Lektion 5: Grafiken	Anwendung: Der reflektierte Punkt
Die Anwendung für Lektion 5 ist ein etwas umfangreicheres Grafikprogramm.	**Lernziele:** • Den Unterschied zwischen einem TI-84 Plus und einem TI-84 CE-T im Programm ermitteln. • Einen Punkt an den Displayrändern reflektieren lassen.

Hinweis: Das Projekt hat einen physikalischen Hintergrund, denn es handelt sich eigentlich um das Modell eines sich bewegenden Teilchens, das in einen „Kasten“ eingesperrt ist. Das Teilchen bewegt sich mit konstanter Geschwindigkeit. Das erreicht man, indem man bei jedem Schleifendurchlauf dieselben Werte Δx und Δy zu den Punktkoordinaten addiert. Stößt das „Teilchen“ an den Rand, so kehrt sich das Vorzeichen des entsprechenden Δ um. Das sieht dann so aus, als würde das „Teilchen“ von der „Wand“ reflektiert.

Das Programm “Pong”

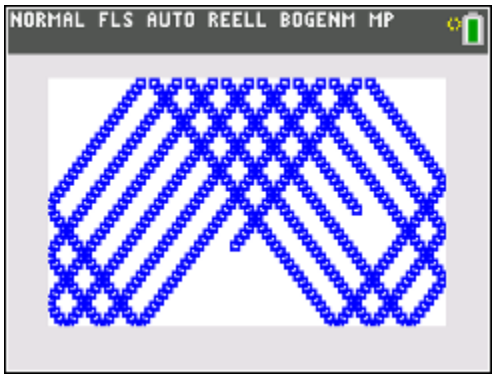
„Pong“ war eines der ersten Videospiele. Ein Ping-Pong-„Ball“ bewegt sich auf dem Display und zwei Spieler müssen mit „Schlägern“ versuchen, den Ball wie beim richtigen Tischtennis im Spiel zu halten. Im Pausenmodus gibt es keine Schläger und der Ball bewegt sich und wird von den Rändern des Displays reflektiert. Diese Bewegung soll das Programm nachbilden. Dabei wird der „Ball“ eine Spur hinterlassen.

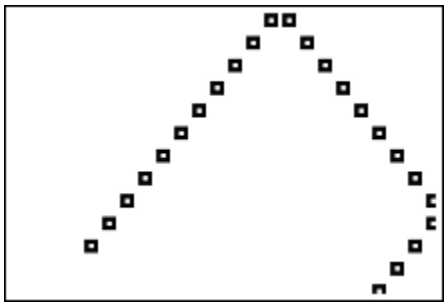
Zunächst soll das Programm ermitteln, auf welchem Taschenrechner es läuft, da die Auflösung beim TI-84 Plus ja kleiner ist als beim TI-84 Plus C/CE. Mit den abgebildeten Befehlen wird der Rechnertyp ermittelt:

```
0→Xmin
1→ΔX
If Xmax>95 then
   <es ist ein TI-84 Plus CE-T>
Else
   <es ist ein TI-84 Plus>
End
```

Die ersten beiden Zeilen sorgen automatisch dafür, dass **Xmax** die maximale Spaltenzahl zugewiesen wird. Dann ergibt sich für die beiden Blöcke:

Then… **264→M** und **165→N** (CE-T)
Else… **94→M** und **63→N** (Plus)





Dasselbe Programm – oben ein TI-84 PLUS CE-T, darunter ein TI-84 Plus

Anfangswerte der Variablen

Auch die Fenstervariablen für y werden eingestellt:

```
Ø→Ymin
1→ ΔY
```

Der Punktabstand beträgt jetzt sowohl in x als auch in y-Richtung 1.

Die Koordinaten des Startpunktes **(A/B)** werden zufällig ermittelt, aber nicht zu dicht an den Displaygrenzen:

```
zufInt(1Ø,M-1Ø)→A
zufInt(1Ø,N-1Ø)→B
```

Auch die Ortsveränderungen in x- und y-Richtung werden zufällig ermittelt:

`zufInt(2,5)→D` x-Richtung
`zufInt(2,5)→E` y-Richtung

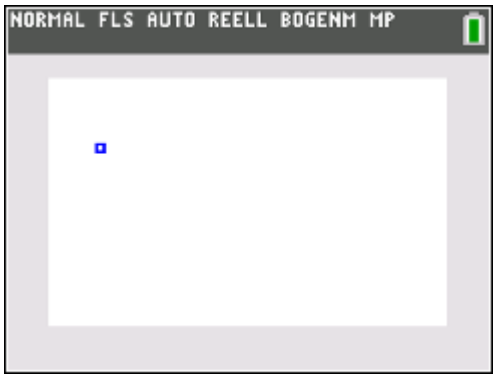
Hinweis: Das sind die Werte für Δx und Δy.

Die Schleife

Nun kann der Punkt bewegt werden. Dazu wird eine unendliche Schleife Verwendet. Die Zahl 2 steht für einen großen quadratischen Punkt:

```
While 1
  Punkt-Ein(A,B,2)

    <restliches Programm>

End
```

Hinweis: Man sollte das **End** *zusammen mit dem* **While** *in das Programm einbauen, um es nicht zu vergessen.*

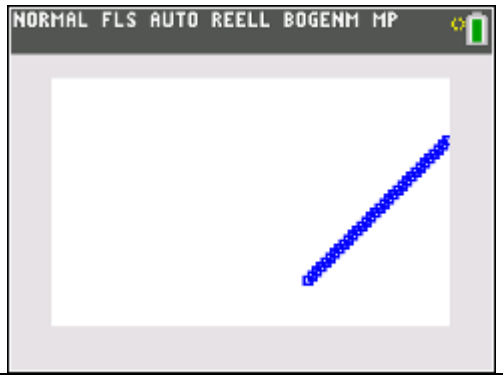


Die Bewegung

Nachdem der erste Punkt gezeichnet wurde, werden die Koordinaten für den nächsten Punkt erzeugt::

```
A+D→A
B+E→B
```

Lässt man das Programm in diesem Stadium laufen, so erkennt man, dass sich der Punkt nach rechts oben bewegt. Allerdings verschwindet er recht schnell aus dem Display.



Reflexion

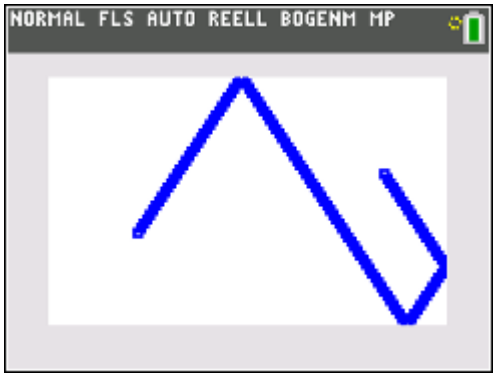
Dazu muss festgestellt werden, ob der Punkt die Displaygrenzen erreicht hat:

```
If A>M or A<Ø
Then

<Befehle für den Fall, dass der linke oder rechte Rand erreicht wurde>

End
```

Zwei Dinge müssen innerhalb des **Then…** passieren:

+ der Punkt muss zurück aufs Display `A-D→A`
+ die Richtung muss umgekehrt werden `-D→D`

Für die y-Koordinate muss eine gleiche Befehlsgruppe erstellt werden.



Ergänzungen

1. Die „Spur“ abschalten

Der Befehl **Punkt-Aus** verbirgt einen Punkt. Fügt man diesen Befehl an der richtigen Stelle im Programm hinzu, so wird die „Spur“ des Punktes gelöscht und man sieht nur den sich bewegenden Punkt. Aber Achtung: man darf nicht denselben Punkt löschen, den man gerade gezeichnet hat, sondern man muss den davor löschen. Dazu braucht man zwei weitere Variablen.

2. Die unendliche Schleife ersetzen

getKey (PRGM E/A Menü) erzeugt einen Wert, der den Tastencode darstellt. Dabei wird das Programm nicht wie bei **Prompt** und **Input** unterbrochen. ENTER hat den Wert 105 (Reihe 10 Spalte 5 auf der Tastatur).
So könnten die zu verwendenden Befehle aussehen:

```
Ø→K
While K≠1Ø5

   <Befehlsblock>

getKey→K
End
```

Das Programm soll nun so verändert werden, dass es beim Drücken von ENTER endet.

3. Einen Neustart aus dem Programm heraus erzeugen

Drückt man CLEAR (Tastencode?), so soll der Bildschirm gelöscht werden und das Programm neu beginnen, bevor es durch ENTER beendet wird.

Beispielprogramm:

```
NORMAL FLS AUTO REELL BOGENM MP
PROGRAM:PONG
:FKtAus
:PlotsAus
:AchsenAus
:LöBild
:0→Xmin
:1→△X
:If Xmax>95
:Then
:264→M
:165→N
:Else
:94→M
:63→N
:End
:0→Ymin
:1→△Y
:zufInt(10,M-10)→A
:zufInt(10,N-10)→B
:zufInt(2,5)→D
:zufInt(2,5)→E
:While 1
:Punkt-Ein(A,B,2)
:A+D→A
:B+E→B
:If A>M oder A<0
:Then
:A-D→A
:-D→D
:End
:If B>N oder B<0
:Then
:B-E→B
:-E→E
:End
:End
```