

Figure 1: Bok 2

#	Beskriving
5	line och stroke
6	Låt bollen åka åt höger i all evighet
7	rect och fill

Contents

Förord	1
line och stroke	2
Låt bollen åka åt höger i all evighet	22
rect och fill	40

Förord

Detta är en bok om Processing för tonåringar. Processing är ett programmeringsspråk. Denna bok lär dig det programmeringsspråket.

Om den här boken

Denna bok är licensierad av CC-BY-NC-SA.



Figure 1: Licensen för denna bok

(C) Richèl Bilderbeek och alla lärare och alla elever

Med det här häftet kan du göra vad du vill, så länge du hänvisar till originalversionen på denna webbplats: https://github.com/richelbilderbeek/processing_foer_tonaaringar. Detta häfte kommer alltid att förbli gratis, fritt och öppet.

Det är fortfarande en lite slarvig bok. Det finns stafvel och *layouten är inte alltid vacker*. Eftersom den här boken finns på en webbplats kan alla som tycker att den här boken är för slarvig göra den mindre slarvig.

rect och fill: slutuppgift

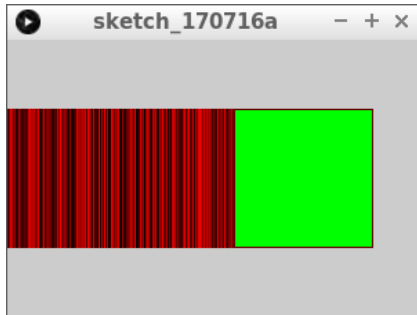


Figure 33: Slutuppgift `rect` och `fill`

Nu ska du ändra både linjefärg och ifyllnadsfärg. Linjefärgen ska vara röd, men det röda värdet ska vara slumpmässigt mellan 0 och 256. Ifyllnadsfärgen ska vara grön, och öka från 0 till 256 med 3 steg åt gången. Om grönvärdet är större än 256, måste den sättas till 0 igen.

line och stroke



Figure 2: Moria, ett av de allra första spelen med färg

Under den här lektionen kommer vi att lära oss att rita färgade linjer.

line och stroke: uppgift 1

Kör den här koden:

```
void setup()
{
  size(300, 200);
}

void draw()
{
  line(0, 100, 300, 200);
}
```



line(0, 100, 300, 200);

‘Kära dator, rita ut en linje från (0, 100) till (300, 200).’



(100, 200) är pixeln som är 100 pixlar till höger om och 200 pixlar under fönstrets övre vänstra hörn

rect och fill: lösning 8

```
float x = -50;
float gron = 0;

void setup()
{
  size(300, 200);
}

void draw()
{
  stroke(255, 255, 0);
  fill(x, gron, random(256));
  rect(x - 50, height / 2 - 50, 100, 100);
  x = x + 1;
  gron = gron + 2;
  if (x > width + 50)
  {
    x = -50;
  }
  if (gron > 255)
  {
    gron = 0;
  }
}
```

rect och fill: uppgift 8

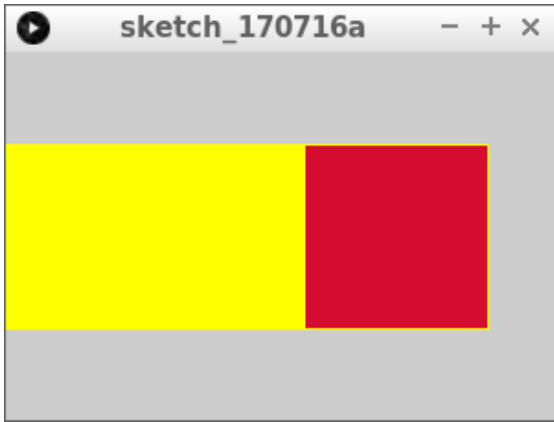


Figure 32: Uppdrag 8

Den ifyllda färgen ska nu ha ett blåvärde som är ett slumpmässigt tal mellan 0 och 256.

line och stroke: lösning 1

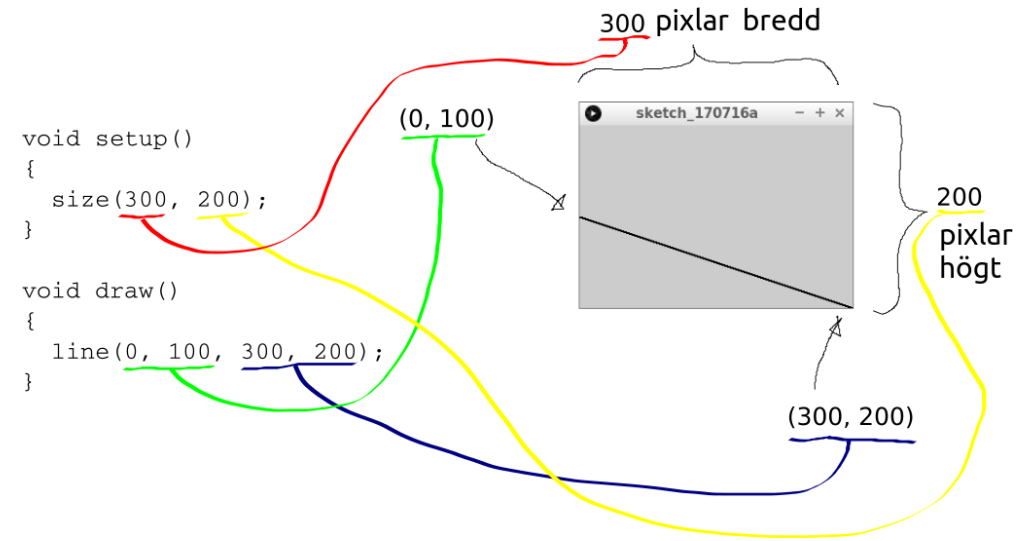


Figure 3: Lösning 1

line och stroke: uppgift 2

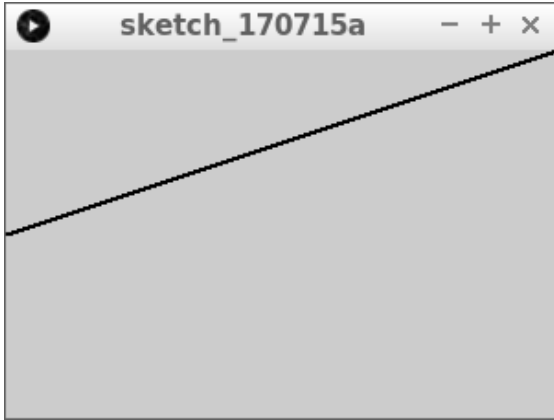


Figure 4: Uppgift 2

Ändra linjen så att den går till det övre högra hörnet, istället för till den nedre högra hörnet.

rect och fill: lösning 7

```
float x = -50;
float gron = 0;

void setup()
{
  size(300, 200);
}

void draw()
{
  stroke(255, 255, 0);
  fill(x, gron, 255);
  rect(x - 50, height / 2 - 50, 100, 100);
  x = x + 1;
  gron = gron + 2;
  if (x > width + 50)
  {
    x = -50;
  }
  if (gron > 255)
  {
    gron = 0;
  }
}
```

rect och fill: uppgift 7

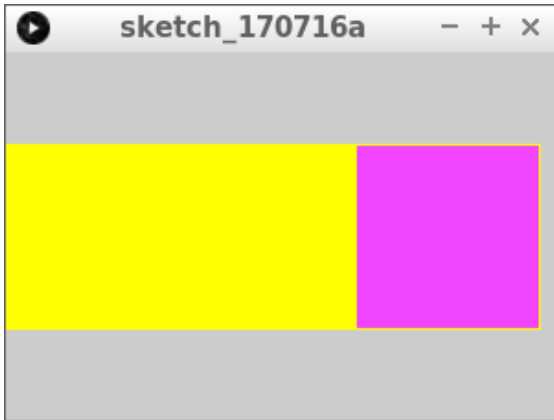


Figure 31: Uppdrag 7

Variabeln `gron` kan inte bli högre än 255. Gör en ny `if`-sats: om `gron` är mer än 255, ska `gron` bli 0.

line och stroke: lösning 2

```
void setup()
{
  size(300, 200);
}

void draw()
{
  line(0, 100, 300, 0);
}
```

line och stroke: uppgift 3

Ändra linjen så att den börjar längst ner till vänster istället för i mitten på vänster sida.

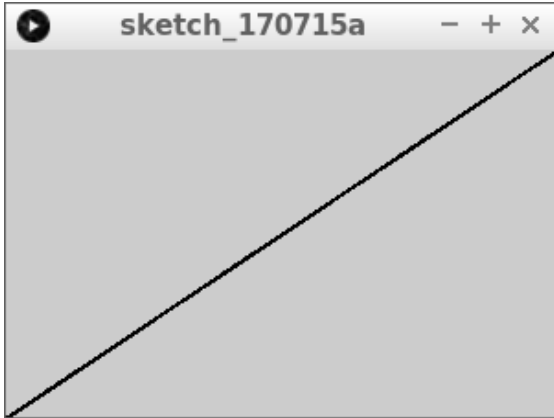


Figure 5: Uppgift 3

rect och fill: lösning 6

```
float x = -50;
float gron = 0;

void setup()
{
  size(300, 200);
}

void draw()
{
  stroke(255, 255, 0);
  fill(x, gron, 255);
  rect(x - 50, height / 2 - 50, 100, 100);
  x = x + 1;
  gron = gron + 2;
  if (x > width + 50)
  {
    x = -50;
  }
}
```


rect och fill: uppgift 6

Gör en ny variabel som heter **gron** (datorn gillar inte ord med ö). **gron** ska ha startvärdet noll. **gron** är det andra talet i **fill** (platsen efter 0). Varje gång ökar **gron** med två.

line och stroke: lösning 3

```
void setup()
{
  size(300, 200);
}

void draw()
{
  line(0, 200, 300, 0);
}
```

line och stroke: uppgift 4

Låt linjen gå från nedre vänster till överst till höger, men använd nu `width` och `height` istället.

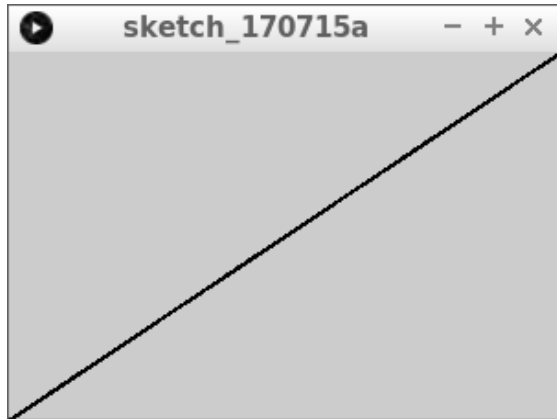


Figure 6: Uppgift 4

rect och fill: lösning 5

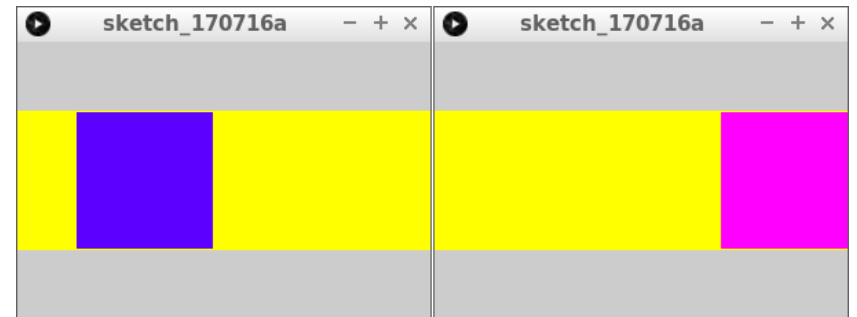


Figure 30: Uppdrag 5

```
float x = -50;

void setup()
{
  size(300, 200);
}

void draw()
{
  stroke(255, 255, 0);
  fill(x, 0, 255);
  rect(x - 50, height / 2 - 50, 100, 100);
  x = x + 1;
  om (x > bredd + 50)
  {
    x = -50;
  }
}
```



`fill(0, 128, 255);` 'Kära dator, fyll i med en färg utan rött, som är halvt grön och helt blå.'

rect och fill: uppgift 5

Lägg till följande text under `stroke(255, 255, 0);`:

```
fill(x, 0, 255);
```

line och stroke: lösning 4

```
void setup()
{
  size(300, 200);
}

void draw()
{
  line(0, height, width, 0);
}
```

line och stroke: uppgift 5

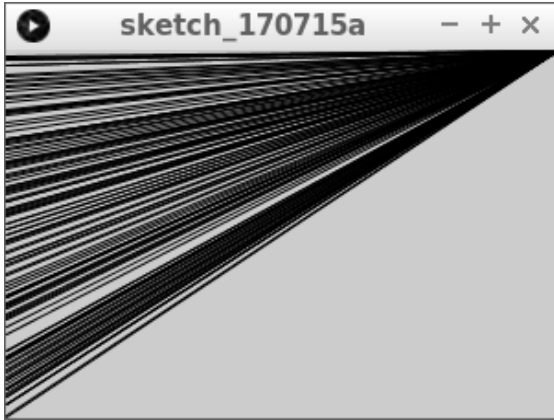


Figure 7: Uppgift 5

Låt nu linjen starta i vänstra kanten på en slumpmässig höjd. Du gör detta med `random`.

rect och fill: lösning 4

```
float x = -50;

void setup()
{
  size(300, 200);
}

void draw()
{
  stroke(255, 255, 0);
  rect(x - 50, height / 2 - 50, 100, 100);
  x = x + 1;
  if (x > width + 50)
  {
    x = -50;
  }
}
```

rect och fill: uppgift 4

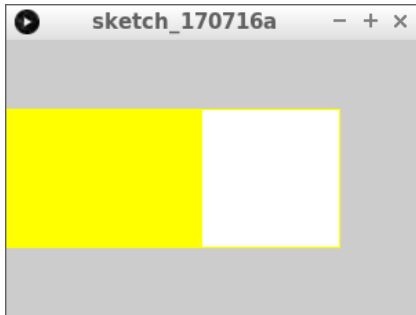


Figure 28: Uppdrag 4

Ta nu bort den dolda bollen. På denna plats, använd `stroke` igen, men nu med färgen gul. Kolla på färgbollen nedanför:

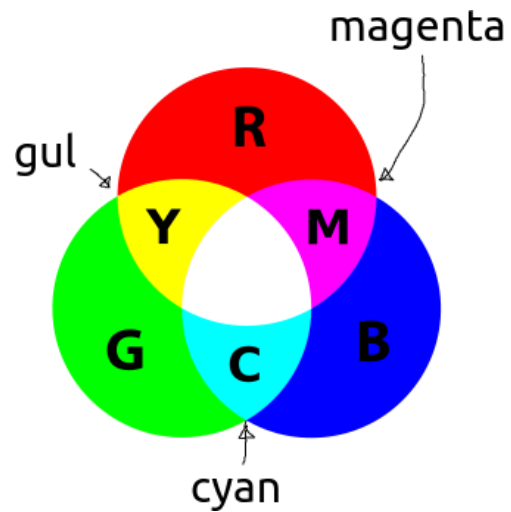


Figure 29: Färgbollen

line och stroke: lösning 5

```
void setup()
{
  size(300, 200);
}

void draw()
{
  line(0, random(height), width, 0);
}
```

line och stroke: uppgift 6

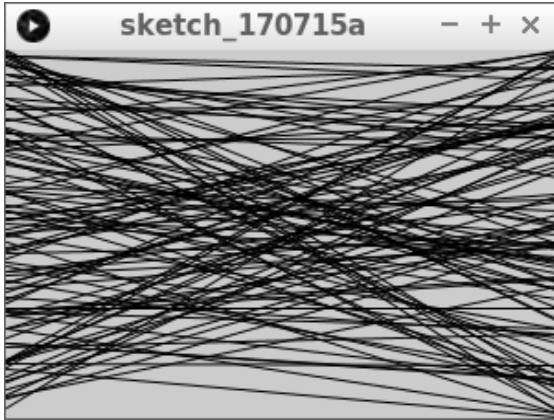


Figure 8: Uppgift 6

Låt linjen nu också sluta på en slumpmässig höjd i högra kanten.

rect och fill: lösning 3

```
float x = -50;

void setup()
{
  size(300, 200);
}

void draw()
{
  ellipse(x, height / 2, 100, 100);
  rect(x - 50, height / 2 - 50, 100, 100);
  x = x + 1;
  if (x > width + 50)
  {
    x = -50;
  }
}
```

rect och fill: uppgift 3

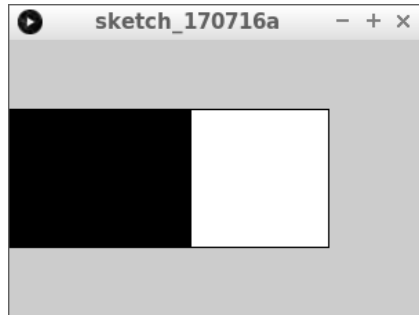


Figure 27: Uppdrag 3

Rita ut fyrkanten ovanpå bollen. Gör det genom att minska x och y-koordinaterna med 50.

line och stroke: lösning 6

```
void setup()
{
  size(300, 200);
}

void draw()
{
  line(0, random(height), width, random(height));
}
```

line och stroke: uppgift 7

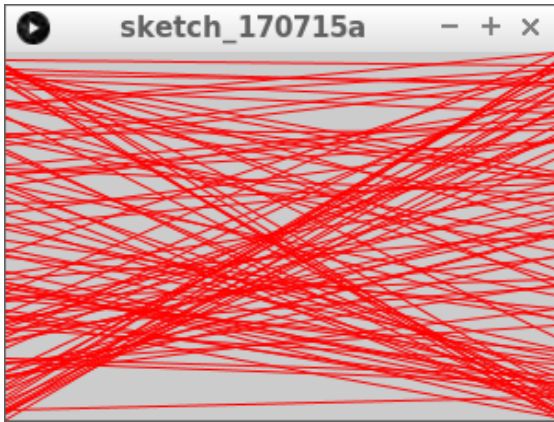


Figure 9: Uppgift 7

Precis ovanför meningen med `line`, skriv nu texten `stroke(255, 0, 0);`.

rect och fill: lösning 2

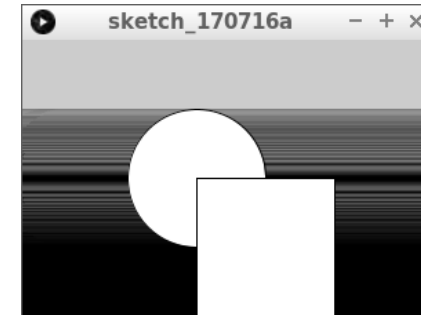


Figure 26: lösning 2

```
float x = -50;

void setup()
{
  size(300, 200);
}

void draw()
{
  ellipse(x, height / 2, 100, 100);
  rect(x, height / 2, 100, 100);
  x = x + 1;
  if (x > width + 50)
  {
    x = -50;
  }
}
```


rect och fill: uppgift 2

Lägg till följande text, efter `ellipse(x, height / 2, 100, 100);`:

```
rect(x, height / 2, 100, 100);
```



<code>rect (100, 200, 300, 400)</code>	‘Kära dator, rita ut en fyrkant med övre vänstra hörnet på (100, 200), 300 pixlar bred och 400 pixlar hög.’
--	---

line och stroke: lösning 7

```
void setup()
{
  size(300, 200);
}

void draw()
{
  stroke(255, 0, 0);
  line(0, random(height), width, random(height));
}
```



<code>stroke (255, 0, 0);</code>	‘Kära dator, färga linjerna röda.’
<code>stroke (255, 0, 0);</code>	‘Kära dator, färga linjerna helt röda, utan grönt och utan blått.’

line och stroke: uppgift 8

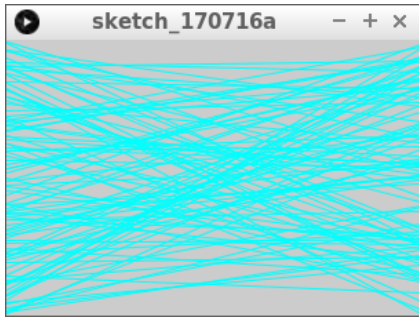


Figure 10: Uppgift 8

Gör linjerna cyan (ljusblå) nu. Titta på figuren 'Färgcirkel' hur du gör det

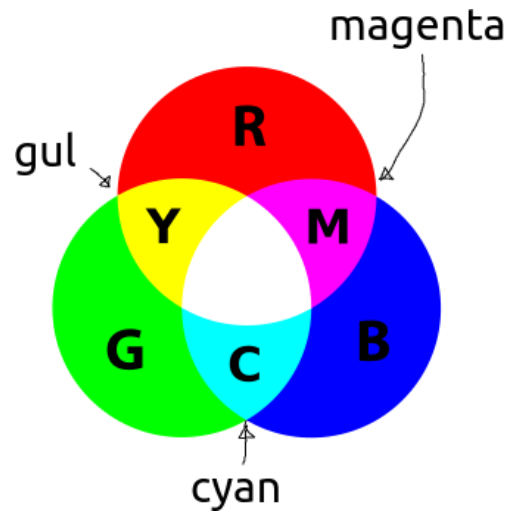


Figure 11: Färgcirkel

rect och fill: lösning 1

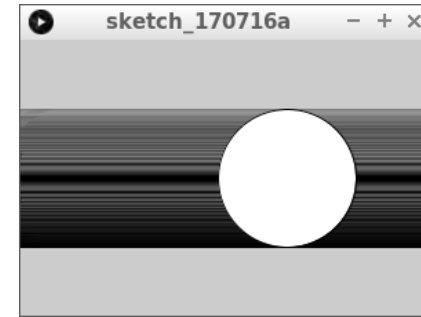


Figure 25: lösning 1

rect och fill: uppgift 1

Kör denna kod.

```
float x = -50;

void setup()
{
  size(300, 200);
}

void draw()
{
  ellipse(x, height / 2, 100, 100);
  x = x + 1;
  if (x > width + 50)
  {
    x = -50;
  }
}
```

line och stroke: lösning 8

```
void setup()
{
  size(300, 200);
}

void draw()
{
  stroke(0, 255, 255);
  line(0, random(height), width, random(height));
}
```



```
stroke
(0, 255,
 255);
stroke
(0, 255,
 255);
```

‘Kära dator, färga linjerna cyan.’

‘Kära dator, färga linjerna utan rött, helt gröna och helt blåa.’

line och stroke: uppgift 9

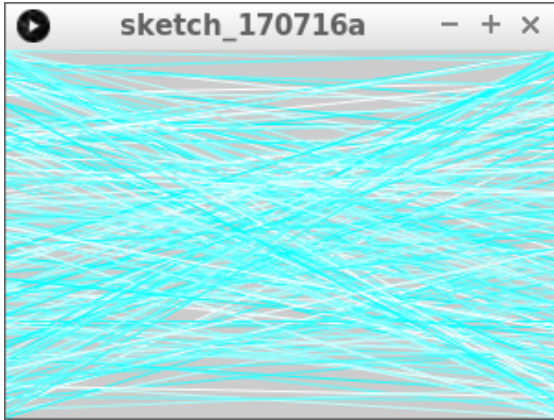


Figure 12: Uppgift 9

Sätt nu det röda värdet till ett slumpmässigt tal mellan 0 och 256.

rect och fill

Här ser du ett av de mest berömda spel någonsin, som är gjort med enkla fyrkanter som är ifyllda med olika färger:

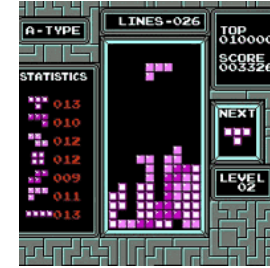


Figure 24: Tetris

Du kan göra en fyrkant med fyra linjer, men `rect` är lättare.

Låt bollen åka åt höger i all evighet: slutuppgift

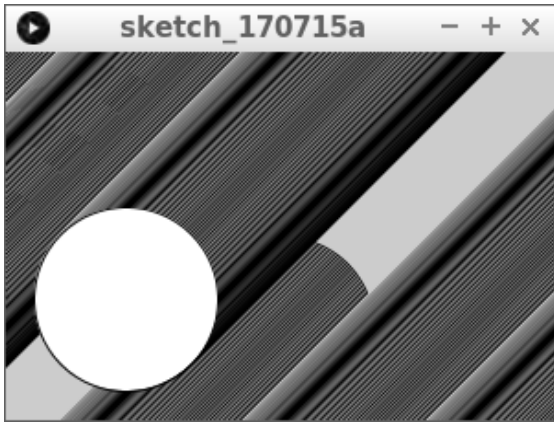


Figure 23: slutuppgift 'Låt bollen åka åt höger i all evighet'

Låt nu bollen åka snett neråt vänster i all evighet.

line och stroke: lösning 9

```
void setup()
{
  size(300, 200);
}

void draw()
{
  stroke(random(256), 255, 255);
  line(0, random(height), width, random(height));
}
```

line och stroke: slutuppgift

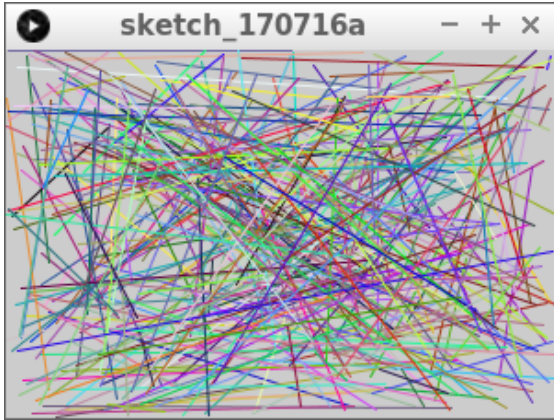


Figure 13: Slutuppgift line och stroke

Låt nu linjer börja och sluta på slumpmässiga platser. Linjefärgen måste också vara slumpmässig.

Låt bollen åka åt höger i all evighet: lösning 6

```
float x = -50;
float y = -50;

void setup()
{
  size(300, 200);
}

void draw()
{
  ellipse(x,y,100,100);
  x = x + 1;
  y = y + 1;
  if (x > 350)
  {
    x = -50;
  }
  if (y > 250)
  {
    y = -50;
  }
}
```

Låt bollen åka åt höger i all evighet: uppgift 6

Oj, nu när vi har skapat en variabel `x` och en variabel `y`, låt oss använda båda samtidigt!

När vi slår ihop kod gäller följande regler:

- allt som finns ovanför `setup`-funktionen ska vara kvar där
- allt som finns inuti `setup`-funktionen måste vara kvar inuti `setup`-funktionen
- allt som finns inuti funktionen `draw` måste vara kvar inuti funktionen `draw`

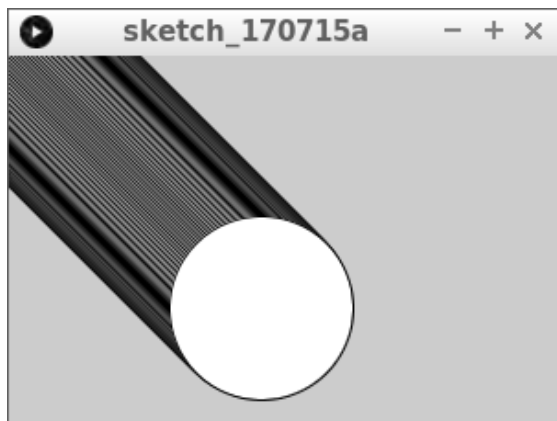


Figure 22: Uppgift 6

- Slå ihop koden för “Låt bollen åka åt höger i all evighet” med “Låt bollen åka neråt i all evighet”
- Ändra koden så att bollen också åker neråt

Låt bollen åka åt höger i all evighet

Under den här lektionen ska vi få en boll att åka åt höger i all evighet.

Under den här lektionen lär vi oss också vad en `if`-sats är. Du kan (nästan) inte programmera utan `if`-satser.

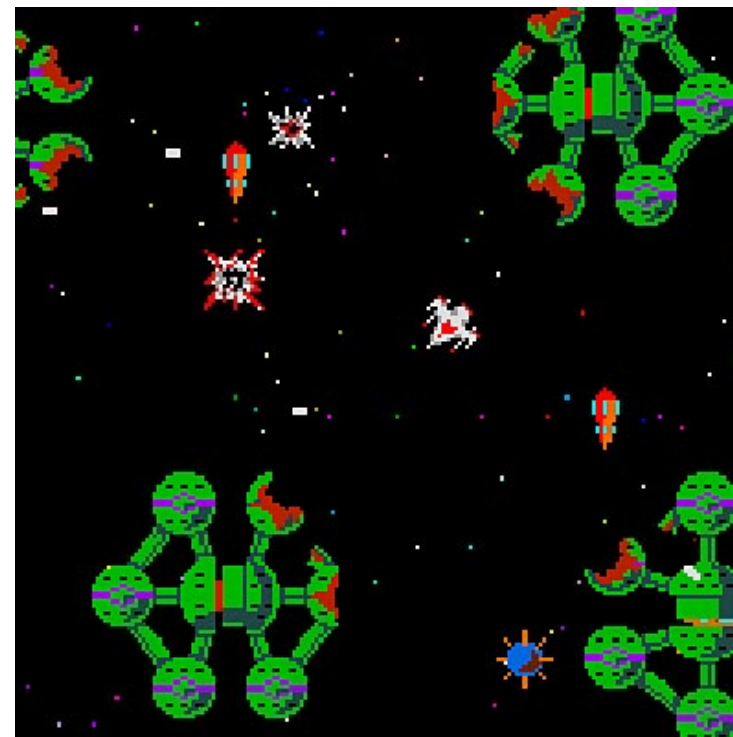


Figure 14: På Bosconian kan du också åka åt höger i all evighet

Låt bollen åka åt höger i all evighet: intro

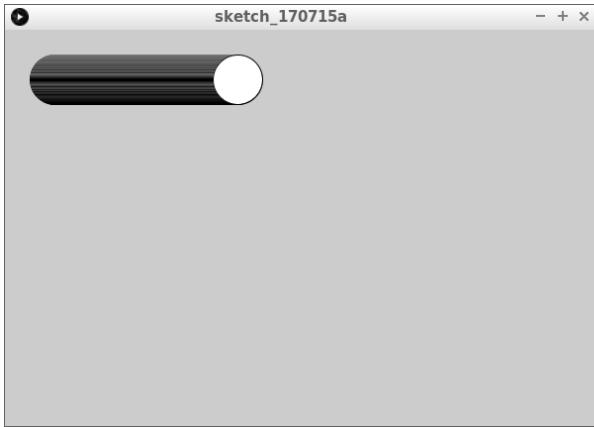


Figure 15: Låt bollen åka höger

Detta är en boll som åker åt höger:

```
float x = 50;

void setup()
{
  size(600, 400);
}

void draw()
{
  ellipse(x, 50, 50, 50);
  x = x + 1;
}
```

Nackdel: bollen återvänder aldrig till fönstret.

Låt bollen åka åt höger i all evighet: lösning 5

```
float y = -50;

void setup()
{
  size(300, 200);
}

void draw()
{
  ellipse(50, y, 100, 100);
  y = y + 1;
  if (y > 250)
  {
    y = -50;
  }
}
```


Låt bollen åka åt höger i all evighet: uppgift 5

Vi fick en boll att åka åt höger och åt vänster när vi ändrade koordinaten **x**. Bollen kan också åka neråt och uppåt om vi ändrar y-koordinaten.

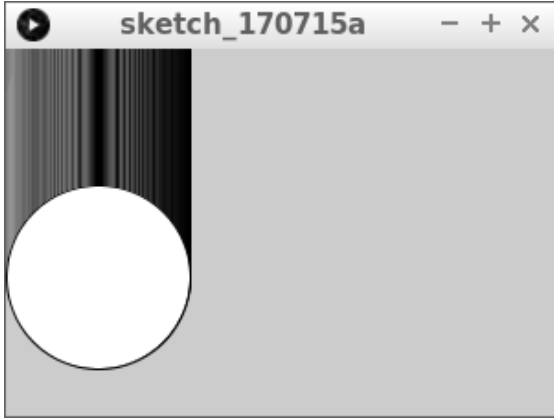


Figure 21: Uppgift 5

Skriv ett program där en boll åker neråt i all evighet:

- gör skärmen 300 pixlar bred och 200 pixlar hög
- använd en variabel som heter 'y'
- ersätt koden `ellipse(x, 50, 100, 100)` med `ellipse(50, y, 100, 100)`
- om bollen åker neråt och utanför fönstret så måste bollen börja om uppifrån igen

Vi vill kunna säga, "Kära dator, **om** bollen är för långt bort åt höger, ska du teleportera bollen till vänster". **if** är engelska för 'om'.

Så det här kan bli:

```
if (x > 200)
{
  x = 100;
}
```

Tecknet > betyder 'större än'. Mer exakt blir det: "Kära dator, **om** x är större än 200, sätt x till 100. **if** är engelska för 'om'.

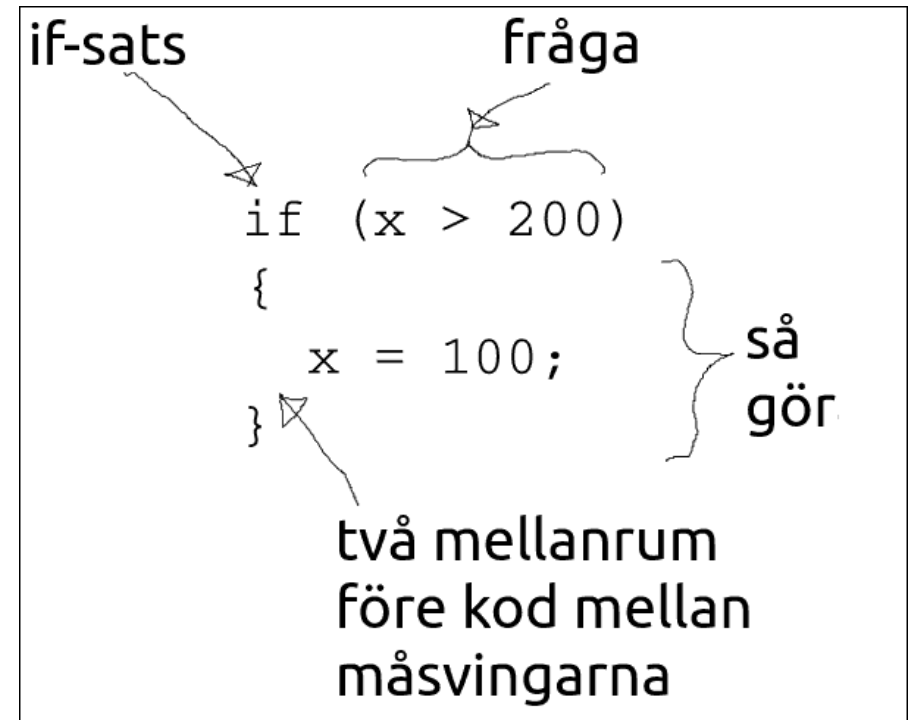


Figure 16: Ett om



```
if (x > 200) {}
```

‘Kära dator, om x är större än 200, gör det som står inom måsvingarna.’

```
x = 100;
```

‘Kära dator, låt x vara 100.’

Låt bollen åka åt höger i all evighet: lösning 4

Detta är en evighetsboll som åker åt vänster:

```
float x = 650;

void setup()
{
  size(600, 400);
}

void draw()
{
  ellipse(x, 50, 100, 100);
  x = x - 1;
  if (x < -50)
  {
    x = 650;
  }
}
```



$x = x - 1$	‘Kära dator, minska x med ett.’
$x -= 1$	‘Kära dator, minska x med ett.’
$x-$	‘Kära dator, minska x med ett.’
$--x$	‘Kära dator, minska x med ett.’



Figure 20: Uppgift 4

Gör en boll som åker åt vänster i all evighet:

- Bollen startar utanför fönstret
- Bollen åker helt utanför fönstret
- Om bollen åker utanför fönstret ska den omedelbart börja om igen på andra sidan

Låt bollen åka åt höger i all evighet: uppgift 1

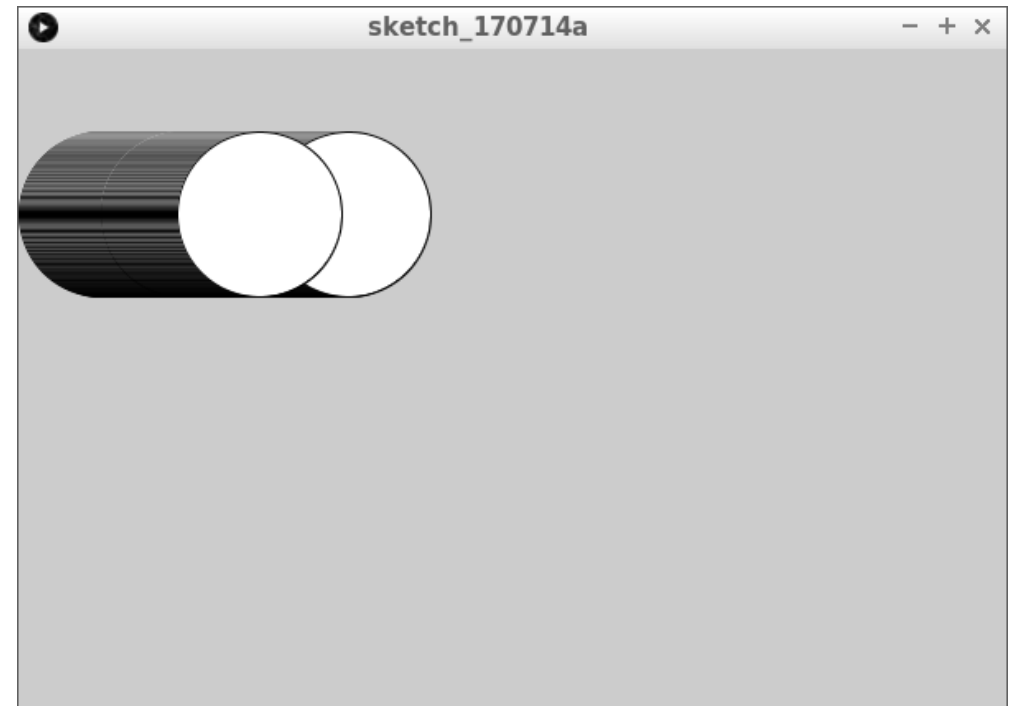


Figure 17: Uppgift 1

Skriv `if` inuti programmets kod. Skriv `if` i slutet av `draw`, före den avslutande måsvingen (`}`).

Låt bollen åka åt höger i all evighet: lösning 1

Koden blir då:

```
float x = 50;

void setup()
{
  size(600, 400);
}

void draw()
{
  ellipse(x,100,100,100);
  x = x + 1;
  if (x > 200)
  {
    x = 100;
  }
}
```

Låt bollen åka åt höger i all evighet: uppgift 4

Lurad! Även om lektionen heter 'Låt bollen åka åt höger i all evighet', så ska nu bollen byta håll.

Vi ska nu programmera en boll som åker åt vänster i all evighet.

Det du behöver titta på nu är `if`-uttalandet för att kunna avgöra när `x` är för litet:

```
if (x < 100)
{
  x = 500;
}
```

Med detta säger du: 'Kära dator, om `x` är mindre än (<) hundra, sätt `x` till femhundra istället'.



```
if (x < 100) {}
```

'Kära dator, om' `x` 'är mindre än 100, gör det som står innanför måsvingarna.'

Låt bollen åka åt höger i all evighet: lösning 3

Ändra if (x > 200) till if (x > 650).

```
float x = -50;

void setup()
{
  size(600, 400);
}

void draw()
{
  ellipse(x, 50, 100, 100);
  x = x + 1;
  if (x > 650)
  {
    x = 0;
  }
}
```

Låt bollen åka åt höger i all evighet: uppgift 2

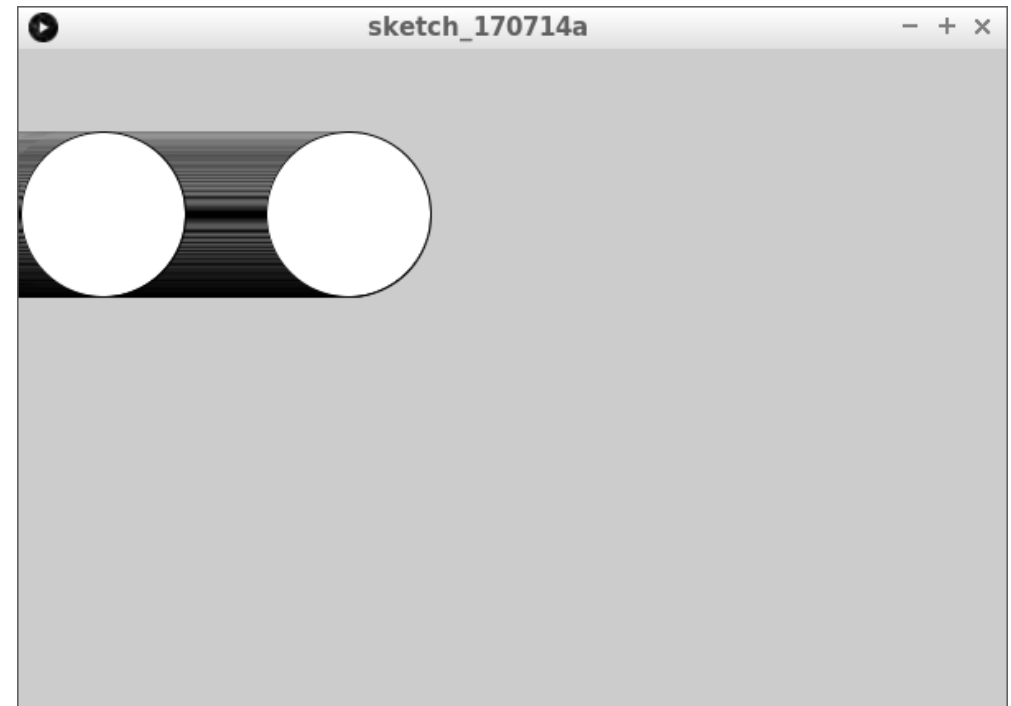


Figure 18: Uppgift 2

Se till att bollen startar allra längst till vänster i fönstret

Låt bollen åka åt höger i all evighet: lösning 2

- Ändra float $x = 50$ till float $x = 0$ eller float $x = -50$: båda är bra.
- Ändra $x = 100$ till $x = 0$ eller $x = -50$: båda är bra.

```
float x = 50;

void setup()
{
  size(600, 400);
}

void draw()
{
  ellipse(x,100,100,100);
  x = x + 1;
  if (x > 200)
  {
    x = 0;
  }
}
```

Låt bollen åka åt höger i all evighet: uppgift 3

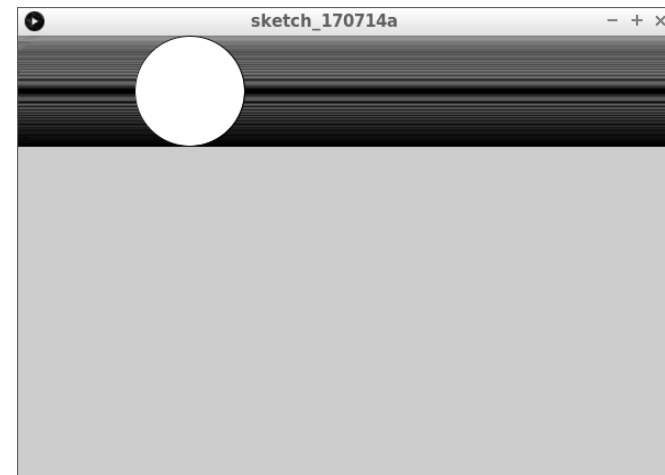


Figure 19: Uppgift 3

Se till att bollen åker hela vägen till höger innan den hoppar till vänster sida av fönstret