Unit 5: Graphics | **Application: Making a Point Bounce**

In this application for Unit 5 we'll build graphics-based programs.

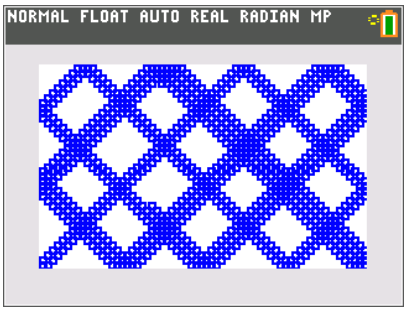
Objectives:

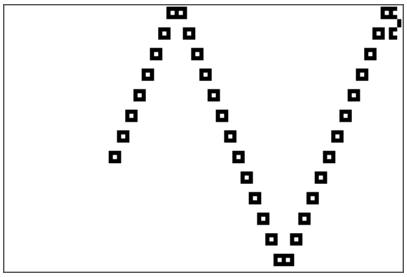
- Learn how to detect which device the program is running on.
- Making a point bounce around the screen.

Teacher Tip: This is a rather complex project that has some interesting physics hidden inside. The 'particle' moves by adding *delta-x* and *delta-y* values to its coordinates during each iteration of the loop. These are the horizontal and vertical components of velocity. When the particle hits an edge of the screen, the appropriate component is 'reversed' (negated) so that the particle appears to 'bounce' off the edge of the screen.

Program "Pong"

In the original video game 'Pong' a pixel bounced around on the screen. The players controlled 'paddles' as in ping-pong to keep the ball in play. This program will produce the 'bouncing ball'. A point will move in an oblique line across the screen and when it hits a boundary it will change direction to look like it's bouncing off the edge.





The same program running on two different calculators.

The first problem to consider is the screen size, since the TI-84 Plus has a different screen size than the TI-84 C/CE. The following code segment will detect the screen size:

```
0→Xmin
1→ΔX          also on the [VARS] Window… menu
If Xmax>95 then
   <it's a C or CE>
Else
   <it's an 84 Plus>
End
```

At this point we now know the device and we set up variables in the **Then** and the **Else** blocks to use in the rest of the program: **M** for the width and **N** for the height.

For the C or CE: **264→M** and **165→N**

For the 84 Plus : **94→M** and **63→N**

Initialize Variables

We also set up the *y*-range to be nice values, too:

```
0→Ymin
1→ ΔY
```

We'll start the point **(A,B)** at a random location, but not too close to the edge of the screen:

```
randInt(10,M-10)→A
randInt(10,N-10)→B
```

We also set up two variables to represent the movement. These will be added to the point's coordinates to move the

point to a new position:

```
randInt(2,5)→D      change in A
randInt(2,5)→E      change in B
```

Teacher Tip: These are the delta-x and delta-y values.

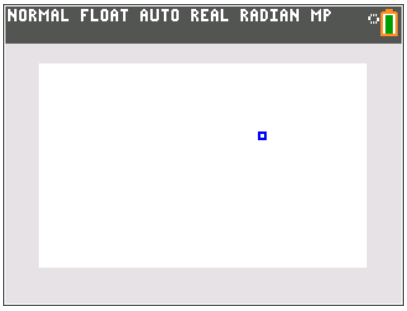
Getting Started

We're ready to start the action. We'll use an infinite loop…

```
While 1
 Pt-On(A,B,2)          style 2 is a large square dot

   <The rest of the program>

End
```



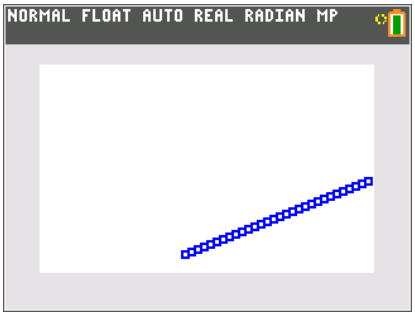
Tip: It's a good idea to put the **End** *in at the same time to keep track of them.*

Making it Move

In the loop and after **Pt-On(** add the 'change' variables to the point's coordinates:

```
A+D→A
B+E→B
```

This changes the point's coordinates.
If you run the program now you will see the point go off in some direction and quickly go right off the screen as in the image to the right.



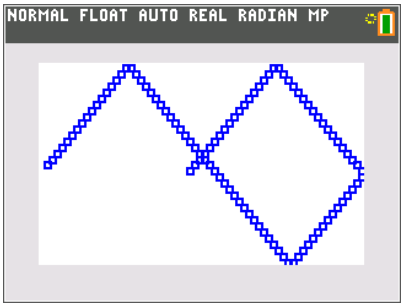
Bouncing

To get the point to detect the edges of the screen we add **If** statements…

```
If A>M or A<0
Then

<this happens when the point is off the right or left side of the screen>

End
```



Do you get this?

Two things need to happen inside the **Then**:

+ Move the point back on the screen `A-E→A`
+ Change the direction to the opposite direction `-E→E`

Write a similar If statement to handle the y-coordinate **B** and it's 'change' variable **E.**

Extensions

Don't leave a trail…

The **Pt-Off** statement will hide a point. Add a **Pt-Off** statement to your program so that the trail of points is not displayed, only the moving point. Caution: don't turn off the same point you turned on. Turn off the *previous* point (you'll need two more variables). The new code needs to go in the right place in the program.

Lose the infinite loop…

getKey (PRGM I/O menu) will get a value representing a key *without* pausing the program like **Prompt** and **Input.** ENTER has the value 105 (row 10, column 5 on the keypad).

Here's a skeleton of what needs to be done:

```
0→K
While Kø105

   <Your program here>

getKey→K
End
```

Your task is to decide where in the program these statements belong so that when you press ENTER the program ends.

How About This…

When you press CLEAR (what **getKey** value is that?) the program starts over with a clear screen and new random values and when you press ENTER the program ends.

Sample Answer:

```
prgmPONG
FnOff
PlotsOff
AxesOff
ClrDraw
0→Xmin
1→ΔX
If Xmax>95
Then
  264→M
  165→N
Else
  94→M
  63→N
End
0→Ymin
1→ΔY

randInt(10,M-10)→A
randInt(10,N-10)→B
randInt(2,5)→D
randInt(2,5)→E

While 1
  Pt-On(A,B,2)
  A+D→A
  B+E→B
  If A>M or A<0
  Then
    A-D→A
    -D→D
  End
  If B>N or B<0
  Then
    B-E→B
    -E→E
  End
End
```