Programmering

KOMPLEMENT TILL MATTE DIREKT BORGEN UPPLAGA 2

SANOMA UTBILDNING
Postadress: Box 38013, 100 64 Stockholm
Besöksadress: Rosenlundsgatan 54
www.sanomautbildning.se
info@sanomautbildning.se

Order/Läromedelsinformation
Telefon: 08-587 642 10

Författare och redaktör: Linda Kempe
Grafisk form och illustrationer: Anna Hild

Till er som använder Programmering

Programmering är ett komplement till basläromedlet Matte Direkt Borgen åk 4–6. Komplementet innehåller analoga programmerings-uppdrag och följer Lgr11 reviderad 2018.

Materialet är uppbyggt med förslag på färdiga lektionsupplägg. *Programmering* är ett kopieringsunderlag som innehåller instruktioner till läraren, arbetsblad till eleverna samt facit. Under 2019 kommer flera lektioner att läggas ut kontinuerligt.

I instruktionerna till läraren finns:

- Uppskattad tidsåtgång
- Moment ur Lgr11 reviderad 2018
- Arbetsgång. Rubrikerna A, B, C och D informerar om arbetsgång och kan skrivas på tavlan som instruktion till eleverna.
- Extra uppgifter. Dessa kan ha olika karaktär men består ofta av förslag på gemensam aktivitet, öva mera och/eller utmaning.
- Facit

Arbetsbladens upplägg:

- Begreppsrutor och faktarutor
- En progression mellan uppdragen
- Eleverna får olika uppdrag där de ska hjälpa Turtle att ta sig fram i en visuell analog programmeringsmiljö.

Lycka till!

Linda Kempe
Sanoma Utbildning

Skriva kod och hitta buggar

LEKTION 1

Tidsåtgång: ca 90 minuter.
I slutet av lektionen finns förslag på extra uppgifter som du kan använda vid behov eller vid andra tillfällen.

Moment ur Lgr11 reviderad 2018:
Algebra: Hur algoritmer kan skapas och användas vid programmering.
Programmering i visuella programmeringsmiljöer.

A. Gemensam introduktion

1. Diskussion: Vad i klassrummet är programmerat?

EXEMPEL dator, bildskärm, tv, fjärrkontroll, tidsinställt fläktsystem. En del elever kan nämna datorspel och associerar programmering till att få programmera egna spel på dator. Här är det viktigt att behålla dessa elevers entusiasm samtidigt som du poängterar att vi under lektionerna kommer att lära oss principer inom programmering.

2. Öva praktiskt: illustrering av ”exakta instruktioner”

Material: klossar.

INTRODUKTION Berätta för eleverna att en algoritm består av exakta steg-för-steg-instruktioner som kallas för kod. Du styr till exempel en dator med en kod.

ÖVNING Programmeraren och maskinen.

Dela in eleverna två och två. Placera dem rygg mot rygg och med varsin uppsättning av lika många klossar. Den som först är programmeraren bygger en konstruktion av sina klossar och beskriver för maskinen hur konstruktionen ser ut. Maskinen bygger en likadan av sina klossar. När eleverna är klara jämför de konstruktionerna med varandra. Blev de likadana? Var instruktionerna tydliga? Eleverna byter roller och upprepar uppgiften.

REFLEKTION ENSKILT/I PAR/ALLA

- Liknar konstruktionerna varandra?
  - Om ja, vad bidrog till likadana konstruktioner?
    Svar: exakta instruktioner/exakt kod.
  - Om nej, vad kunde Du ha gjort annorlunda?
    Svar: tydligare instruktioner/kod).

När konstruktionerna ser olika ut så innehåller koden fel instruktioner/buggar. Genom att gå tillbaka och rätta instruktionerna blir koden tydlig, buggarna försvinner och konstruktionerna blir likadana.

SAMMANFATTNING

- Vad lärde vi oss?
  Svar: Vikten av att vara tydliga med de stegvisa instruktionerna. Samma sak gäller vid programmering av ett datorprogram.

3. Begrepp

Skriv begreppen på tavlan. Diskutera, resonera och exemplifiera begreppen. Detta underlättar för eleverna när de arbetar vidare med uppgifterna i arbetsbladet.

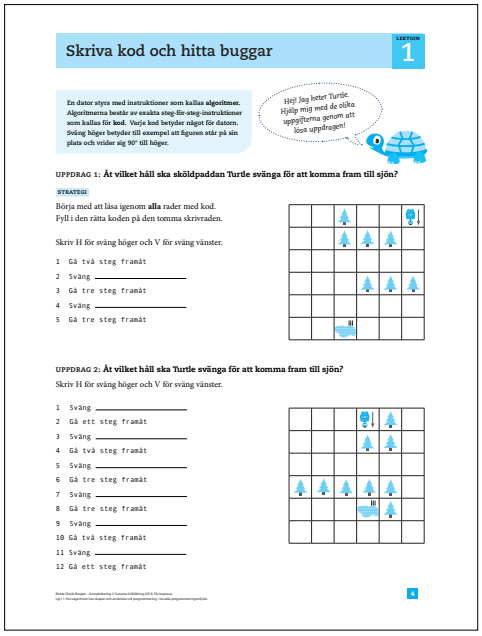
- **Algoritm**, **kod** och **bugg**.
  Algoritm består av exakta steg-för-steg-instruktioner, som kallas för ***kod***. Varje kod betyder något för en dator. Om datorprogrammet inte kan tolka eller tolkar instruktionerna fel, har det blivit en ***bugg***.
  I den praktiska övningen använde vi oss av det talade språket som kod, men koden kunde även ha bestått av symboler. Datorprogrammen som vi använder oss av är skrivna med programmeringsspråk som till exempel Phyton och C++.

B. Arbeta med arbetsblad

Material: blyertspennor och arbetsblad ”Lektion 1”.

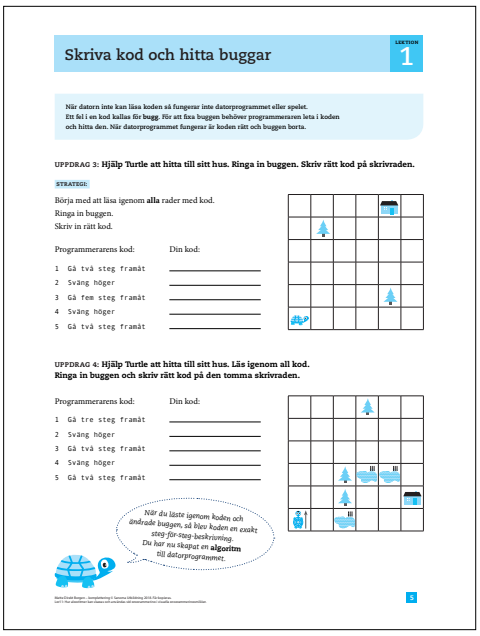
I detta arbetsblad får eleverna följa sköldpaddan Turtle i en analog programmeringsmiljö. Turtle kommer med sju olika uppdrag som eleverna ska utföra.

- *Uppdrag 1 och 2:* Eleverna lär sig ”sväng höger” och ”sväng vänster”.
  Förtydliga för eleverna: Koden ”sväng höger” innebär att figuren roterar 90° runt sin egen axel. Den tar alltså inte ett steg till höger, vilket är en vanlig missuppfattning.



Uppdrag 1 och 2

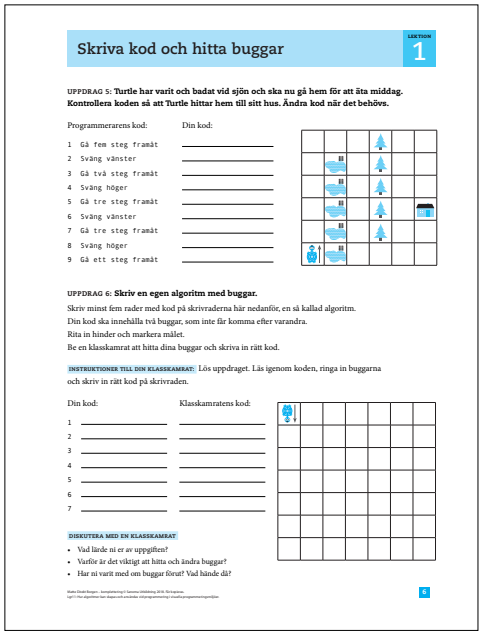
- *Uppdrag 3 och 4:* Eleverna läser kod, hittar bugg, skriver rätt kod.



Uppdrag 3 och 4

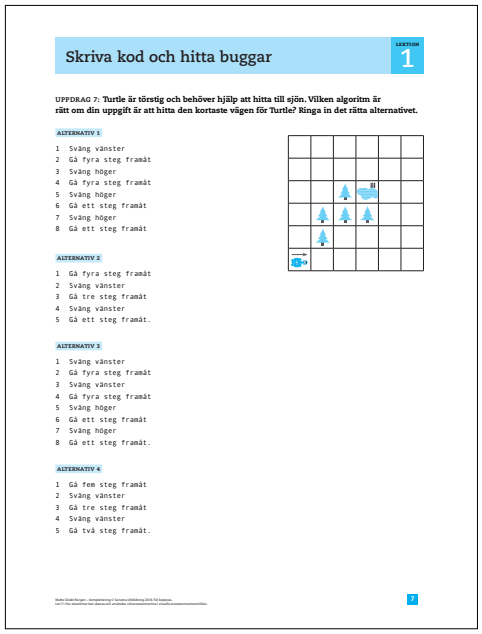
- *Uppdrag 5 och 6:* Eleverna läser kod, hittar flera buggar, skriver rätt kod. Eleverna skriver egen kod med buggar, löser varandras kod och rättar koden.

TIPS Bestäm gärna paren i förväg så att ingen blir utanför.



Uppdrag 5 och 6

- *Uppdrag 7:* Eleverna behöver läsa all kod, resonera och avgöra vilken kod som ger Turtle den kortaste vägen hem.



Uppdrag 7

C. Genomgång av arbetsblad

D. Gemensam reflektion

- Vilka **begrepp** har du lärt dig idag?
  Svar: algoritm, kod och bugg.
- Kan du ge exempel på begreppet **algoritm**?
  Svar: exakta steg-för-steg-instruktioner.
- Kan du ge exempel på vad begreppet **kod** innebär?
  Svar: en instruktion.
- Kan du ge exempel på vad begreppet **bugg** innebär?
  Svar: en felaktig del i en kod.
- Vad har du lärt dig idag?
  Svaren kan variera.

EXTRA UPPGIFTER:

Material: penna, papper eller räknehäftet.

AKTIVITET Gemensam uppgift. Tusenfoting. Alla elever ställer sig i en kö och ser framåt. Eleven som står först i kön vänder sig om med ansiktet mot de andra. Uppgiften för den eleven är att visa rörelser och säga instruktioner så att tusenfotingen/eleverna rör sig lika.

Utmaning: försök att enbart säga instruktioner.

ÖVA MERA Enskild-/paruppgift. Skriva egna algoritmer. Eleverna skriver ner en algoritm, det vill säga den exakta steg-för-steg-instruktionen, för hur man borstar tänderna, sitter ner på en stol eller tar på sig gummistövlar. Eleverna avslutar med att testa varandras algoritmer.

UTMANING Enskild-/paruppgift. Skriva egna algoritmer. Eleverna skriver ner algoritmen för 11 + 14. Algoritmen ska innehålla exakta steg-för-steg-instruktioner för hur de löser uträkningen. Eleverna diskuterar och jämför sina algoritmer. Du kan variera uppgiften med t.ex. $12 - 6$, $2 \cdot 3$, $\frac{4}{2}$.

Skriva kod och hitta buggar

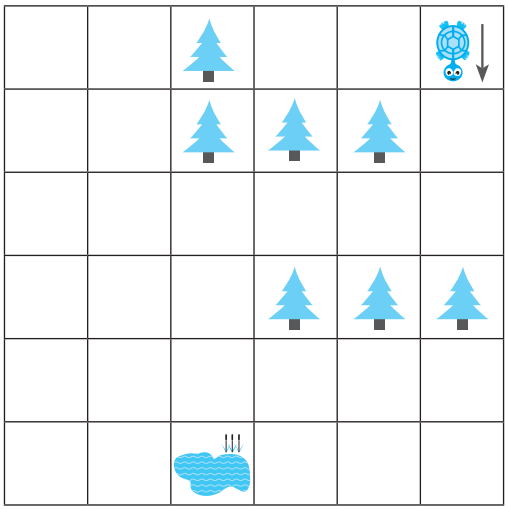
LEKTION 1

En dator styrs med instruktioner som kallas algoritmer. Algoritmerna består av exakta steg-för-steg-instruktioner som kallas för kod. Varje kod betyder något för datorn. Sväng höger betyder till exempel att figuren står på sin plats och vrider sig 90° till höger.

UPPDRAG 1: Åt vilket håll ska sköldpaddan Turtle svänga för att komma fram till sjön?

STRATEGI

Börja med att läsa igenom **alla** rader med kod.
Fyll i den rätta koden på den tomma skrivraden.

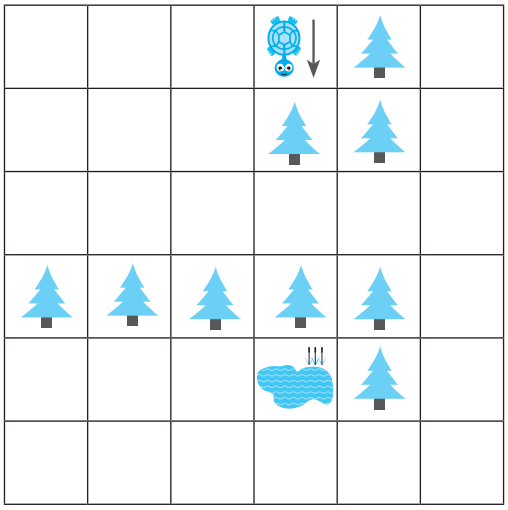


Skriv H för sväng höger och V för sväng vänster.

```
1  Gå två steg framåt
2  Sväng _____
3  Gå tre steg framåt
4  Sväng _____
5  Gå tre steg framåt
```

UPPDRAG 2: Åt vilket håll ska Turtle svänga för att komma fram till sjön?

Skriv H för sväng höger och V för sväng vänster.



```
1  Sväng _____
2  Gå ett steg framåt
3  Sväng _____
4  Gå två steg framåt
5  Sväng _____
6  Gå tre steg framåt
7  Sväng _____
8  Gå tre steg framåt
9  Sväng _____
10 Gå två steg framåt
11 Sväng _____
12 Gå ett steg framåt
```


Lgr11: Hur algoritmer kan skapas och användas vid programmering i visuella programmeringsmiljöer.

Skriva kod och hitta buggar

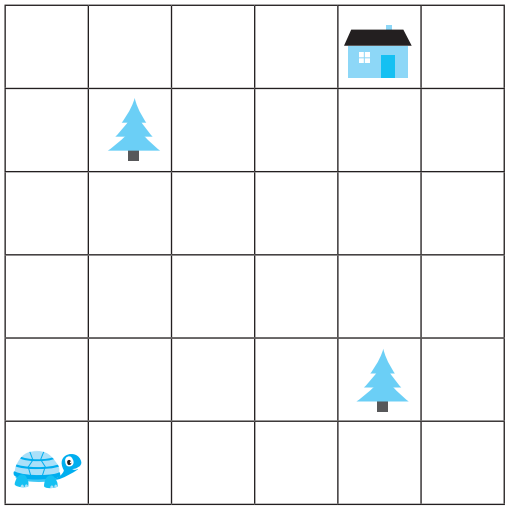
LEKTION 1

När datorn inte kan läsa koden så fungerar inte datorprogrammet eller spelet.
Ett fel i en kod kallas för **bugg**. För att fixa buggen behöver programmeraren leta i koden och hitta den. När datorprogrammet fungerar är koden rätt och buggen borta.

UPPDRAG 3: Hjälp Turtle att hitta till sitt hus. Ringa in buggen. Skriv rätt kod på skrivraden.

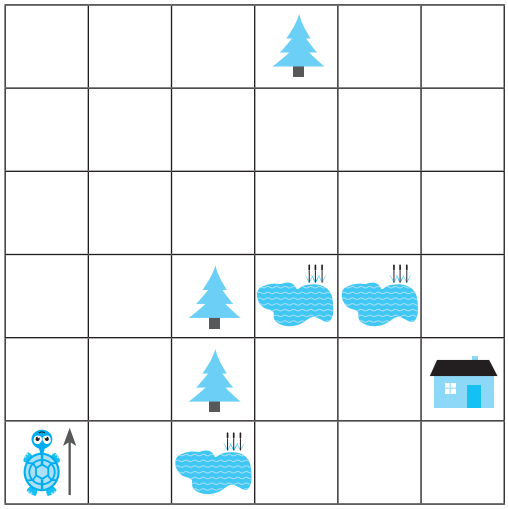
STRATEGI:

Börja med att läsa igenom **alla** rader med kod.
Ringa in buggen.
Skriv in rätt kod.



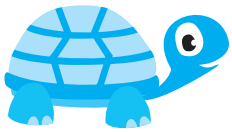
Programmerarens kod:	Din kod:
1 Gå två steg framåt	_____
2 Sväng höger	_____
3 Gå fem steg framåt	_____
4 Sväng höger	_____
5 Gå två steg framåt	_____

UPPDRAG 4: Hjälp Turtle att hitta till sitt hus. Läs igenom all kod. Ringa in buggen och skriv rätt kod på den tomma skrivraden.



Programmerarens kod:	Din kod:
1 Gå tre steg framåt	_____
2 Sväng höger	_____
3 Gå två steg framåt	_____
4 Sväng höger	_____
5 Gå två steg framåt	_____

När du läste igenom koden och ändrade buggen, så blev koden en exakt steg-för-steg-beskrivning. Du har nu skapat en **algoritm** *till datorprogrammet.*

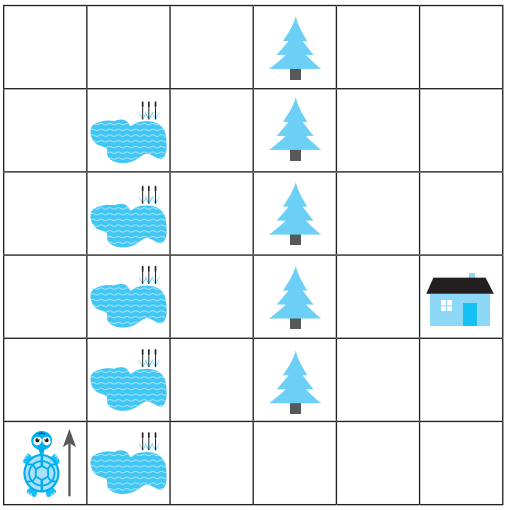


Skriva kod och hitta buggar

LEKTION 1

UPPDRAG 5: Turtle har varit och badat vid sjön och ska nu gå hem för att äta middag. Kontrollera koden så att Turtle hittar hem till sitt hus. Ändra kod när det behövs.

Programmerarens kod:	Din kod:
1 Gå fem steg framåt	_____
2 Sväng vänster	_____
3 Gå två steg framåt	_____
4 Sväng höger	_____
5 Gå tre steg framåt	_____
6 Sväng vänster	_____
7 Gå tre steg framåt	_____
8 Sväng höger	_____
9 Gå ett steg framåt	_____

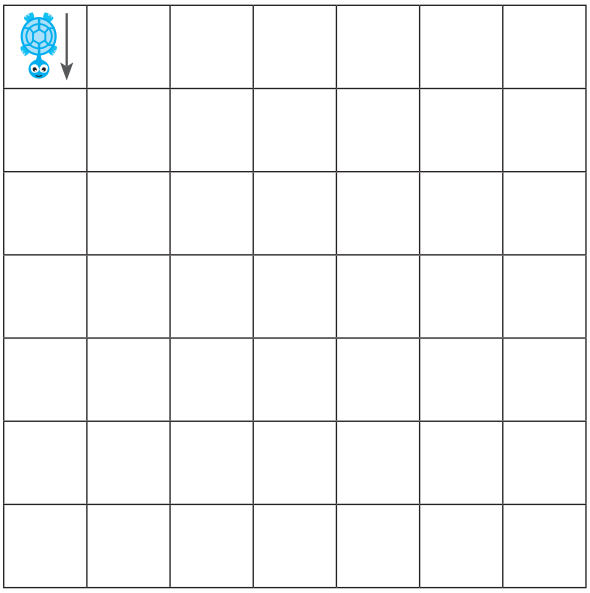


UPPDRAG 6: Skriv en egen algoritm med buggar.

Skriv minst fem rader med kod på skrivraderna här nedanför, en så kallad algoritm.
Din kod ska innehålla två buggar, som inte får komma efter varandra.
Rita in hinder och markera målet.
Be en klasskamrat att hitta dina buggar och skriva in rätt kod.

INSTRUKTIONER TILL DIN KLASSKAMRAT: Lös uppdraget. Läs igenom koden, ringa in buggarna och skriv in rätt kod på skrivraden.

Din kod:	Klasskamratens kod:
1 _____	_____
2 _____	_____
3 _____	_____
4 _____	_____
5 _____	_____
6 _____	_____
7 _____	_____



DISKUTERA MED EN KLASSKAMRAT

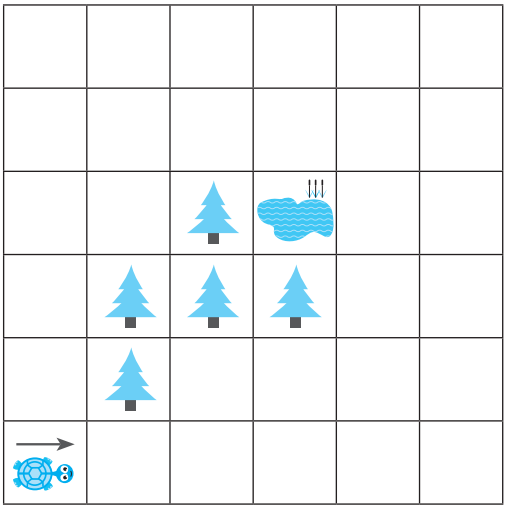
- Vad lärde ni er av uppgiften?
- Varför är det viktigt att hitta och ändra buggar?
- Har ni varit med om buggar förut? Vad hände då?

Lgr11: Hur algoritmer kan skapas och användas vid programmering i visuella programmeringsmiljöer.

Skriva kod och hitta buggar

LEKTION 1

UPPDRAG 7: Turtle är törstig och behöver hjälp att hitta till sjön. Vilken algoritm är rätt om din uppgift är att hitta den kortaste vägen för Turtle? Ringa in det rätta alternativet.



ALTERNATIV 1

```
1  Sväng vänster
2  Gå fyra steg framåt
3  Sväng höger
4  Gå fyra steg framåt
5  Sväng höger
6  Gå ett steg framåt
7  Sväng höger
8  Gå ett steg framåt
```

ALTERNATIV 2

```
1  Gå fyra steg framåt
2  Sväng vänster
3  Gå tre steg framåt
4  Sväng vänster
5  Gå ett steg framåt.
```

ALTERNATIV 3

```
1  Sväng vänster
2  Gå fyra steg framåt
3  Sväng vänster
4  Gå fyra steg framåt
5  Sväng höger
6  Gå ett steg framåt
7  Sväng höger
8  Gå ett steg framåt.
```

ALTERNATIV 4

```
1  Gå fem steg framåt
2  Sväng vänster
3  Gå tre steg framåt
4  Sväng vänster
5  Gå två steg framåt.
```

FACIT översikt Skriva kod och hitta buggar

LEKTION 1

En dator styrs med instruktioner som kallas **algoritmer.** Algoritmerna består av exakta steg-för-steg-instruktioner som kallas för **kod.** Varje kod betyder något för datorn. Sväng höger betyder till exempel att figuren står på sin plats och vrider sig 90° till höger!

Hej! Jag heter Turtle. Hjälp mig med de olika uppgifterna genom att lösa uppdragen!

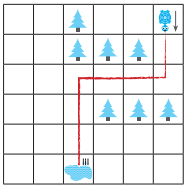
UPPDRAG 1: Åt vilket håll ska sköldpaddan Turtle svänga för att komma fram till sjön?

STRATEGI

Börja med att läsa igenom **alla** rader med kod.
Fyll i den rätta koden på den tomma skrivraden.

Skriv H för sväng höger och V för sväng vänster.

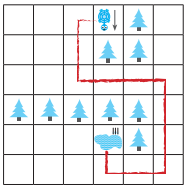
1 Gå två steg framåt
2 Sväng H
3 Gå tre steg framåt
4 Sväng V
5 Gå tre steg framåt



UPPDRAG 2: Åt vilket håll ska Turtle svänga för att komma fram till sjön?

Skriv H för sväng höger och V för sväng vänster.

1 Sväng H
2 Gå ett steg framåt
3 Sväng V
4 Gå två steg framåt
5 Sväng V
6 Gå tre steg framåt
7 Sväng H
8 Gå tre steg framåt
9 Sväng H
10 Gå två steg framåt
11 Sväng H
12 Gå ett steg framåt



Uppdrag 1 och 2

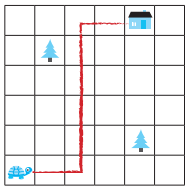
När datorn inte kan läsa koden så fungerar inte datorprogrammet eller spelet.
Ett fel i en kod kallas för **bugg.** För att fixa buggen behöver programmeraren leta i koden och hitta den. När datorprogrammet fungerar är koden rätt och buggen borta.

UPPDRAG 3: Hjälp Turtle att hitta till sitt hus. Ringa in buggen. Skriv rätt kod på skrivraden.

STRATEGI:

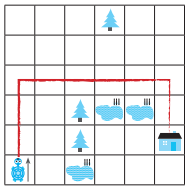
Börja med att läsa igenom **alla** rader med kod.
Ringa in buggen.
Skriv in rätt kod.

Programmerarens kod:	Din kod:
1 Gå två steg framåt	
2 Sväng höger	VÄNSTER
3 Gå fem steg framåt	
4 Sväng höger	
5 Gå två steg framåt	



UPPDRAG 4: Hjälp Turtle att hitta till sitt hus. Läs igenom all kod. Ringa in buggen och skriv rätt kod på den tomma skrivraden.

Programmerarens kod:	Din kod:
1 Gå tre steg framåt	
2 Sväng höger	
3 Gå två steg framåt	FEM
4 Sväng vänster	
5 Gå två steg framåt	

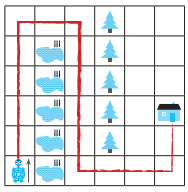


När du läste igenom koden och ändrade buggen, så blev koden en exakt steg-för-steg-beskrivning. Du har nu skapat en **algoritm** till datorprogrammet.

Uppdrag 3 och 4

UPPDRAG 5: Turtle har varit och badat vid sjön och ska nu gå hem för att äta middag. Kontrollera koden så att Turtle hittar hem till sitt hus. Ändra kod när det behövs.

Programmerarens kod:	Din kod:
1 Gå fem steg framåt	
2 Sväng vänster	HÖGER
3 Gå två steg framåt	
4 Sväng höger	
5 Gå tre steg framåt	FEM
6 Sväng vänster	
7 Gå tre steg framåt	
8 Sväng höger	VÄNSTER
9 Gå ett steg framåt	TVÅ

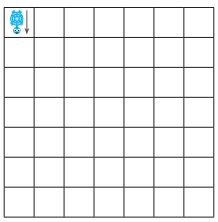


UPPDRAG 6: Skriv en egen algoritm med buggar.

Skriv minst fem rader med kod på skrivraderna här nedanför, en så kallad algoritm.
Din kod ska innehålla två buggar, som inte får komma efter varandra.
Rita in hinder och markera målet.
Be en klasskamrat att hitta dina buggar och skriva in rätt kod.

INSTRUKTIONER TILL DIN KLASSKAMRAT: Lös uppdraget. Läs igenom koden, ringa in buggarna och skriv in rätt kod på skrivraden.

Din kod:	Klasskamratens kod:
1	
2	
3	
4	
5	
6	
7	



DISKUTERA MED EN KLASSKAMRAT

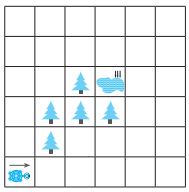
- Vad lärde ni er av uppgiften?
- Varför är det viktigt att hitta och ändra buggar?
- Har ni varit med om buggar förut? Vad hände då?

Uppdrag 5 och 6

UPPDRAG 7: Turtle är törstig och behöver hjälp att hitta till sjön. Vilken algoritm är rätt om din uppgift är att hitta den kortaste vägen för Turtle? Ringa in det rätta alternativet.

ALTERNATIV 1

1 Sväng vänster
2 Gå fyra steg framåt
3 Sväng höger
4 Gå fyra steg framåt
5 Sväng höger
6 Gå ett steg framåt
7 Sväng höger
8 Gå ett steg framåt



ALTERNATIV 2

1 Gå fyra steg framåt
2 Sväng vänster
3 Gå tre steg framåt
4 Sväng vänster
5 Gå ett steg framåt.

ALTERNATIV 4

1 Sväng vänster
2 Gå fyra steg framåt
3 Sväng vänster
4 Gå fyra steg framåt
5 Sväng höger
6 Gå ett steg framåt
7 Sväng höger
8 Gå ett steg framåt.

ALTERNATIV 3

1 Gå fem steg framåt
2 Sväng vänster
3 Gå tre steg framåt
4 Sväng vänster
5 Gå två steg framåt.

Uppdrag 7

Skriva kod och hitta buggar — FACIT — LEKTION 1

En dator styrs med instruktioner som kallas **algoritmer.** Algoritmerna består av exakta steg-för-steg-instruktioner som kallas för **kod.** Varje kod betyder något för datorn. Sväng höger betyder till exempel att figuren står på sin plats och vrider sig 90° till höger!

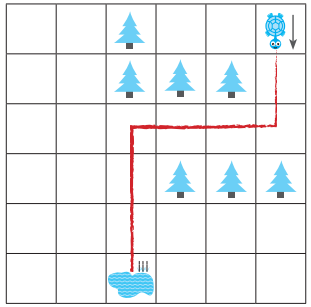
UPPDRAG 1: Åt vilket håll ska sköldpaddan Turtle svänga för att komma fram till sjön?

STRATEGI

Börja med att läsa igenom **alla** rader med kod.
Fyll i den rätta koden på den tomma skrivraden.

Skriv H för sväng höger och V för sväng vänster.

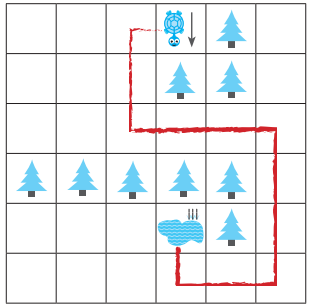
1 Gå två steg framåt
2 Sväng H
3 Gå tre steg framåt
4 Sväng V
5 Gå tre steg framåt



UPPDRAG 2: Åt vilket håll ska Turtle svänga för att komma fram till sjön?

Skriv H för sväng höger och V för sväng vänster.

1 Sväng H
2 Gå ett steg framåt
3 Sväng V
4 Gå två steg framåt
5 Sväng V
6 Gå tre steg framåt
7 Sväng H
8 Gå tre steg framåt
9 Sväng H
10 Gå två steg framåt
11 Sväng H
12 Gå ett steg framåt



Skriva kod och hitta buggar — FACIT — LEKTION 1

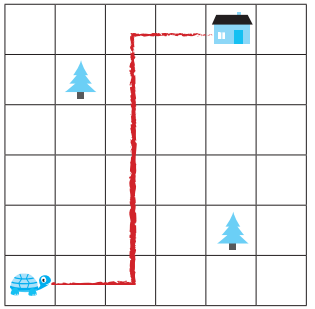
När datorn inte kan läsa koden så fungerar inte datorprogrammet eller spelet.
Ett fel i en kod kallas för **bugg.** För att fixa buggen behöver programmeraren leta i koden och hitta den. När datorprogrammet fungerar är koden rätt och buggen borta.

UPPDRAG 3: Hjälp Turtle att hitta till sitt hus. Ringa in buggen. Skriv rätt kod på skrivraden.

STRATEGI:

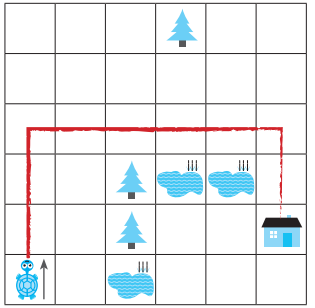
Börja med att läsa igenom **alla** rader med kod.
Ringa in buggen.
Skriv in rätt kod.

Programmerarens kod:	Din kod:
1 Gå två steg framåt	
2 Sväng höger	Vänster
3 Gå fem steg framåt	
4 Sväng höger	
5 Gå två steg framåt	



UPPDRAG 4: Hjälp Turtle att hitta till sitt hus. Läs igenom all kod. Ringa in buggen och skriv rätt kod på den tomma skrivraden.

Programmerarens kod:	Din kod:
1 Gå tre steg framåt	
2 Sväng höger	
3 Gå två steg framåt	Fem
4 Sväng vänster	
5 Gå två steg framåt	

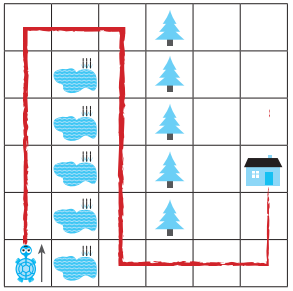


När du läste igenom koden och ändrade buggen, så blev koden en exakt steg-för-steg-beskrivning. Du har nu skapat en **algoritm** *till datorprogrammet.*

Skriva kod och hitta buggar — FACIT — LEKTION 1

UPPDRAG 5: Turtle har varit och badat vid sjön och ska nu gå hem för att äta middag. Kontrollera koden så att Turtle hittar hem till sitt hus. Ändra kod när det behövs.

Programmerarens kod:	Din kod:
1 Gå fem steg framåt	
2 Sväng vänster	Höger
3 Gå två steg framåt	
4 Sväng höger	
5 Gå tre steg framåt	Fem
6 Sväng vänster	
7 Gå tre steg framåt	
8 Sväng höger	Vänster
9 Gå ett steg framåt	Två

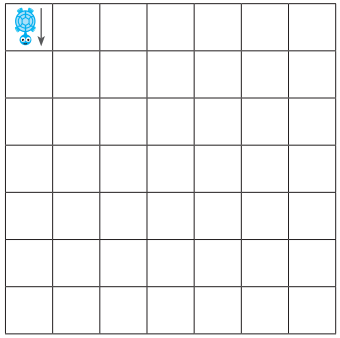


UPPDRAG 6: Skriv en egen algoritm med buggar.

Skriv minst fem rader med kod på skrivraderna här nedanför, en så kallad algoritm.
Din kod ska innehålla två buggar, som inte får komma efter varandra.
Rita in hinder och markera målet.
Be en klasskamrat att hitta dina buggar och skriva in rätt kod.

INSTRUKTIONER TILL DIN KLASSKAMRAT: Lös uppdraget. Läs igenom koden, ringa in buggarna och skriv in rätt kod på skrivraden.

Din kod:	Klasskamratens kod:
1	
2	
3	
4	
5	
6	
7	



DISKUTERA MED EN KLASSKAMRAT

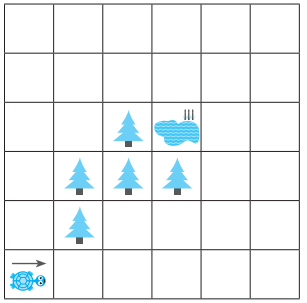
- Vad lärde ni er av uppgiften?
- Varför är det viktigt att hitta och ändra buggar?
- Har ni varit med om buggar förut? Vad hände då?

Skriva kod och hitta buggar — FACIT — LEKTION 1

UPPDRAG 7: Turtle är törstig och behöver hjälp att hitta till sjön. Vilken algoritm är rätt om din uppgift är att hitta den kortaste vägen för Turtle? Ringa in det rätta alternativet.

ALTERNATIV 1

1 Sväng vänster
2 Gå fyra steg framåt
3 Sväng höger
4 Gå fyra steg framåt
5 Sväng höger
6 Gå ett steg framåt
7 Sväng höger
8 Gå ett steg framåt



ALTERNATIV 2

1 Gå fyra steg framåt
2 Sväng vänster
3 Gå tre steg framåt
4 Sväng vänster
5 Gå ett steg framåt.

ALTERNATIV 4

1 Sväng vänster
2 Gå fyra steg framåt
3 Sväng vänster
4 Gå fyra steg framåt
5 Sväng höger
6 Gå ett steg framåt
7 Sväng höger
8 Gå ett steg framåt.

ALTERNATIV 3

1 Gå fem steg framåt
2 Sväng vänster
3 Gå tre steg framåt
4 Sväng vänster
5 Gå två steg framåt.

Matte Direkt Borgen – komplettering © Sanoma Utbildning 2018. Får kopieras.
Lgr 11: Hur algoritmer kan skapas och användas vid programmering i visuella programmeringsmiljöer.