Kapitel 5 Grafik — Tillämpning: Få en punkt att studsa

I denna tillämpning för kapitel 5 ska vi bygga ett grafikbaserat program.

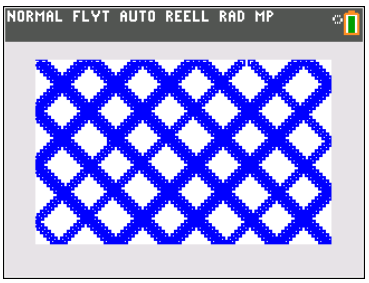
Syfte:
- Hur man kan ta reda på vilken typ av räknare som programmet körs på.
- Få en punkt att studsa runt på skärmen.

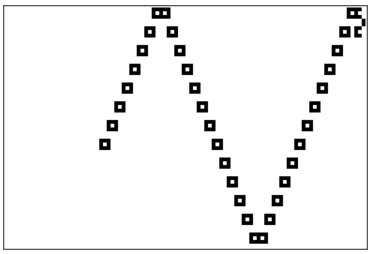
Lärarkommentar: Detta är ett ganska komplext projekt som har en del intressant inbyggd fysik. ”Partikeln” rör sig genom att man adderar delta x- och delta y-värden till koordinaterna under varje iteration hos loopen. Det är de horisontella och vertikala komponenterna av hastighet. När partikeln träffar en kant hos skärmen så blir den passande komponenten ”reverserad” (negerad) så att partikeln verkar ”studsa” vid skärmkanten.

Programmet

I det ursprungliga videospelet Pong studsade en pixel runt på skärmen. Spelarna styrde spelet med paddlar (staplar) och höll spelet igång. Detta program simulerar en ”studsande boll”. En punkt på skärmen rör sig i en sned linje över skärmen och när den träffar skärmkanten så byter den riktning så att det ser ut som sen studsar mot skärmen.
Det första problemet vi ska lösa gäller skärmstorleken. TI-84 Plus har en mindre skärm än TI-84 Plus C/CE-T. Kodsegmentet nedan detekterar skärmstorleken. För att skriva in Xmin och ΔX så trycker du först på [vars], väljer VAR och sedan 1:Fönster.

```
0→Xmin
1→ΔX
If Xmax>95
Then
        <Det är en TI-84 Plus C/CE-T>
Else
        <Det är en 84 Plus>
End
```





Samma program körs på två olika räknarmodeller.

När vi har bestämt räknare så ställer vi upp variablerna i Then- och Else-blocken och använder dessa i resten av programmet. Vi använder **M** för bredden och **N** för höjden på skärmen.

För räknarmodellen C eller CE-T: **264→M** och **165→N**
För räknarmodellen 84 Plus: **94→M** och **63→N**

Initialisera variablerna

Vi sätter också upp y-intervallet med bra värden:

```
0→Ymin
1→ΔY
```

Nu ställer vi också in startpunkten (A, B) slumpmässigt. Vi ser dock till att punkten inte hamnar kanterna på skärmen.

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

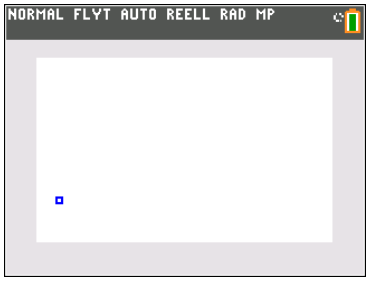
Vi sätter också upp två variabler som representerar rörelsen. Dessa kommer att adderas till punktens koordinater för att flytta punkten till en ny position.

slumpHel(2,5)→D förändring hos A
slumpHel(2,5)→E förändring hos B

Lärarkommentar: Detta är delta x- och delta y-värden.

Komma igång:

Nu är vi klara att sätta igång. Vi använder en oändlig loop...



While 1
Pkt-På(A,B,2) *stil 2 är stora fyrkanter*

<resten av programmet>

End

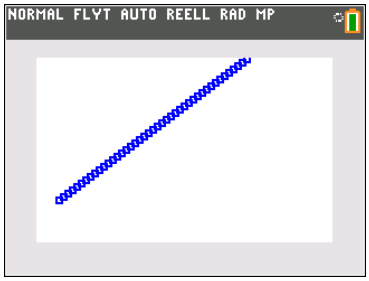
***Tips**: Se till att lägga in* **End** *samtidigt för att hålla ordning på dessa kommandon.*

Starta rörelsen

Vi lägger till förändringsvariablerna (D och E) till punktkoordinaterna i loopen och efter Pkt-På:

A+D→A
B+E→B

Detta ändrar punktens koordinater. Om du kör programmet nu kommer du att se punkten ge sig iväg i någon riktning till höger och utanför skärmen som på bilden till höger



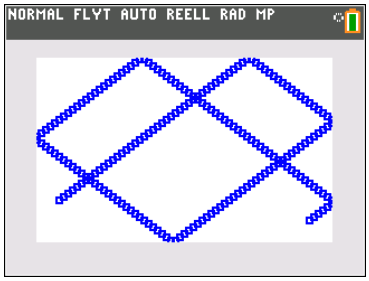
Studsarna

För att få punkten att upptäcka kanterna så lägger vi till If-satser…

If A>M eller A<0
Then

<Detta händer när punkten är utanför den högra eller vänstra sidan av skärmen>

End



Två saker måste inträffa innanför **Then**:

- Flytta punkten tillbaka på skärmen **A-D→A**
- Vända riktningen till den motsatta **- D→D**

Skriv därefter en liknande If-sats som hanterar y-koordinaten B och dess förändringsvariabel E.

Utvidgning

Lämna inte något spår…

Pkt-Av-kommandot gömmer en punkt. Lägg nu till en **Pkt-Av**-sats till programmet så att spåret av punkter inte visas utan bara den punkt som rör sig. Se till att du inte stänger av samma punkt som du sätter på. Stäng av den *föregående* punkten (du behöver två variabler till). Den nya koden behöver inpassas på rätt ställe i programmet.

Tappa den oändliga loopen…

getKey (programeditorn och I/O menyn) ger ett värde som representerar en tangent utan att pausa programmet som med **Prompt** och **Input.** enter har värdet 105 (rad 10, kolumn 5 på tangentbordet (knappsatsen).
Här är ett skelett för vad som behövs:

```
0→K
While K≠105

   <Ditt program ska in här>

getKey→K
End
```

Din uppgift nu är att bestämma var i programmet dessa satser ska in så att när du trycker enter så avslutas programmet.

Vad sägs om detta…

När du trycker på clear (vilket **getKey**-värde är det?) startar programmet på nytt med en tom skärm och mednya slumptalsvärden. När du trycker på enter avslutas programmet.

Programlistning

```
FnAv
DiagrAv
AxlarAv
RensaRitn
0→Xmin
1→ΔX
If Xmax>95
Then
264→M
165→N
Else
94→M
63→N
End
0→Ymin
1→ΔY
slumpHel(10,M-10)→A
slumpHel(10,N-10)→B
slumpHel(2,5)→D
slumpHel(2,5)→E
While 1
Pkt-På(A,B,2)
A+D→A
B+E→B
If A>M eller A<0
Then
A-D→A
-D→D
End
If B>N eller B<0
Then
B-E→B
-E→E
End
End
```