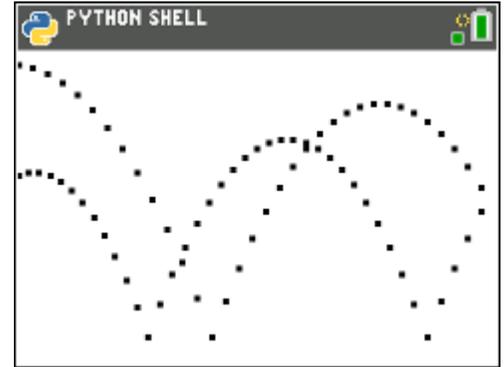




After completing the 'Getting Started...' activity let's make an object move on the screen. Animation is big business and computers have changed the landscape.

- 0. This project produces a 'bouncing ball' animation on the graphics 'canvas'.



- 1. Begin with this short program that causes a point to move across the screen.

The point starts at the middle of the left edge of the screen: (0, 106).

`plot_xy(, ,)` is found in the `ti_draw...` module and simply plots the point (x, y).

In `plot_xy(x, y, 1)` the 1 represents the point *style* which can be a value from 1 to 13. Try the other styles.

The `if` statement at the end causes the point to 'wrap around' on the screen. Your first modification will cause the point to move back and forth between the sides of the screen.

```

EDITOR: DRAW1
PROGRAM LINE 0003
from ti_system import *
from ti_draw import *

x=0
y=106
while not escape():
  clear()
  plot_xy(x,y,1)
  sleep(.1)
  x+=10
  if x>318:x=0
  
```

- 2. Introduce another variable, `dx = 10`, and use it to make the point move to the right in place of the constant 10 used in the previous step.

```

EDITOR: DRAW1
PROGRAM LINE 0006

x=0
y=106
dx=10
while not escape():
  clear()
  plot_xy(x,y,1)
  sleep(.1)
  x+=dx
  if x>318:x=0
  
```



10 MOC: Python Modules

TI-84 PLUS CE PYTHON

- Change the **if** statement to reverse direction by changing the value of **dx**.

if x > 318 : dx = -dx

Add another bit of code to cause the point to change direction at the left side of the screen, too. Try it yourself first.

```
EDITOR: DRAW1
PROGRAM LINE 0013

x=0
y=106
dx=10
while not escape():
  clear()
  plot_xy(x,y,1)
  sleep(.1)
  x+=dx
  if x>318:dx=-dx
  ...
Fns... a A # Tools Run Files
```

- Did you add an **or...** clause to the condition?

if x > 318 or x<0: dx = -dx

Test the program now to see that the point moves right and left on the screen until you press **[clear]**.

```
EDITOR: DRAW1
PROGRAM LINE 0013

x=0
y=106
dx=10
while not escape():
  clear()
  plot_xy(x,y,1)
  sleep(.1)
  x+=dx
  if x>318 or x<0:dx=-dx
  ...
Fns... a A # Tools Run Files
```

- Modify the program to introduce *vertical* motion as well, but this time instead of moving at a fixed rate, make the point *accelerate* like a falling object: seemingly under the effect of *gravity*!

Introduce two new variables: **dy = 0** for the change in the y-position and **g = 2** to represent 'gravity'.

The value of g is arbitrary and can be edited later to see the effect.

```
EDITOR: DRAW1
PROGRAM LINE 0010

from ti_system import *
from ti_draw import *

x=0
y=106
dx=10

dy=0
g=2

while not escape():
  ...
Fns... a A # Tools Run Files
```

- When **x** changes value by adding **dx**, so do **y** and **dy**:
 - dy** increases by the amount **g** and
 - y** increases by the amount **dy**.

Note: (x, y) is position. dx and dy represent velocity (changes in position). g represents acceleration (change in velocity).

```
EDITOR: DRAW1
PROGRAM LINE 0017

while not escape():
  clear()
  plot_xy(x,y,12)
  sleep(.1)
  x+=dx
  if x>318 or x<0:dx=-dx

  dy+=g
  y+=dy
  ...
Fns... a A # Tools Run Files
```



10 MOC: Python Modules

TI-84 PLUS CE PYTHON

- 7. When the point reaches the bottom of the screen (200 is close to the bottom), make the point 'bounce' upward using $dy = -dy + g$ (change direction). The extra $+g$ here is to decrease the energy in the bouncing ball. Without it the ball will return to its original height and that is unrealistic.

We also make the point slow down (lose energy) horizontally:

$$dx *= 0.9.$$

Eventually the point will stop moving.

- 8. Change the starting position of the point from (0, 106) to (0,10) to give the point more vertical room to fall.

<Run> the program and experiment with the numbers used in the code to see the effect of each variable.

Change the plot_xy(x, y, style) and color (use `set_color(, ,)`) of the point.

- 9. Move the `clear()` function *before* the `while not escape()` statement so that each position of the point remains on the screen, thus showing the complete *path* of the point. This is point style 3. What patterns do you see in this path?

Challenge: connect the dots!

```

EDITOR: DRAW1
PROGRAM LINE 0025
x+=dx
if x>318 or x<0:dx=-dx

dy+=g
y+=dy

if y>200:
y=200
dy=-dy+g
dx*=.9

```

```

EDITOR: DRAW1
PROGRAM LINE 0010
from ti_system import *
from ti_draw import *

x=0
y=10
dx=10

dy=0
g=2

while not escape():

```

