

Introdução à Física Computacional I (4300218)

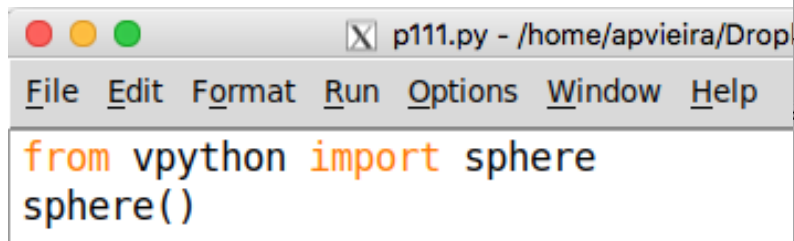
Prof. André Vieira
apvieira@if.usp.br
Sala 3120 – Edifício Principal

Aula 5

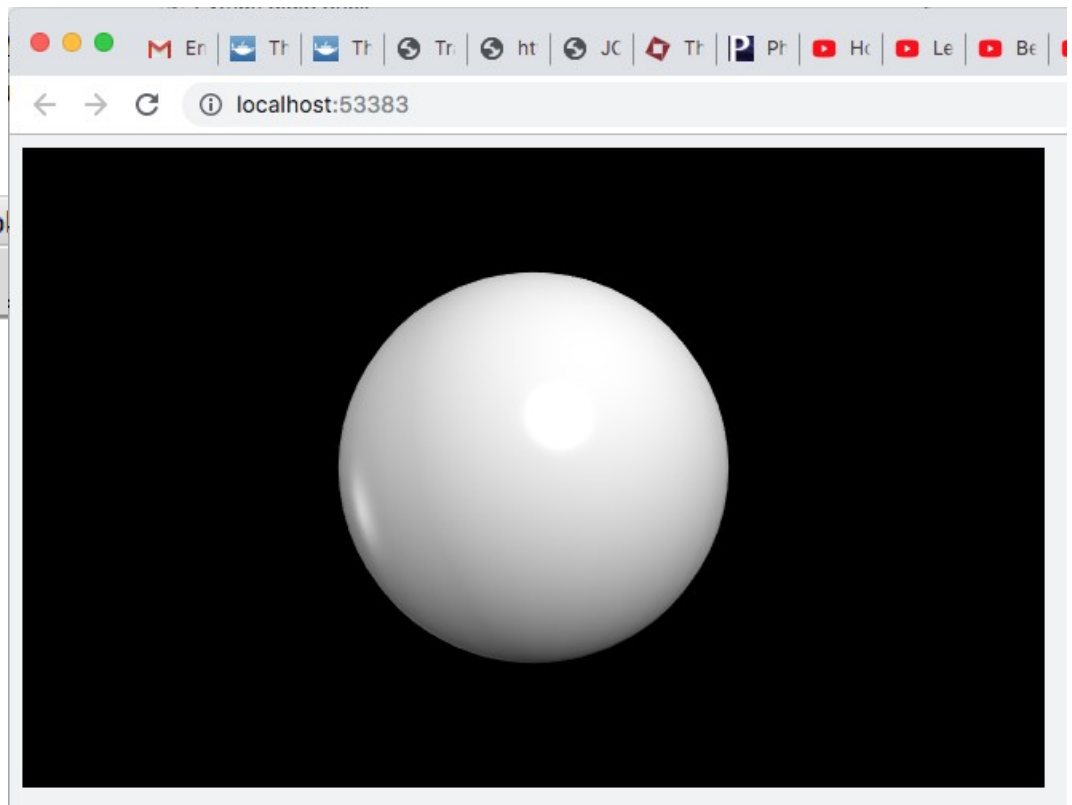
Gráficos e visualização:
gráficos em 3D e exercícios

Gráficos em 3D

- Para criar gráficos em 3D usando Python, invoque o pacote `vpython`.



```
p111.py - /home/apvieira/Drop
File Edit Format Run Options Window Help
from vpython import sphere
sphere()
```



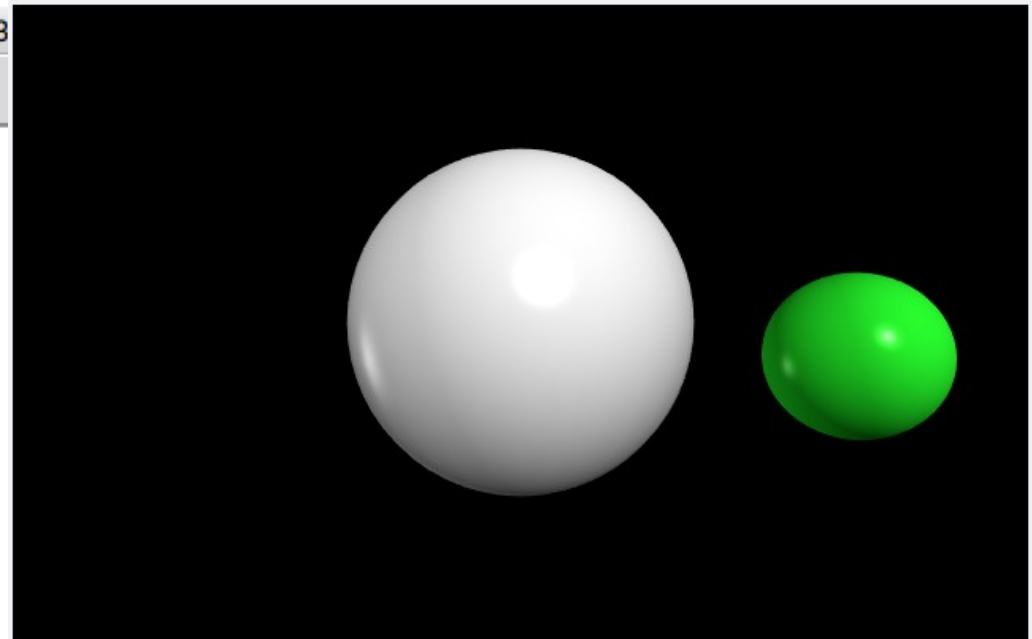
Note que o livro-texto (Newman) refere-se a uma versão antiga do VPython, que se diferencia da versão atual tanto pelo nome do pacote quanto por algumas questões de sintaxe.

Gráficos em 3D

- É possível controlar vários atributos da esfera que criamos, como tamanho, posição e cor.

```
p112.py - /home/apvieira/Dropbox/disciplinas/43
File Edit Format Run Options Window Help
from vpython import sphere,vector,color
sphere()
sphere(radius=0.5,pos=vector(2.0,-0.2,0.0),\
      color=color.green)
```

Por padrão, o VPython supõe o eixo x para a direita, o eixo y para cima e o eixo z saindo da tela.



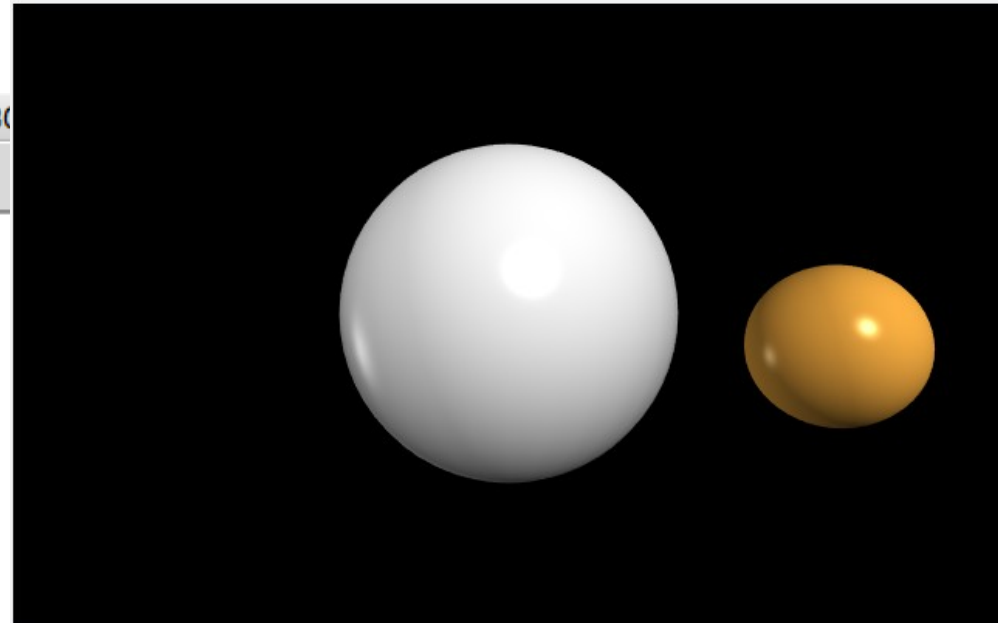
Note que o livro-texto (Newman) refere-se a uma versão antiga do VPython, que se diferencia da versão atual tanto pelo nome do pacote quanto por algumas questões de sintaxe.

Gráficos em 3D

- As cores definidas através do objeto `color` são descritas na documentação do VPython, em <https://bit.ly/2k6mRxU>. Mas você pode criar suas próprias cores, como mostrado abaixo.

```
*cor.py - /home/apvieira/Dropbox/disciplinas/430
File Edit Format Run Options Window Help
from vpython import sphere,vector

acobreado = vector(1.0,0.7,0.2)
sphere()
sphere(radius=0.5,pos=vector(2.0,-0.2,0.0),\
      color=acobreado)
```

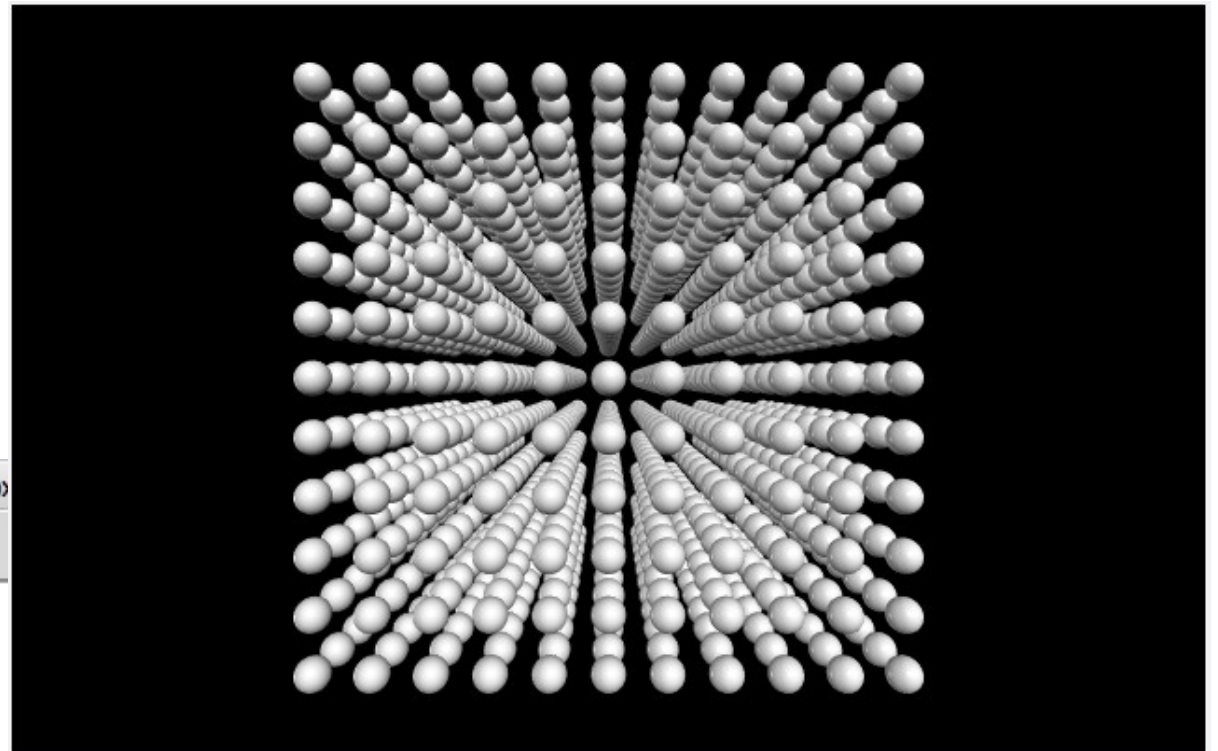


No vetor que define a cor, as componentes indicam a quantidade de vermelho, verde e azul na composição da cor.

Gráficos em 3D

- Exemplo 1: uma rede cristalina.

```
p113.py - /home/apvieira/Dropbox
File Edit Format Run Options Window Help
from vpython import sphere, vector
L=5
R = 0.3
for i in range(-L,L+1):
    for j in range(-L,L+1):
        for k in range(-L,L+1):
            sphere(pos=vector(i,j,k),radius=R)
```

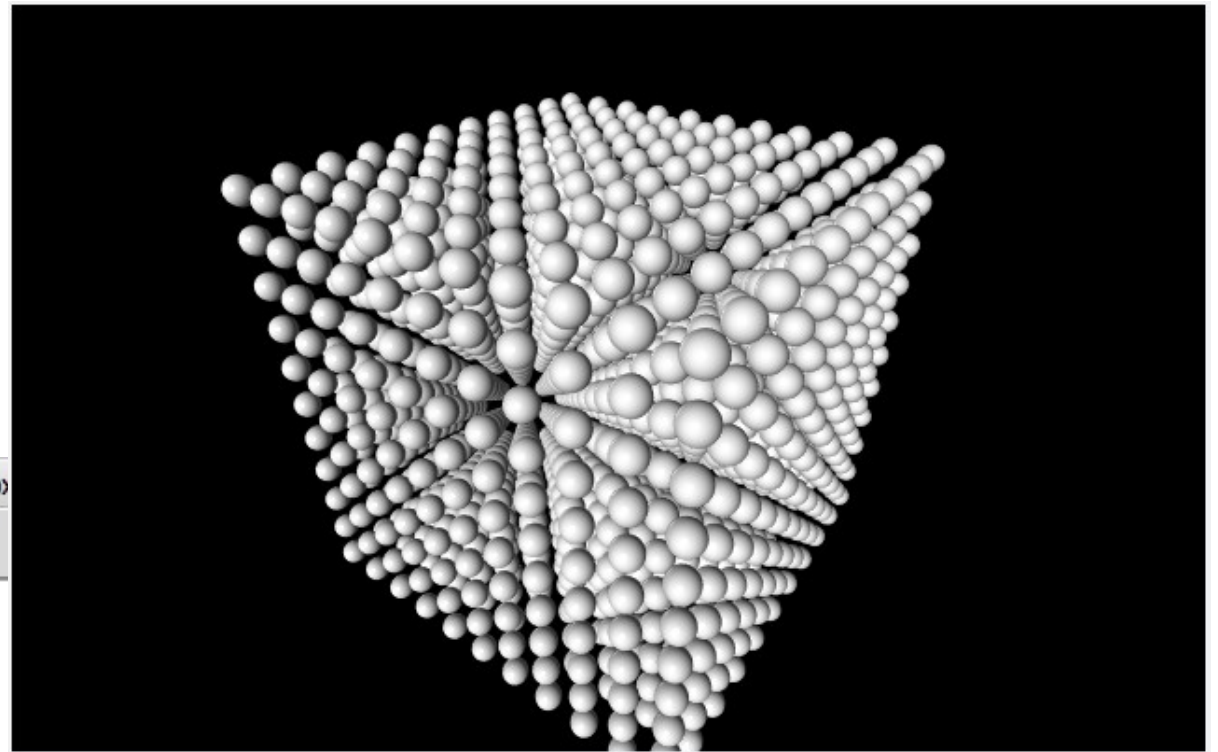


Utilize o mouse ou o touchpad em combinação com os botões para girar a figura e aproximá-la ou afastá-la.

Gráficos em 3D

- Exemplo 1: uma rede cristalina.

```
p113.py - /home/apvieira/Dropbox
File Edit Format Run Options Window Help
from vpython import sphere, vector
L=5
R = 0.3
for i in range(-L,L+1):
    for j in range(-L,L+1):
        for k in range(-L,L+1):
            sphere(pos=vector(i,j,k),radius=R)
```

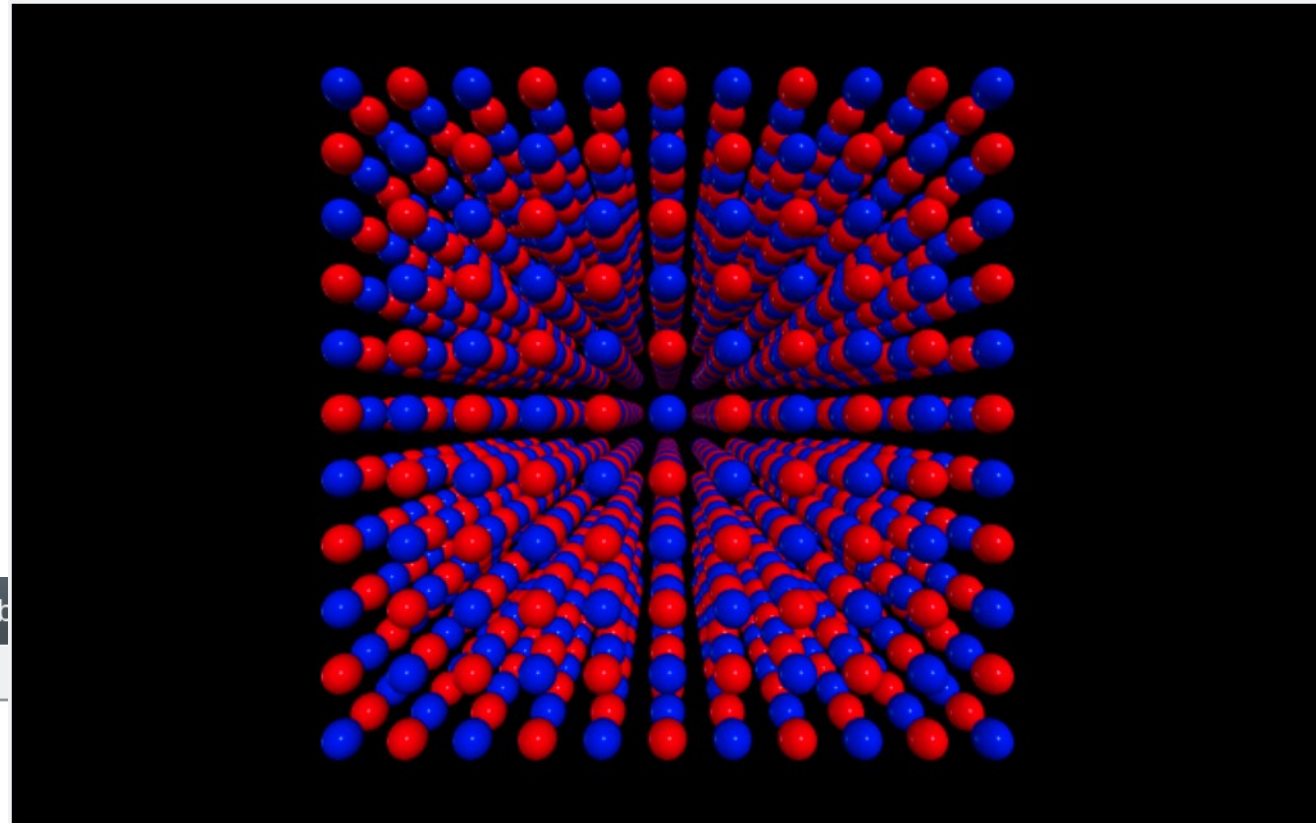


Utilize o mouse ou o touchpad em combinação com os botões para girar a figura e aproximá-la ou afastá-la.

Exercício 1

Em um cristal de cloreto de sódio os átomos de sódio e de cloro arranjam-se alternadamente em uma rede cúbica, de modo que cada átomo de sódio está cercado por 6 átomos de cloro e vice-versa. Crie uma visualização da rede do cloreto de sódio, contendo 11 átomos na aresta do cubo, representando por cores diferentes os dois tipos de átomo.

Exercício 1: solução



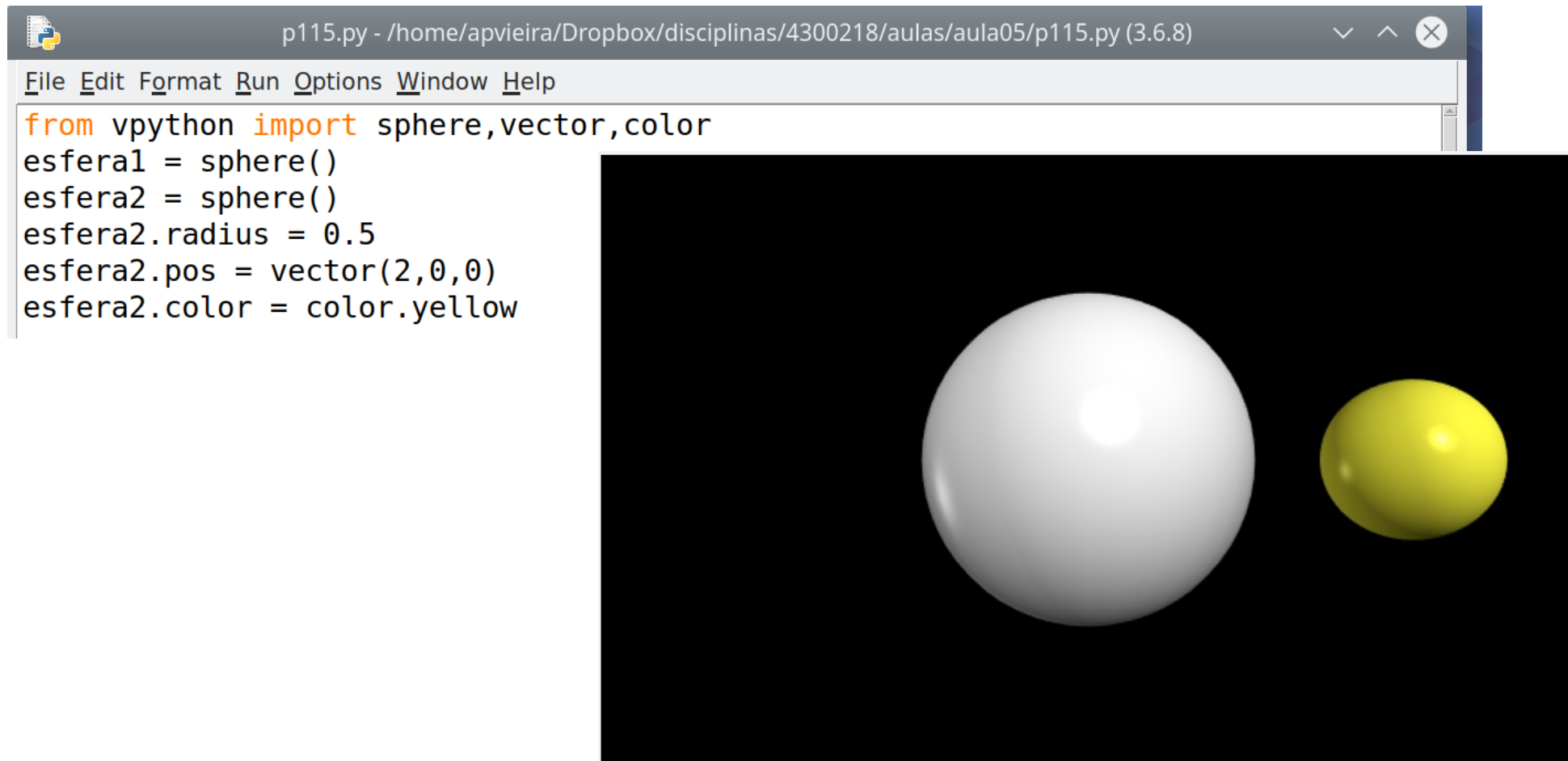
exercicio1.py - /home/apvieira/Dropbox

File Edit Format Run Options Window Help

```
from vpython import sphere, vector
L=5
R = 0.3
for i in range(-L,L+1):
    for j in range(-L,L+1):
        for k in range(-L,L+1):
            ijk=(-1)**(i+j+k)
            vm = (1.+ijk)/2
            az = 1-vm
            sphere(pos=vector(i,j,k),radius=R,\
                    color=vector(vm,0,az))
```

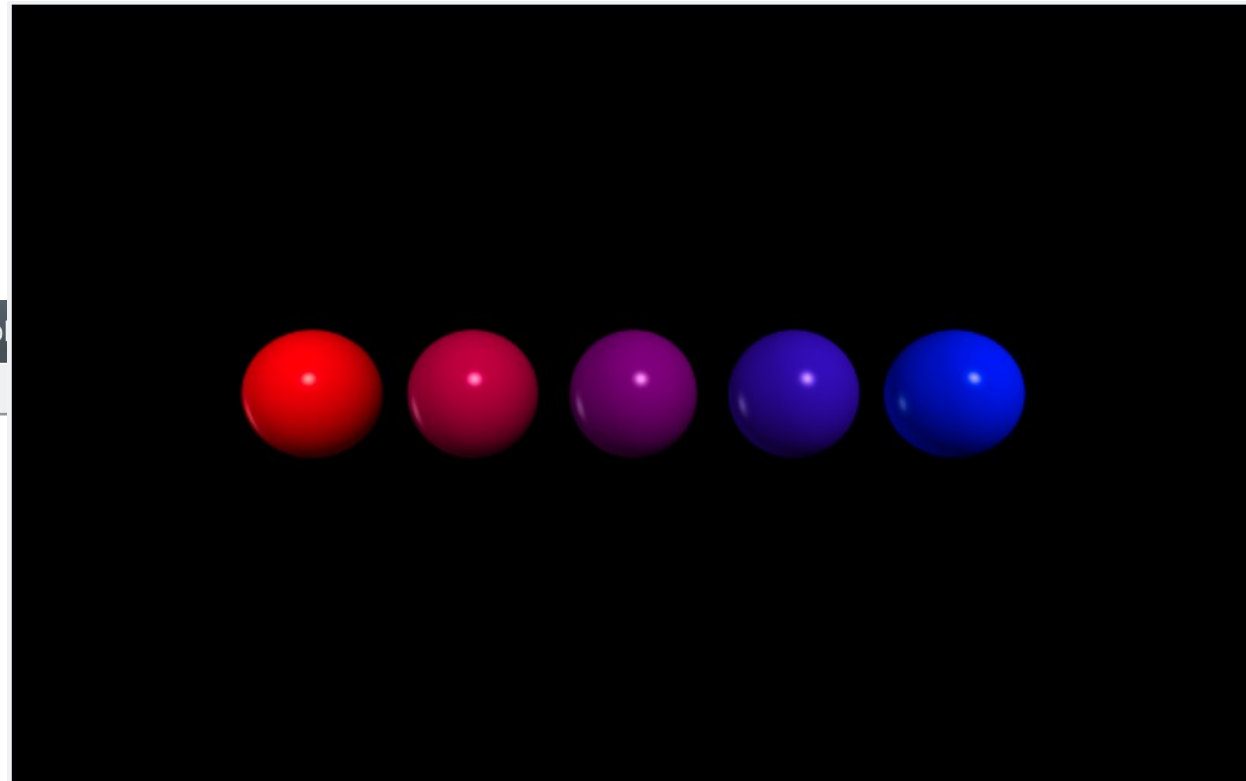

Gráficos em 3D

- É possível atribuir um objeto gráfico a uma variável, de modo a alterar seus atributos ao longo do programa. Veja o exemplo abaixo.



Gráficos em 3D

- Da mesma forma, é possível criar uma lista ou array e populá-la com objetos 3D.



```
p115b.py - /home/apvieira/Dropbox/discip  
File Edit Format Run Options Window Help  
from vpython import sphere, vector  
from numpy import empty  
L = 2  
N = 2*L+1  
esfera = empty(N, sphere)  
R = 0.4  
for n in range(-L, L+1):  
    vm = (1.-n/L)/2  
    az = 1 - vm  
    esfera[n] = sphere(pos=vector(n,0,0),\  
                        radius=R)  
    esfera[n].color = vector(vm,0.,az)
```

Note como a array vazia é criada já com a informação sobre o tipo de objeto que irá conter.

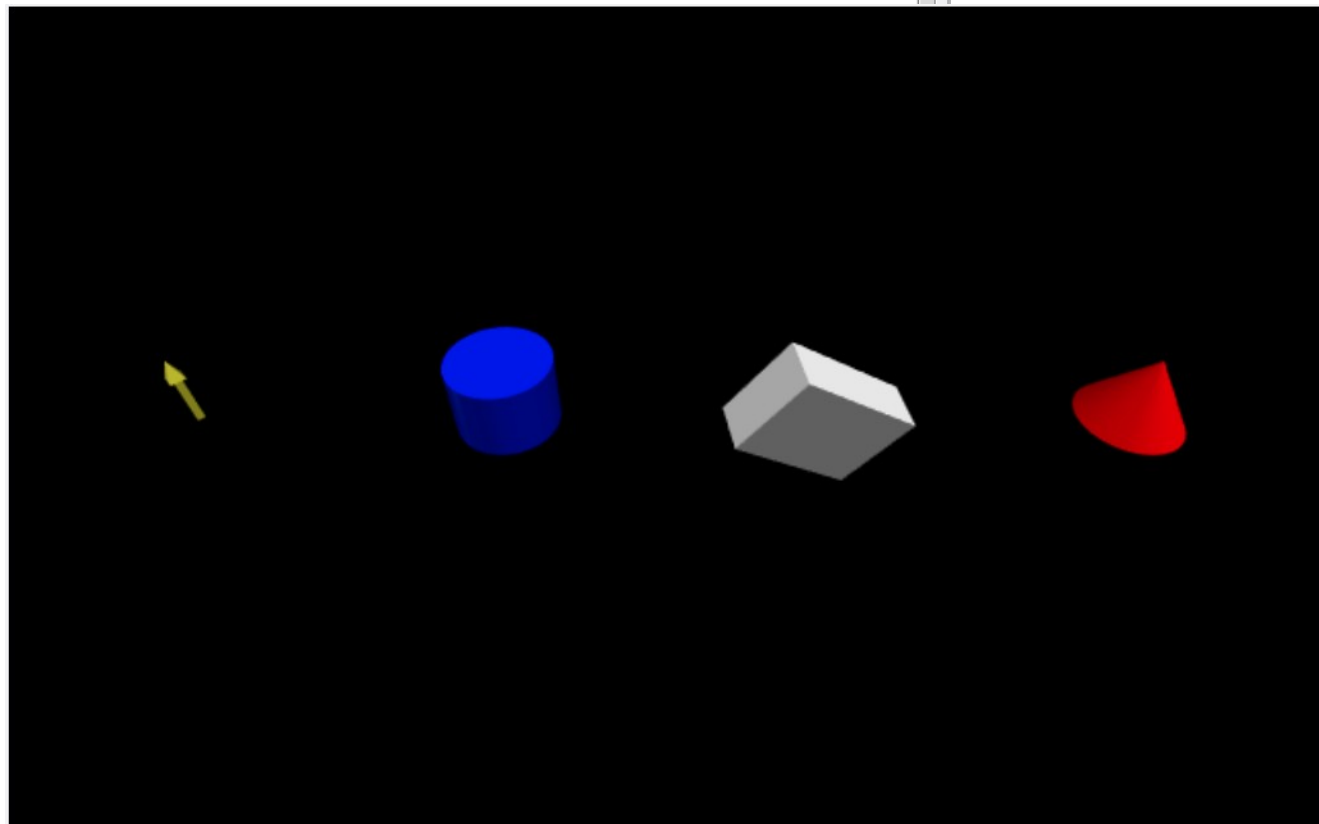
Gráficos em 3D

- Outros tipos de objetos 3D podem ser criados. Vejam em <http://www.vpython.org>

```
p116a.py - /home/apvieira/Dropbox/disciplinas/4300218/aulas/aula05/p116a.py (3.6.8)
File Edit Format Run Options Window Help
from vpython import box, cone, cylinder, pyramid, arrow, vector, color

x,y,z = 30,0,0
a,b,c = 0,10,10
L,H,W = 10,20,30
q,r,s = 1,1,1
R = 10

box(pos=vector(x,y,z), \
    axis=vector(a,b,c), \
    length=L, height=H, \
    width=W, up=vector(q,r,s))
cone(pos=vector(3*x,y,z), \
    axis=vector(a,b,c), \
    radius=R, color=color.red)
cylinder(pos=vector(-x,y,z), \
    axis=vector(a,b,c), \
    radius=R, \
    color=color.blue)
arrow(pos=vector(-3*x,y,z), \
    axis=vector(a,b,c), \
    color=color.yellow)
```



Gráficos em 3D

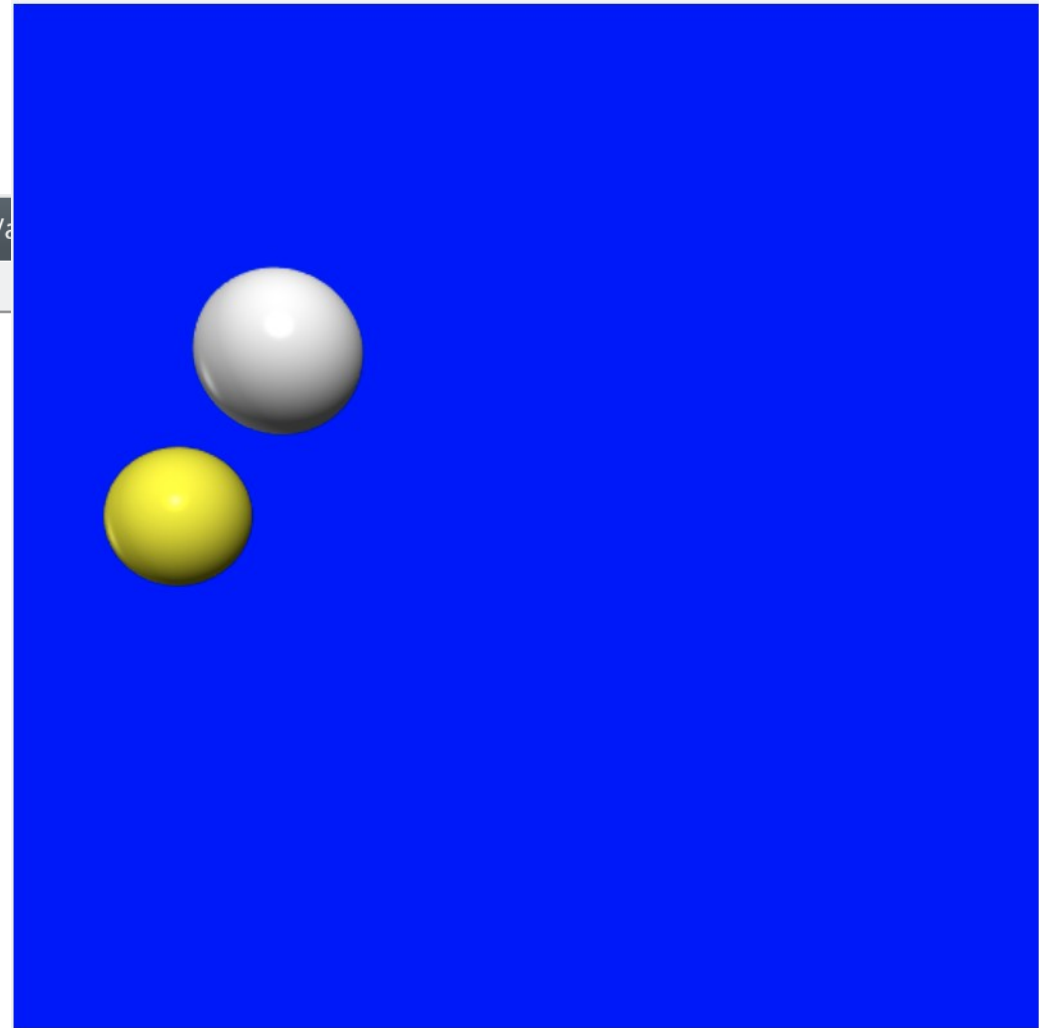
- É possível controlar várias propriedades da visualização.

```
p116b.py - /home/apvieira/Dropbox/disciplinas/4300218/a
File Edit Format Run Options Window Help
from vpython import canvas,color,vector,sphere

canvas(width=500,height=500, \
      center=vector(5,0,0),\
      forward=vector(0,0,-1), \
      background=color.blue)

# A opção `foreground` não está mais disponível

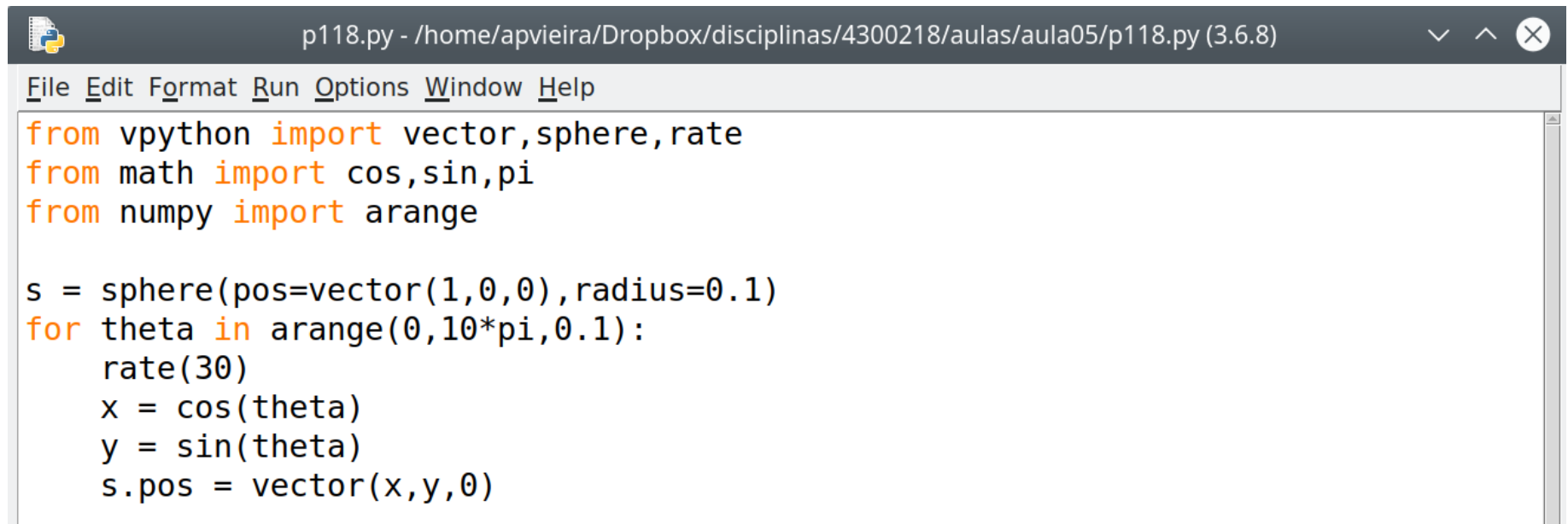
sphere(color=color.yellow)
sphere(color=color.white,pos=vector(2,2,2))
```



Note as diferenças de sintaxe entre a versão antiga e a versão atual do Vpython: `display` agora é `canvas`

Animações

- Alterando repetidamente a posição de um objeto, é possível criar uma animação.

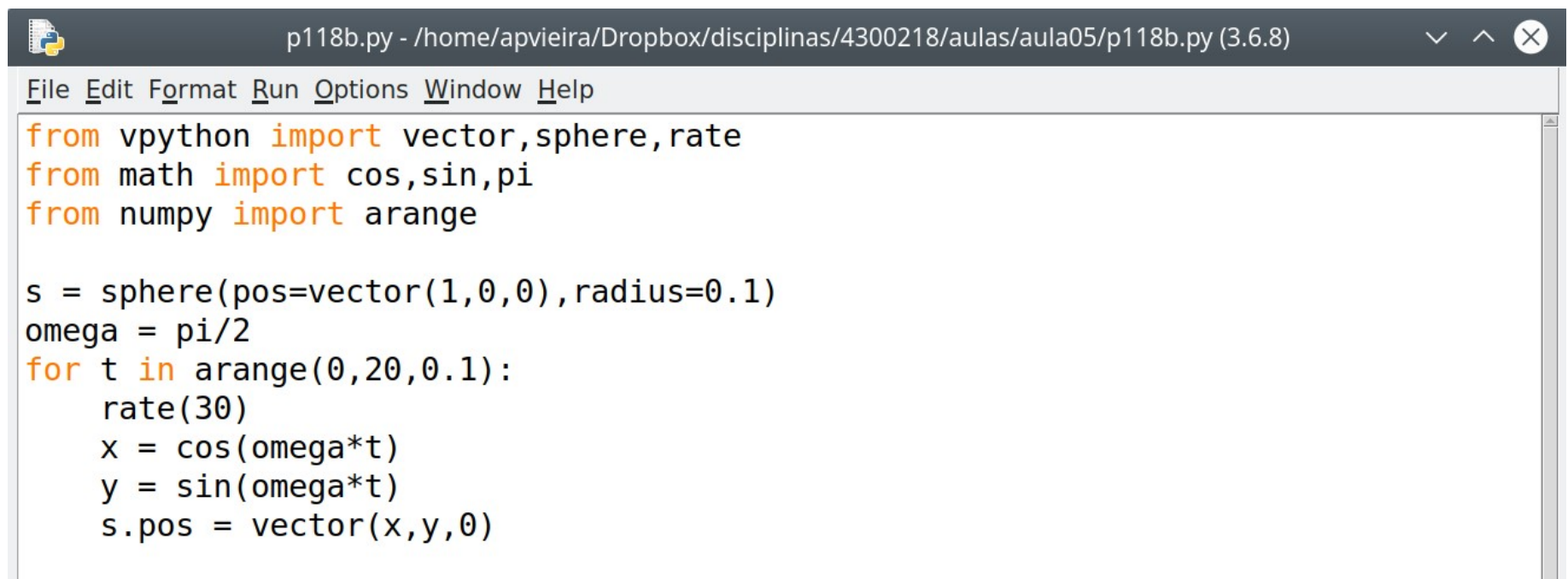


```
p118.py - /home/apvieira/Dropbox/disciplinas/4300218/aulas/aula05/p118.py (3.6.8)
File Edit Format Run Options Window Help
from vpython import vector, sphere, rate
from math import cos, sin, pi
from numpy import arange

s = sphere(pos=vector(1,0,0), radius=0.1)
for theta in arange(0, 10*pi, 0.1):
    rate(30)
    x = cos(theta)
    y = sin(theta)
    s.pos = vector(x, y, 0)
```

Animações

- Normalmente queremos que a animação se desenrole à medida que o “tempo” passa. Isso pode ser feito da seguinte forma:

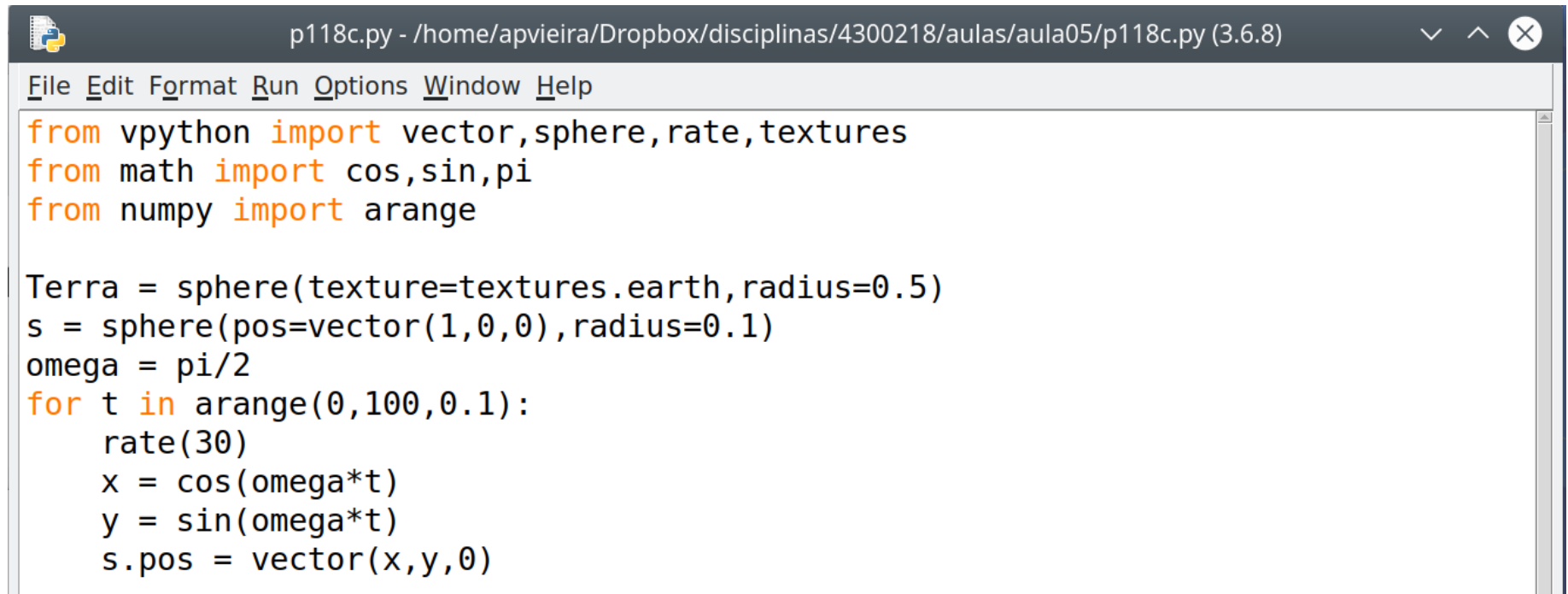


```
p118b.py - /home/apvieira/Dropbox/disciplinas/4300218/aulas/aula05/p118b.py (3.6.8)
File Edit Format Run Options Window Help
from vpython import vector, sphere, rate
from math import cos, sin, pi
from numpy import arange

s = sphere(pos=vector(1,0,0), radius=0.1)
omega = pi/2
for t in arange(0,20,0.1):
    rate(30)
    x = cos(omega*t)
    y = sin(omega*t)
    s.pos = vector(x,y,0)
```

Animações

- Podemos sofisticar a animação:

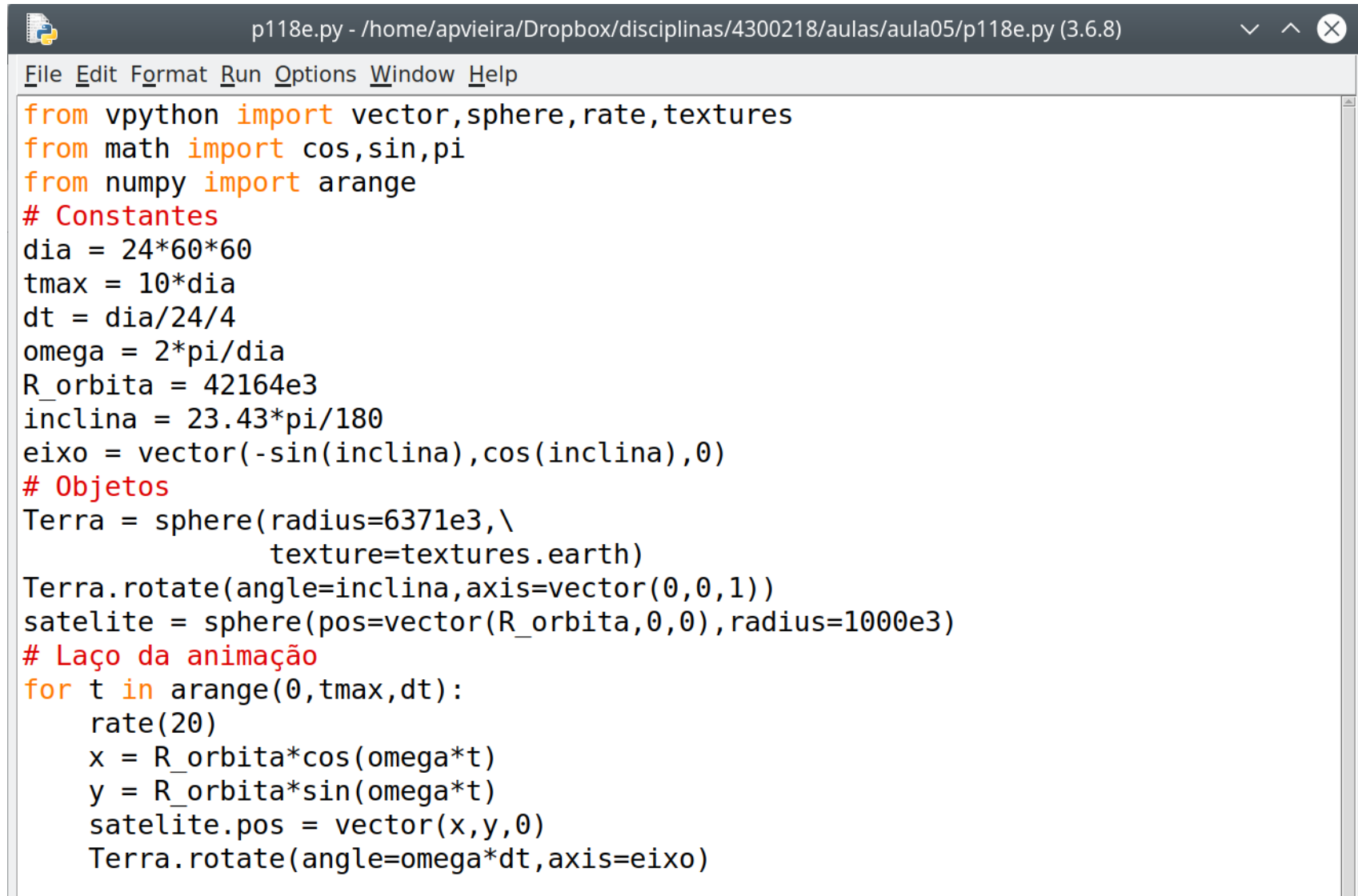


```
p118c.py - /home/apvieira/Dropbox/disciplinas/4300218/aulas/aula05/p118c.py (3.6.8)
File Edit Format Run Options Window Help
from vpython import vector, sphere, rate, textures
from math import cos, sin, pi
from numpy import arange

Terra = sphere(texture=textures.earth, radius=0.5)
s = sphere(pos=vector(1, 0, 0), radius=0.1)
omega = pi/2
for t in arange(0, 100, 0.1):
    rate(30)
    x = cos(omega*t)
    y = sin(omega*t)
    s.pos = vector(x, y, 0)
```


Animações

- Sofisticando ainda mais:



```
p118e.py - /home/apvieira/Dropbox/disciplinas/4300218/aulas/aula05/p118e.py (3.6.8)
File Edit Format Run Options Window Help
from vpython import vector, sphere, rate, textures
from math import cos, sin, pi
from numpy import arange
# Constantes
dia = 24*60*60
tmax = 10*dia
dt = dia/24/4
omega = 2*pi/dia
R_orbita = 42164e3
inclina = 23.43*pi/180
eixo = vector(-sin(inclina), cos(inclina), 0)
# Objetos
Terra = sphere(radius=6371e3, \
                texture=textures.earth)
Terra.rotate(angle=inclina, axis=vector(0, 0, 1))
satelite = sphere(pos=vector(R_orbita, 0, 0), radius=1000e3)
# Laço da animação
for t in arange(0, tmax, dt):
    rate(20)
    x = R_orbita*cos(omega*t)
    y = R_orbita*sin(omega*t)
    satelite.pos = vector(x, y, 0)
    Terra.rotate(angle=omega*dt, axis=eixo)
```

Animações

- E mais um pouco:

```
p118f.py - /home/apvieira/Dropbox/disciplinas/4300218/aulas/aula05/p118f.py (3.6.8)
File Edit Format Run Options Window Help
from vpython import vector,sphere,rate,textures,scene
from math import cos,sin,pi
from numpy import arange
# Ajustando a visualização
scene.forward = vector(0,-0.25,-1)
# Constantes
dia = 24*60*60
tmax = 10*dia
dt = dia/24/4
R_orbita = 42164e3
omega = 2*pi/dia
phi = 210*pi/180
# Objetos
Terra = sphere(radius=6371e3,texture=textures.earth)
satelite = sphere(pos=vector(R_orbita*cos(phi),0,-R_orbita*sin(phi)),\
                        radius=1000e3)
# Laço da animação
for t in arange(0,tmax,dt):
    rate(20)
    x = R_orbita*cos(omega*t+phi)
    z = -R_orbita*sin(omega*t+phi)
    satelite.pos = vector(x,0,z)
    Terra.rotate(angle=omega*dt,axis=vector(0,1,0))
```

Exercício 2

Para pequenas oscilações, um pêndulo simples realiza um movimento harmônico simples, em que o ângulo do fio do pêndulo com a vertical depende do tempo segundo a equação

$$\theta(t) = A \cos(\omega t + \phi),$$

em que as constantes A e ϕ dependem das condições iniciais do movimento. Utilizando uma esfera vermelha de raio 1 para representar a massa do pêndulo e um cilindro estreito branco de comprimento 10 para representar o fio, anime o movimento do pêndulo durante 10 períodos, partindo do repouso, formando um ângulo de 10 graus com a vertical. Você vai precisar utilizar o atributo `axis` do objeto `cylinder`.

Exercício 2: solução

```
exercicio2.py - /home/apvieira/Dropbox/disciplinas/4300218/aulas/aula05/exercicio2.py (3.6.8)
File Edit Format Run Options Window Help
from vpython import vector,sphere,rate,cylinder,color
from math import cos,sin,pi,sqrt
from numpy import arange
# Constantes
R_esfera = 1.
R_fio = 0.1
L_fio = 10
g = 9.8
T = 2*pi*sqrt(L_fio/g)
omega = 2*pi/T
tmax = 10*T
dt = T/100
# Objetos
A = 10*pi/180
theta = A
x,y = L_fio*sin(theta),-L_fio*cos(theta)
esfera = sphere(radius=R_esfera,color=color.red,pos=vector(x,y,0))
fio = cylinder(radius=R_fio,axis=esfera.pos)
# Laço da animação
for t in arange(0,tmax,dt):
    rate(30)
    theta = A*cos(omega*t)
    x,y = L_fio*sin(theta),-L_fio*cos(theta)
    esfera.pos = vector(x,y,0)
    fio.axis = esfera.pos
```

Para a próxima aula

- Exercícios no moodle.