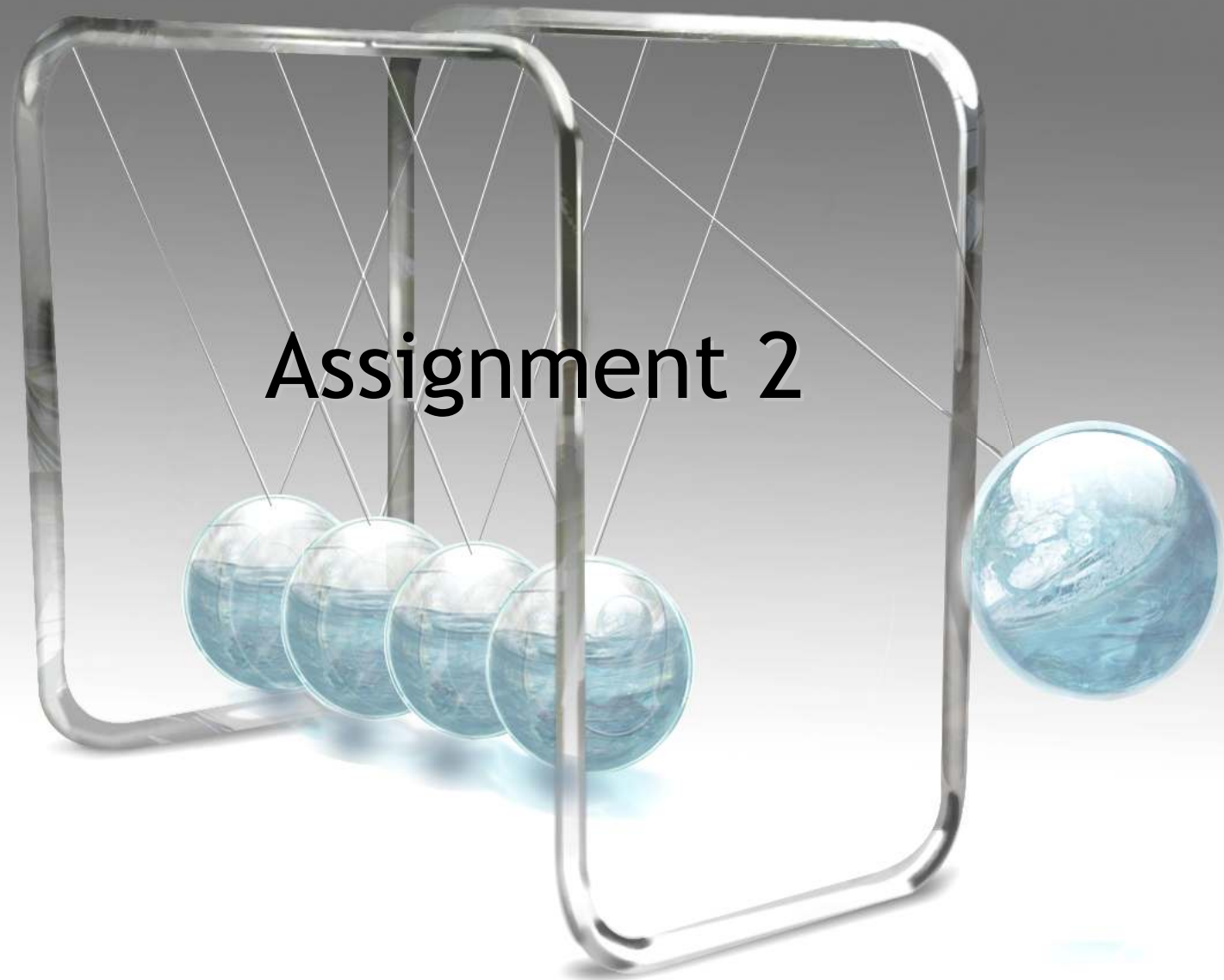
The image shows a VPython window titled "VPython" displaying a 3D scene. The scene consists of a rectangular box with a gray back wall and four other walls: two red walls on the left and right sides, and two blue walls on the top and bottom. A small green sphere (ball) is positioned in the center of the box, slightly above the bottom wall. The box is rendered with perspective, and the walls have a slight thickness. The window has a standard Windows-style title bar and menu bar.

VPython

pythonThompsonVPython1.flv

E-learning homework: watch the first 4 videos from
<http://showmedo.com/videos/series?name=pythonThompsonVPythonSeries>

Assignment 2



Assignment 2

- Rube Goldberg machine
 - rube-goldberg-machine2.wmv
 - honda_commercial_the_cog.mov
 - littlebigplanet_gdc07.wmv

Assignment 2

- Min. interact with 3 objects
- Should be able to modify properties of the ball and objects
- Milestones (all graded):
 - Week 12: design (submit and get approval from instructor)
 - Week 13: scene and objects created, input
 - Week 16: Final submission

References

- python.org
- vpython.org
- python-eggs.org

- <http://www.physics.ucf.edu/~mdj/MinimalPython.html>

- In ref drive:
 - Ralph Noack. “An Incremental Introduction to Python”
 - Michael Williams. “Handbook of the Physics Computing Course”

- <http://showmedo.com/videos/series?name=pythonThompsonVPythonSeries>