102_Basic Motion cm

是的，我們回到了一維等速運動，一個你們不用複習一樣會記得的物理觀念。我們今天要從最簡單，最本質的這個運動開始。這堂課沒意外的話，我們可以做出以下的畫面，而影片如這裡所示。

Code

```
from vpython import *
'''
1. 基礎參數設定
'''
size = 0.1
L = 1
v = 0.03
t = 0
dt = 0.01
'''
2. 畫面
'''
scene = canvas(title = "1D same velocity motion", width = 800, height = 600, x = 0, y = 0, center = vec(0,0,0), background = vec(0.2, 0.4, 0.8))

floor = box(pos=vec(0,0,0), size = vec(L, 0.1*size, L))

cube = box(pos = vec(-0.5*L+0.5*size, 0.55*size, 0), size = vec(size, size, size), color = color.blue)

gd = graph(title = 'x-t plot', width = 600, height = 450, x = 0, y = 600, xtitle= "t(s)", ytitle='v(m/s)')

gd2 = graph(title="v-t plot", width=600, height=450, x=0, y=1050, xtitle="t(s)", ytitle="v(m/s)")

xt = gcurve(graph=gd, color=color.red)
vt = gcurve(graph=gd2, color=color.red)
'''
 3. 物體運動部分, 木塊到達地板邊緣時停止執行
'''
while cube.pos.x <= 0.5*L - 0.5*size:
    rate(1000)
    cube.pos.x += v*dt
    xt.plot(pos=(t, cube.pos.x))
    vt.plot(pos=(t, v))
    t += dt

print("t = ", t)
```

Commentary

接下來我們會一行一行慢慢檢視

```
from vpython import *
```

#第一行的目的是「引入」vpython 這個模組。因為Python內建並沒有這個模組，所以我們必須自行下載並把他「引入」到以下程式中。就是一個標準用法，把他背起來就好

```
'''
1. 基礎參數設定
'''
```

#裡面為吾人自行設定的參數，我習慣一開始就把參數寫在上面，給他一個專屬區域。
#其中size是箱子的大小，L是地板長度，其他你們應看的懂。而how to use在以下說明

```
scene = canvas(title = "1D same velocity motion", width = 800,
height = 600, x = 0, y = 0, center = vec(0,0,0), background =
vec(0.2, 0.4, 0.8))
```

這裡的意思是，我建立了一個canvas，並且把它命名為scene。而canvas則是vpython程式庫裡面定義的一個「物件」，代表「布幕」。而裡面的參數在101_Object Function有細提

```
floor = box(pos=vec(0,0,0), size = vec(L, 0.1*size, L))

cube = box(pos = vec(-0.5*L+0.5*size, 0.55*size, 0), size =
vec(size, size, size), color = color.blue)
```

floor 和上述的scene有點類似，都是屬於吾人給予的名稱。而box則是另一個物件，代表「立體矩形」，一樣在「101_Object Function」都有詳提。
#而程式的意義為，我用box建立一個「地板」，再用box建立了一個「運動的物體」

```
gd = graph(title = 'x-t plot', width = 600, height = 450, x = 0, y
= 600, xtitle= "t(s)", ytitle='v(m/s)')

gd2 = graph(title="v-t plot", width=600, height=450, x=0, y=1050,
xtitle="t(s)", ytitle="v(m/s)")
```

這裡的graph 指的是「圖表」，就是上面那邊的x-t圖, v-t圖，在「101_Useful Function」中有詳提
#和canvas有點像，不一樣的是多了xtitle和ytitle，就是x, y軸

```
xt = gcurve(graph=gd, color=color.red)
vt = gcurve(graph=gd2, color=color.red)
```

這裡的gcurve 指的是「會出現在圖表裡面的線」，在「101_Useful Function」中有詳提，後面接的graph就是他會出現在的圖

```
while cube.pos.x <= 0.5*L - 0.5*size:
    rate(1000)
    cube.pos.x += v*dt
    xt.plot(pos=(t, cube.pos.x))
    vt.plot(pos=(t, v))
    t += dt
```

這裡的while迴圈條件是「當cube邊緣碰到floor的邊緣，停止迴圈」
這裡的rate 指的是「一秒會跑幾次迴圈」，1000算是相當夠了（一秒跑1000*dt次）。但太快的話就很就跑完了。如果你想要現實生活中的一秒=模擬器裡的一秒，可以令rate = 1/dt
#這裡的plot指的是gcurve這個物體執行「畫線」的這個動作，也就是在graph中特定的x,y圖留下軌跡

'''
這裡補一個小細節，我在剛學的時候超級無敵搞不懂的地方：請問scene, cube, floor這些名字我可以亂改嗎？
答案是可以的，因為這些名字都是那個物體的「名稱」。也可以把他們想成儲存「圖形資料」的變數
而物體今天真正出現在畫面上的原因是因為：你今天只要建立了一個儲存圖形的變數，那麼圖形就會同時出現在畫面之中，無需額外的函數等等
'''

Practice

請建立一個「沿x軸等加速度運動」的動畫，並且畫出其x-t plot, v-t plot
參數設定如下：

```
l = 1
size = 0.1
v = 0
a = 0.1
t = 0
dt = 0.01
```