Extramaterial till Matematik Alfa

NIVÅ
TRÄNA
UTVECKLA

Programmering

LÄRARE

I den här uppgiften får du och dina elever en introduktion till programmering. Ni kommer att få testa blockprogrammering i språket Blockly som till viss del liknar upplägget i Scratch.

Även de elever som har testat programmering tidigare kan dra nytta av att göra uppgiften eftersom de får träna på en del begrepp som har med programmering att göra samt testa på parprogrammering, felsökning och problemlösning.

Elevuppgifterna finns i två nivåer, Träna och Utveckla. Båda uppgifterna inleds på samma sätt med Del 1 och 2. Det som skiljer dem åt är ”Del 3” som ingår i Utveckla men inte i Träna. Del 3 handlar om villkor (if-satser). De elever som tycker att de första nivåerna är utmanande nog, kan man stoppa efter del 2.

Förutom elevuppgifterna finns även en ”lathund” där de viktigaste funktionerna i Blockly och uppgifterna på Code.org finns beskrivna.

SYFTE

Syftet med övningen är att eleven ska

- utveckla datalogiskt tänkande.
- träna sig i att lösa problem.
- bekanta sig med ett digitalt hjälpmedel.
- få erfarenhet av blockprogrammering
- lära sig begreppet loop
- få erfarenhet av att felsöka/debugga ett program
- få erfarenhet av parprogrammering

I UTVECKLA även:

- få erfarenhet av villkor/if-satser

TIDSÅTGÅNG

En lektion à 60 min.

KOSTNAD

Gratis

UTRUSTNING

Datorer eller lärplattor och webbsidan Code.org https://code.org/

REDOVISNING

Eleverna diskuterar med varandra under lektionen. Svaren på frågorna kan besvaras i helklass eller lämnas in till läraren.

FALLGROPAR

En del elever tycker det är svårt att komma igång. Ge dem lite tid och låt eleverna hjälpa varandra.

En del elever har en tendens att hasta igenom nivåerna för att hinna klart först. Var tydlig med att de ska läsa instruktionerna noga och att de ska stanna upp, till exempel efter nivå 5, och svara på frågor och reflektera över vad de gjort och lärt sig.

I filmerna får eleverna tips om nya block och hur de kan användas. Om eleverna hoppar över filmerna, kan det bli svårt med nästa avsnitt.

När man programmerar använder man sig oftast av de engelska uttrycken. På sidan code.org har man översatt uttrycken till svenska. Det kan göra eleverna något förvirrade.

På den här sidan finns en sammanställning av de vanligaste uttrycken på svenska och engelska samt deras betydelse: http://volante.se/wp-content/uploads/2015/09/Hej_Ruby_ordlista.pdf

PEDAGOGISKA TIPS

Testa gärna verktyget själv först. Då får du en föraning om vilka eventuella problem eleverna kommer att stöta på. Det tar mindre än en 30 minuter att ta sig igenom alla banor om man ser alla filmer och om man inte gör några fel. För eleverna, som även ska stanna upp och reflektera över vad de gjort, tar övningen ca 1h.

Samla gärna klassen efter genomförd uppgift och repetera begrepp och diskutera deras erfarenheter.

Uppgiften går att byggas ut. På webbsidan Code.org finns flera andra utmaningar av varierande svårighetsgrad.

Skapa konto eller inte?

Fördelen med att ha ett konto på Code.org, är att eleverna kan spara sina arbeten.

Kontrollera vad ni har för policy på skolan och påminn eleverna om att aldrig skapa konton utan att ha ett godkännande från sin lärare och/eller vårdnadshavare.

FÖRMÅGOR

- formulera och lösa problem med hjälp av matematik samt värdera valda strategier och metoder,
- använda och analysera matematiska begrepp och samband mellan begrepp,
- använda matematikens uttrycksformer för att samtala om, argumentera och redogöra för frågeställningar, beräkningar och slutsatser.

CENTRALT INNEHÅLL

- Det binära talsystemet och hur det kan tillämpas i digital teknik samt talsystem som använts i några kulturer genom historien, till exempel den babyloniska.
- Hur algoritmer kan skapas och användas vid programmering. Programmering i visuella programmeringsmiljöer.
- Strategier för matematisk problemlösning i vardagliga situationer.

ELEVUPPGIFTER MED KOMMENTARER OCH FACIT

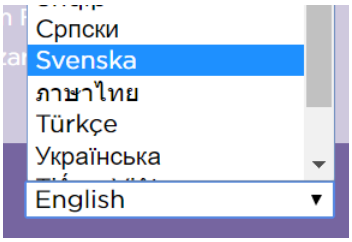
Här följer facit och kommentarer till elevuppgifterna. Ledtrådar till uppgifterna finns även i ”Lathund Programmering”. Ett rent facit, utan kommentarer, finns sist i detta dokument.

NIVÅ
TRÄNA
UTVECKLA

DEL 1: Börja med blockprogrammering

1. Gå in på code.org och välj ”svenska” om du får en fråga om vilket språk du vill använda.

   Om frågan om språk inte kommer upp och du behöver ändra språk, kan du ändra till svenska i menyn som finns längst ned på sidan.



2. Välj ”Elever - utforska alla våra tutorials”

3. Välj ”Klassisk labyrint”

4. Titta på filmen innan ni sätter igång med nivå 1. Där visar och beskriver de grunderna i Blockly. Det dyker upp korta filmer då och då som beskriver kommande moment. Man behöver inte förstå allt från filmen för att kunna sätta igång med nästa nivå.

I ”Lathund Programmering” finns de viktigaste funktionerna i Blockly och uppgifterna på Code.org beskrivna. Där finns även tips och ledtrådar till några av uppgifterna.

Vi rekommenderar att ni tittar på den första filmen tillsammans.

Filmen är på engelska men är textad på svenska.

Börja gärna med att fråga om det var något eller någon de kände igen i filmen.

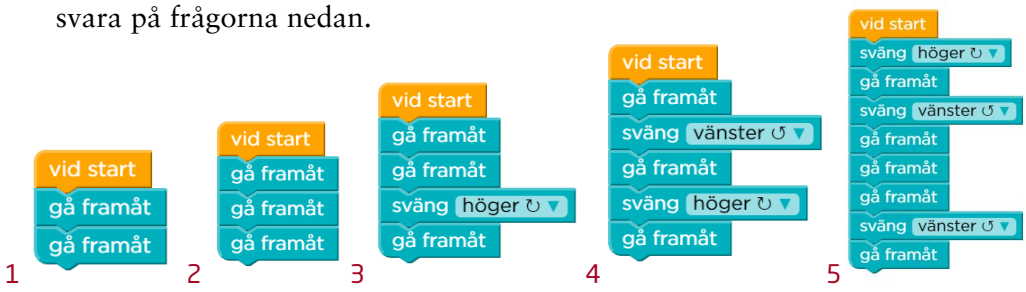
Några kända ansikten, Bill Gates och Mark Zuckerberg, dyker upp i filmen, känner eleverna igen dem? Vad är de kända för? Bill Gates är bland annat medgrundare till Microsoft och Mark Zuckerberg är medgrundare till Facebook.

Eleverna känner säkert igen figurerna från spelet "Angry Bird".

Om ni har möjlighet att visa filmen på storskärm inför klassen, kan ni med fördel även göra den första uppgiften, nivå 1, tillsammans. Visa de olika delarna för eleverna: labyrinten till vänster, verktygslådan och arbetsytan samt visa hur man lägger till/tar bort block och testar sitt program. Använd gärna "Lathund Programmering" som underlag.

5. Nu är det dags att köra igång med nivå 1–5. Överst kan ni se vilken nivå ni är på:

Observera att du inte ska fortsätta till nivå 6, utan pausa efter nivå 5 och svara på frågorna nedan.



6. Sammanfatta/Diskutera med en kompis:

a) Hur gick det? Vilka fel gjorde du? Varför?

b) Vilka fel gjorde dina klasskompisar, tror/märkte du?

c) Vilka fallgropar finns?

d) Varför är det så viktigt att vara tydlig? T ex med rätt antal steg, kommandon i rätt ordning eller att inte skriva vänster när du menar höger?

Datorn kan inte "läsa mellan raderna". Den kan inte tänka själv och förstå vad du menar om du inte ger korrekta instruktioner. Instruktionerna måste vara korrekta (rätt stavade och rätt ord) samt komma i rätt ordning för att datorn ska förstå och kunna följa dem.

NIVÅ
TRÄNA
UTVECKLA

DEL 2: Parprogrammering, loopar och felsökning/ debugging

Många programmerare använder sig av så kallad **parprogrammering** (Pair programming).

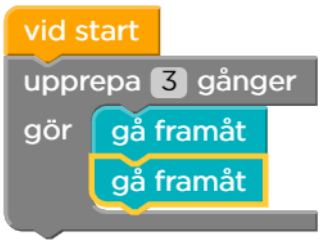
Då jobbar två programmerare tillsammans vid en gemensam dator. Den ena skriver kod medan den andra granskar varje kodrad när den matas in. De två programmerarna växlar ofta mellan rollerna.

Använd tekniken för parprogrammering för kommande uppgifter:

1. Nu ska ni få lära er mer om loopar. Titta på filmen där Mark Zuckerberg berättar om loopar innan du sätter igång med nivå 6 där du får testa på att använda loopar.

2. Gör nivå 6–9

   Ledtråd nivå 9
   Man kan stoppa in flera block inuti looparna även om det bara ser ut som det får plats ett.



6

7

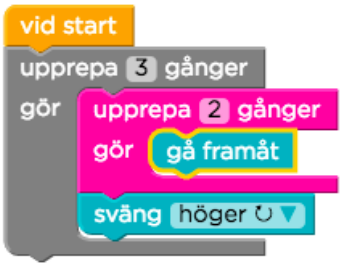
8

9

3. Om ni kör fast – felsök!

   Felsökning heter debugging på engelska. Det kan ni göra genom att t ex:
   Dela upp koden i mindre delar och testa.
   Byt ut delar och testa igen.

   Medan ni kör programmet kan ni se var i programmet ni befinner er genom att ett aktivt block blir gulmarkerat. Om fågeln stannar eller går åt fel håll kan ni på så vis klura ut vilket block som är fel.



4. Diskutera och/eller skriv ned svaren och lämna in:

   a) Vad är en Loop?

   En loop är när något upprepas.

   b) Varför använder man loopar?

   Man använder loopar för att slippa skriva onödigt mycket kod och för att programmet inte ska blir för stort och svårt att överblicka. Om någon del av ett program upprepas flera gånger kan man använda en loop istället. Exempel: Istället för att skriva ”gå framåt, gå framåt, gå framåt” kan man skriva ”Upprepa 3 gånger: Gå framåt”.

c) Varför använder man sig ofta av parprogrammering när man programmerar, tror ni? Vad var era erfarenheter?

Parprogrammering används till exempel för att det är lättare att lösa problem tillsammans med någon och för att enklare hitta fel.

5. Titta på filmen ”Upprepa tills-block” (Repeat until) och gör sedan nivå 10 och 11.

Ledtråd nivå 11
Man kan stoppa in flera block inuti ”Upprepa-tills”, även om det bara ser ut som om det får plats ett.

10

11

6. Diskutera och/eller skriv ned svaren och lämna in:

a) Vad var skillnaden mellan blocken ”upprepa _ (antal) gånger” och ”upprepa tills”?

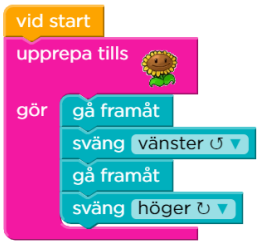
b) När är de olika blocken användbara?

a) Om man använder loopen ”upprepa_(antal) gånger” utförs loopen ett visst antal gånger. Använder man istället ”upprepa tills” utförs loopen så många gånger som behövs för att villkoret ska uppfyllas.

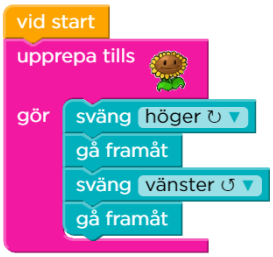
b) Om man vet eller vill styra exakt hur många gånger något ska utföras för att villkoret ska uppfyllas är ”upprepa_(antal) gånger” bäst att använda. Om antalet gånger inte spelar någon roll, bara man når målet, är ”upprepa tills” bättre att använda.

7. Gör nivå 12 och 13.

Om ni kör fast på nivå 13 kan ni börja med att försöka utan loopen först och se så att zombien går rätt i början.



12



13

NIVÅ
UTVECKLA

DEL 3: Villkor (If-satser)

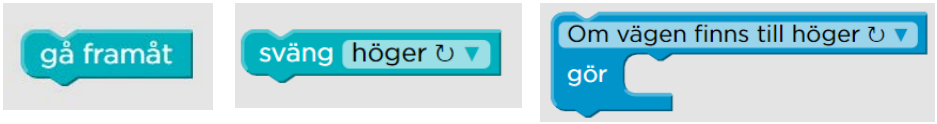
1. Titta på filmen ”Om”-block där Bill Gates berättar om If-satser.
2. Gör nivå 14-17

Ledtrådar till nivå 15-17

Nivå 15

Ledtråd 1
Börja med ”upprepa tills”-block

Ledtråd 2
Du behöver även välja följande block:



Ledtråd 3
De två första blocken ska placeras så här:

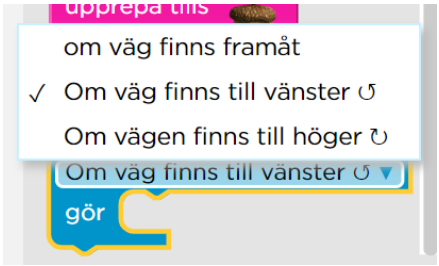
NIvå 16

Ledtråd 1
Börja med ”upprepa tills”-block

Ledtråd 2
Du behöver även välja följande block:

Nivå 17

Om-blocket går att förändra genom att klicka på den lilla blå pilen.



Facit till nivå 14-17

14

15

16

17

3. Titta på filmen ”Om/annars”-block (If/else).
4. Gör nivå 18–20.

Ledtrådar:

Nivå 18

Block som ska användas:

Nivå 20

Läs innehållet i de grå blocken och fundera på vad som ska göras:

Om det finns en väg framåt, vad ska ekorren göra då?

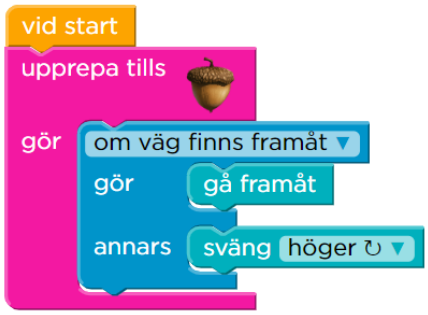
Om det finns en väg till höger, vad ska ekorren göra då?

Vad ska ekorren göra om det INTE finns en väg till höger?

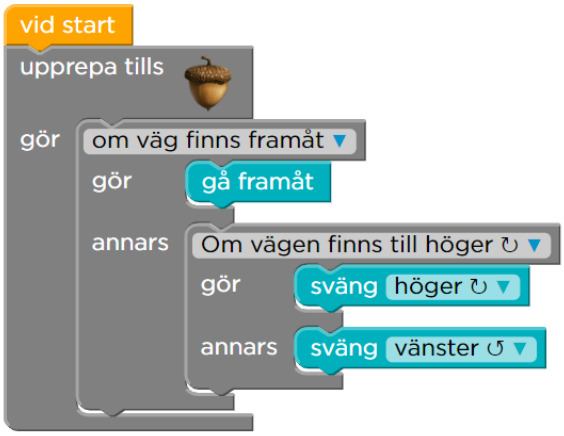
Facit:

18

19



20



5. Diskutera och/eller skriv ned svaren och lämna in:

a) Hur gick det?

b) Varför använder man ”Om-block” (If-satser)?

För att kunna kontrollera vad som ska hända i programmet om ett villkor är uppfyllt eller inte.

c) Kan du komma på något exempel från din egen vardag när du använder ”Om”-block?

If…… then…… else…….

(Om….. så….. annars…….)

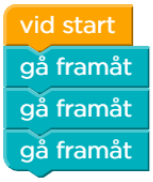
Exempel:
Om klockan ringer, så gå upp, annars fortsätt sova
Om det regnar, så ta på dig gummistövlar, annars sneakers

d) Kan du komma på något, i sitt hem eller i skolan, som är programmerat **och** som använder ”Om”-block?

Exempel: Tvättmaskin, kylskåp, miniräknare, TV, fjärrkontroll, hemlarm.

Facit

1


2


3


4


5


6

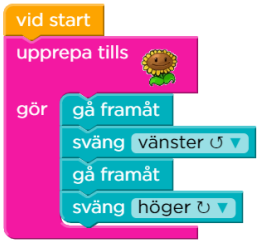

7

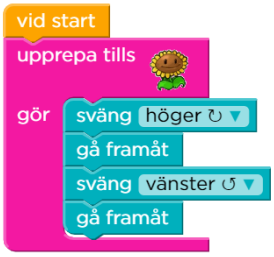

8


9


10


11


12


13


14


15

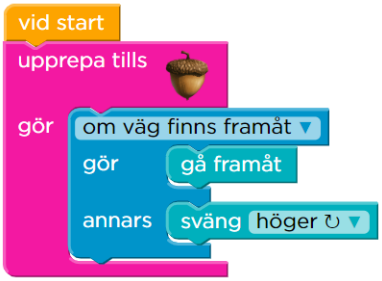

16


17

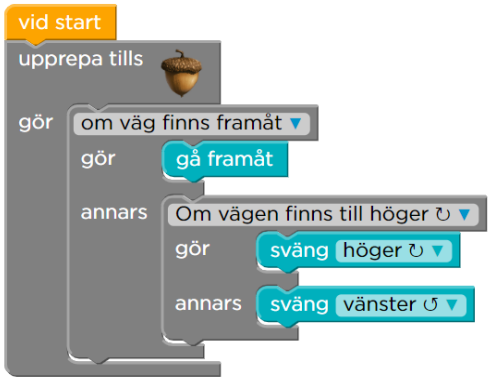

18


vid start
upprepa tills
gör
om väg finns framåt
gör
gå framåt
annars
sväng vänster

19


vid start
upprepa tills
gör
om väg finns framåt
gör
gå framåt
annars
sväng höger

20


vid start
upprepa tills
gör
om väg finns framåt
gör
gå framåt
annars
Om vägen finns till höger
gör
sväng höger
annars
sväng vänster