Co-funded by the Erasmus+ Programme of the European Union

Modul 2

Allmän introduktion till datalogiskt tänkande

En grundläggande modul för alla lärare

Författare: Radboud University (Nederländerna)
Maria Kallia,
Sjaak Smetsers,
Erik Barendsen,
Christos Chytas

Granskare:
Arnold Pears (KTH),
Valentina Dagienė (VU)

Externa granskare:
Piret Luik (Estland),
Renate Motschnig (Österrike)

Pilot:
Ankara University (Turkiet), KTH Royal Institute of Technology (Sverige), Radboud University (Nederländerna), University of Paderborn (Tyskland), Vienna University of Technology (Österrike)

Grafisk design:
Vaidotas Kinčius (Litauen)

Modul *2* bygger på arbetet i projektet ”Future Teachers Education: Computational Thinking and STEAM” (TeaEdu4CT). Samordnare: prof. Valentina Dagienė, Vilnius universitet, Litauen. Partner: Technische Universität Wien (Österrike), CARDET (Cypern), Tallinns universitet (Estland), Åbo universitet (Finland), Paderborns universitet (Tyskland), CESIE (Italien), Radboud University (Nederländerna), KTH Kungliga Tekniska högskolan (Sverige), Ankaras universitet (Turkiet). Projektet har samfinansierats genom Erasmus+-programmet KA2.

Innehåll

Allmän översikt och syfte

Syftet med denna modul är att ge blivande lärare en konkret förståelse av datalogiskt tänkande och praktisk kunskap om principerna för undervisning och lärande samt introducera dem till datalogiska verktyg som kan underlätta undervisningen och lärandet.

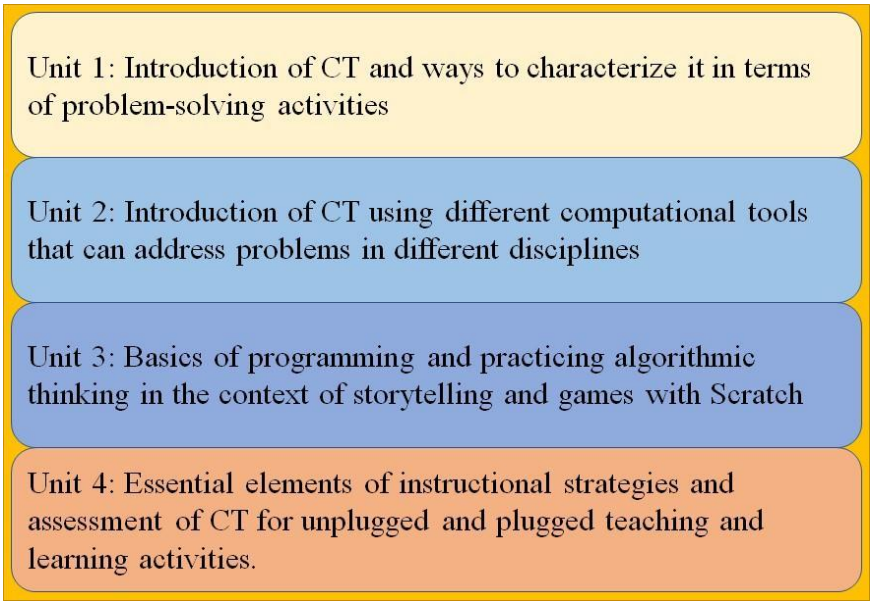
I den här modulen ska lärarstudenterna

- utforska begreppet datalogiskt tänkande och jämföra olika sätt att karakterisera det ur problemlösningssynpunkt,
- studera aspekter av samt metoder och verktyg för datalogiskt tänkande och förstå dess funktion inom olika vetenskapsgrenar,
- bekanta sig med programmeringens grunder och använda algoritmiskt tänkande inom ramen för historieberättande och spel,
- pröva på undervisning och lärande både med och utan dator samt få inblick i utformningen av undervisningsstrategier för datalogiskt tänkande.

I modulen betraktas datalogiskt tänkande som en ram för utveckling av färdigheter och kompetenser som lärare i olika ämnen kan använda sig av.

Modulens uppläggning

Modulen är uppdelad i fyra avsnitt: I avsnitt 1 introduceras begreppet datalogiskt tänkande, och olika sätt att karakterisera det ur problemlösningssynpunkt jämförs. I avsnitt 2 behandlas olika datalogiska verktyg (Ngrams, NetLogo, Excel) och hur dessa verktyg kan användas för att ta itu med problem inom olika vetenskapsgrenar (historia, biologi, geografi). Avsnitt 3 är inriktat på programmeringens grunder och på användningen av algoritmiskt tänkande inom ramen för historieberättande och spel med Scratch. Modulen avslutas med avsnitt 4, om undervisning och lärande både med och utan dator och viktiga delar av undervisningsstrategier och bedömning för datalogiskt tänkande.



Målgrupper och förutsättningar

Den här modulen är avsedd för lärarstudenter samt för fortbildning av lärare som är intresserade av datalogiskt tänkande. Modulen är utformad för närutbildning, men kan enkelt anpassas för distansundervisning.

Inga särskilda förutsättningar måste vara uppfyllda för att använda modulen. Det är bra om studenterna har en grundläggande kunskap om de verktyg som används och har slutfört den föregående modulen ”O1: Frameworks for the support of the modules: CT&STEM for future teacher education”.

Läranderesultat och bedömningsmetoder

En lärarstudent som har gått igenom hela modulen kan

- förklara begreppet datalogiskt tänkande och jämföra olika sätt att karakterisera det ur problemlösningssynpunkt,
- lösa problem med hjälp av det datalogiska tänkandets begrepp, metoder och verktyg och förklara dess funktion inom olika vetenskapsgrenar,
- konstruera enkla program i Scratch och tillämpa algoritmiskt tänkande inom ramen för historieberättande och spel,
- tillämpa undervisningsstrategier och lärandemoment för datalogiskt tänkande både med och utan dator samt beskriva de underliggande principerna.

Mer specifika lärandemål anges varje avsnitt.

Bedömningsstrategi

Uppgifterna i modulerna möjliggör formativ bedömning och feedback. Det finns en valfri slutlig (sammantagen) bedömning i ett separat dokument (se läranderesurser: modul för slutlig bedömning).

Modulplan och undervisningsstrategier

Modulen består av 4 avsnitt för närutbildning. Varje avsnitt omfattar flera lärandemoment som normalt inleds med ett uppvärmningsmoment och avslutas med ett reflekterande moment. Studenterna kommer i kontakt med olika undervisningsstrategier, exempelvis bakgrundsläsning av artiklar eller utvalda bokrecensioner, gruppdiskussioner, problemlösning och reflekterande moment.

Avsnitt och lärandemoment

Avsnitt 1: Introduktion till datalogiskt tänkande

Moment 1.1 Introduktion till datalogiskt tänkande

- Datalogiskt tänkande som problemlösning
- Element i datalogiskt tänkande
- Slutsats

Totalt: 2 timmar

Avsnitt 2: Verktyg för datalogiskt tänkande

Moment 2.1: Google Ngrams

Moment 2.2: Modellering och simulering med NetLogo

Moment 2.3: Microsoft Excel

Totalt: 8 timmar

Avsnitt 3: Datalogiskt tänkande inom programmering

Moment 3.1: Historieberättande i Scratch

Moment 3.2: Skapa ett labyrintspel

Totalt: 10 timmar

Avsnitt 4: Undervisnings- och lärandestrategier för datalogiskt tänkande

Moment 4.1: Databävern

Moment 4.2: Hitta rätt i en labyrint

Moment 4.3: Undervisningsstrategier

Totalt: 8 timmar

AVSNITT 1: Introduktion till datalogiskt tänkande

I det här avsnittet ska du utforska begreppet datalogiskt tänkande och jämföra olika sätt att karakterisera det ur problemlösningssynpunkt.

Bidrag till läranderesultaten

Läranderesultat
• Kunna beskriva datalogiskt tänkande som en förbindelselänk mellan ett ämnesområde och informationsteknik.
• Karakterisera datalogiskt tänkande ur problemlösningssynpunkt med steg och moment som leder till en funktionsduglig och genomförbar lösning.
• Känna igen element av datalogiskt tänkande i scenarier för studentaktiviteter.

Moment 1.1 Introduktion till datalogiskt tänkande

För att utnyttja datorernas fulla potential inom olika ämnen och aktiviteter krävs det mer digital kompetens än att kunna använda Word eller sociala medier. För att angripa ett ämne enligt informationstekniska principer krävs särskilda problemlösningsfärdigheter. Detta är datalogiskt tänkande.

Datalogiskt tänkande som problemlösning

Bakgrundsläsning

Läs dokumentet ”A brief introduction to computational thinking” i läranderesurserna. I denna text beskrivs tre huvudsteg i det datalogiska tänkandet: pilarna (1), (2) och (3) i diagrammet.

Diskussion i par

Diskutera de två klassrumsscenarierna nedan, som har hämtats från Yadav m.fl. (2018, s. 381). Vilka aktiviteter skulle du kalla datalogiskt tänkande? Hur står de i samband med stegen (1), (2) och (3) i material A?

Scenario 1:

Westwoods grundskola ska inleda nästa skolår med ett 1:1 iPad-initiativ. En lärare, Nowak, tänker låta andraklassarna använda sina paddor för väderprognoser (temperatur, nederbörd och vind) under en vecka. Varje elev ritar en bild av vädret som de tror att det kommer att bli. Sara, en av eleverna, ville också hålla koll på allas temperaturprognoser. Nowak startade ett kalkylark i Google där varje elev lade in sina temperaturprognoser. Nästa dag registrerade de det aktuella vädret på sina paddor med hjälp av Accuweathers app och lade in informationen i kalkylarket. Olivia ville också föra in den aktuella temperaturen i Saras kalkylark så att de kunde jämföra sina prognoser med det faktiska vädret. Efter en vecka visade de kalkylarket på den interaktiva skrivtavlan och jämförde den faktiska temperaturen med temperaturprognoserna.
Läraren visade hur man gör ett stapeldiagram över skillnaderna.

Scenario 2:

Alla andraklassare åker på utflykt! I skolbespisningen har personalen packat smörgåsar med jordnötssmör och sylt i likadana papperspåsar för alla, förutom Sara och Olivia som har jordnötsallergi. Lunchpåsarna märks med alla elevers namn och delas upp i 10 lådor med 10 påsar i varje låda. Påsarna placeras i lådorna i alfabetisk ordning enligt efternamn. Nowak vill kontrollera att Sara och Olivia verkligen får en lunch utan jordnötter. De hjälper honom att leta i lådorna. Olivia Velazquez vet att hennes lunch troligen är nära slutet, så hon tittar på den första lunchpåsen i varje låda tills hon hittar en där namnet börjar med en bokstav nära slutet av alfabetet. När hon hittar den låda där Jemal Summers lunchpåse är först tittar hon på den sista påsen i lådan. Det är Billy Wagners, så hon vet att hon är nära! Hon tittar på påsen precis bredvid Billys, och det är hennes. Som tur är har personalen i skolbespisningen kommit ihåg att ge henne en ostsmörgås och morötter.

Fl[illegible]ogiskt tänkande

De[illegible]blemlösningen kan beskrivas på följande övergripande sätt.

1. Dekontextualisering: omvandla ett problem eller en fråga på ett ämnesområde (”kontext”) till datalogiska termer.
2. Datalogisk problemlösning: utforma en genomförbar lösning. 3. (På nytt) sätta in i sitt sammanhang: flytta tillbaka lösningen till ämnesområdet.

Definitionerna av datalogiskt tänkande lägger varierande tonvikt på moment (1), (2) och (3). Selby och Woollard (2013, s. 5) beskriver exempelvis datalogiskt tänkande som en ofta produktinriktad aktivitet som är förknippad med, men inte är begränsad till, problemlösning. Det är en kognitiv process eller tankeprocess som återspeglar

- förmågan att tänka abstrakt,
- förmågan att göra en problemuppdelning,
- förmågan att tänka algoritmiskt,
- förmågan att utvärdera,
- förmågan att generalisera.

Även om det finns en viss överlappning kan vi i grova drag knyta dessa element till stegen i vår modell. De första två elementen är i huvudsak förknippade med steg (1): analysera mönster i ett problem eller en situation (abstraktion) och dela upp ett problem i mindre delproblem (problemuppdelning). Algoritmiskt tänkande används främst i steg (2), medan utvärderingen av en situation och undersökningen av hur den kan generaliseras knyter den datalogiska lösningen till ämnesområdet, vilket sker i steg (3). Detta kan sammanfattas i en tabell:

	(1)	(2)	(3)
Selby & Woollard (2013)	abstraktion problemuppdelning	algoritmiskt tänkande	utvärdering generalisering

Uppgift i par

Tre andra välkända operationaliseringar beskrivs i dokumentet "”B. Definitions of CT”". Utvidga ovanstående tabell med ytterligare tre rader. Kategorisera elementen i de tre kolumnerna. Känner du igen några inslag i klassrumsscenarierna 1 och 2?

Slutstas Allmän diskussion

Jämför dina resultat med klasskamraternas. Skapa tillsammans en kortfattad arbetsdefinition av datalogiskt tänkande i termer av steg och aktiviteter.

Läranderesurser

A. Brief introduction to CT

B. Definitions of CT

Referenser

Selby, C. & Woollard, J. (2013) Computational thinking: the developing definition, på internet: http://eprints.soton.ac.uk/356481, hämtat 10 februari 2021.

Yadav, A., Krist, C., Good, J., & Caeli, E. N. (2018). ”Computational thinking in elementary classrooms: measuring teacher understanding of computational ideas for teaching science”. *Computer Science Education*, 28(4), 371–400.

AVSNITT 2: Verktyg för datalogiskt tänkande

I detta avsnitt får du praktisk erfarenhet av datalogiskt tänkande med hjälp av enkla men kraftfulla digitala hjälpmedel för verklig problemlösning inom olika vetenskapsgrenar – ingen ytterligare programmering krävs just nu.

I avsnittet introduceras tre hjälpmedel, nämligen NetLogo, Microsoft Excel och Google Ngrams. Dessutom visas det hur dessa hjälpmedel kan användas för att integrera datalogiskt tänkande i olika skolämnen. Varje hjälpmedel och dess tillämpning inom olika vetenskapsgrenar beskrivs i detalj i respektive underavsnitt. Genom dessa aktiviteter bekantar du dig med aspekter av samt metoder och verktyg för datalogiskt tänkande och förstår dess funktion inom olika vetenskapsgrenar.

Bidrag till läranderesultaten

Läranderesultat
● Studera och tillämpa datalogiskt tänkande i praktiken genom en rad fallscenarier ● Klassificera problem som datalogiskt lösbara ● Använda datalogiska verktyg för att lösa problem ● Ta fram detaljerade steg-för-steg-lösningar på problem ● Fundera över, tolka och visualisera data ● Använda dataanalys för att öka förståelsen av naturliga system ● Utvärdera vilken typ av problem som kan lösas med hjälp av modellering och simulering ● Förstå och beskriva hur modellering och simulering kan användas för att lösa ett problem ● Förstå och beskriva hur modellering och simulering kan användas för att lösa ett problem ● Analysera data och identifiera mönster genom modellering och simulering. ● Samarbeta och kommunicera med andra för att lösa ett problem

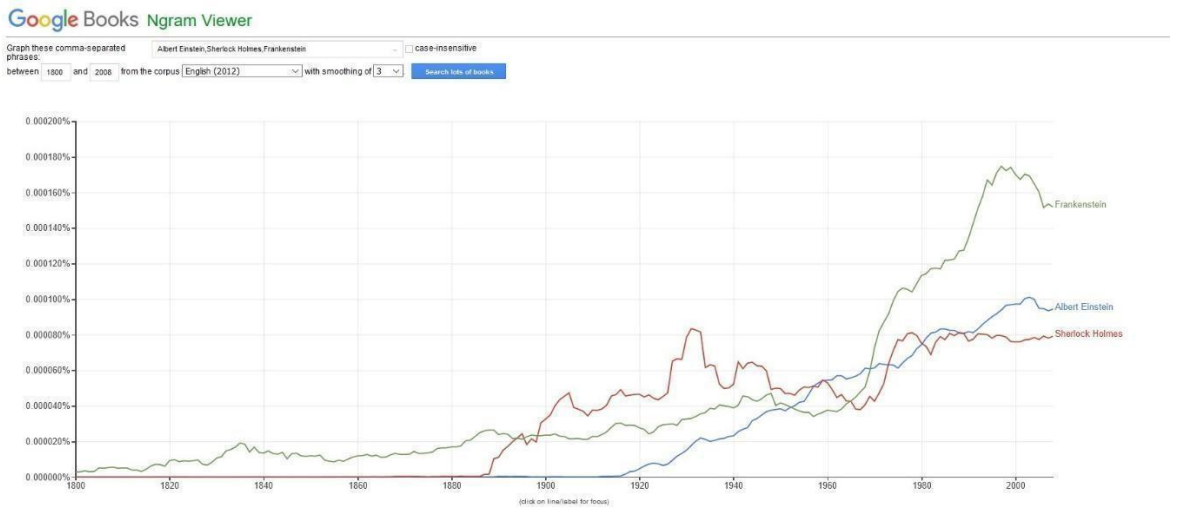
Moment 2.1: Google Ngrams

Totalt: 2–3 timmar

Google Ngram Viewer visar diagram över hur ofta ord förekommer i böcker samt användningen av termer eller fraser över tid. Verktyget använder litteraturkällor som trycktes mellan 1500 och 2008 på amerikansk engelska, brittisk engelska, franska, tyska, italienska, spanska, ryska, hebreiska och kinesiska. Därmed kan det användas för att utforska språkförändringar över tid. Förändrat språkbruk kan återspegla sociokulturella förändringar och därmed ge inblick i den historiska utvecklingen. I denna aktivitet ska vi använda Google Ngrams för att undersöka kulturella och sociala förändringar över tid som de återspeglas genom bruket av vissa termer i böcker.

U[illegible]diskussion

Innan du börjar ska du klicka på följande länk för att få tillgång till Google Ngram Viewer (https://books.google.com/ngrams). Du ser att verktyget redan föreslår ett exempel som visas i följande figur. I detta exempel ser vi resultaten för tre fraser, ”Albert Einstein”, ”Sherlock Holmes” och ”Frankenstein”. Verktyget visar hur ofta dessa fraser förekommer i böcker från perioden 1800–2008. Som synes har termen Frankenstein förekommit i litteraturen sedan början av 1800-talet, medan de övriga två termerna dyker upp senare i litteraturen.



Figur 1. Google Ngram, exempel för termerna Albert Einstein, Sherlock Holmes och Frankenstein.

Diskussion

Vilka andra trender lägger du märke till när det gäller hur ofta termerna i figur 1 förekommer?

F[illegible]beskrivning

Fr[illegible]n undersöka i denna aktivitet rör kulturella förändringar i USA under de senaste två hundra åren. Den sociokulturella teori vi ska tillämpa beskrivs i följande stycke, följt av vår fråga.

Enligt teorin om social förändring och mänsklig utveckling sker det en övergång från *Gemeinschaft* till *Gesellschaft* genom sociodemografiska förändringar. Detta begreppspar lanserades av Toennies 1887 och har använts för att studera samhällsförändringar. Gemeinschaft avser bindande, primära relationer som bygger på en känsla av gemenskap och karakteriseras av en bondemiljö, naturliga personliga relationer, enkel teknik, begränsad utbildning samt av nöd snarare än välstånd. Gesellschaft, däremot, avser ett system som

karakteriseras av egennytta, konkurrens och förhandlade uppgörelser samt en stadsmiljö, ett modernt samhälle med avancerad teknik och välstånd (källa: Younes och Reips, 2018, s. 1; Christenson, 1984, s. 160).

Under de senaste århundradena har urbaniseringstakten ökat drastiskt i hela världen. I denna fallstudie undersöker vi om denna förändring åtföljs av en övergång från *Gemeinschaft* till *Gesellschaft*. Mer bestämt ska vi undersöka följande fråga:

"Har USA gått från Gemeinschaft till Gesellschaft under de senaste två århundradena och åtföljs detta av en övergång från landsbygdssamhälle till stadssamhälle?"

För att svara på denna fråga ska vi undersöka hur de sociokulturella förändringarna återspeglas i bruket av ord i USA under de senaste två århundradena. Google Ngram Viewer är särskilt användbar för det, eftersom kulturella värderingar återspeglas i de ord/termer som används i skrift, vilket gör att vi kan påvisa långsiktiga förändringar av de kulturella värderingarna genom att undersöka hur ofta ord förekommer i böcker.

F[illegible]ktermer

För att kunna besvara vår fråga är att hitta lämpliga söktermer som är förknippade med begreppen Gemeinschaft och Gesellschaft och kan ge en bild av de kulturella värderingarna i USA. I just detta exempel är följande kriterier viktiga för att välja representativa söktermer: a. att termen förekommer ofta, och b. att den inte kan tolkas på så många olika sätt.

Diskussion

Varför är ovanstående kriterier viktiga?

*

Det är viktigt att välja termer som förekommer ofta så att diagrammet faktiskt visar de kulturella förändringarna över tid och eftersom ord som förekommer ofta är karakteristiska för landets kultur. I detta exemplet är det lika viktigt att välja ord som inte kan tolkas på så många olika sätt. Skälet är att ord med många betydelser kan användas i olika sammanhang som inte alltid har att göra med kulturella värderingar. I följande tabell anges ord som uppfyller ovanstående kriterier och återspeglar Gemeinschaft respektive Gesellschaft.

Tabell 1: Termer förknippade med Gemeinschaft och Gesellschaft

Gemeinschaft	förpliktad, plikt, ge, godhet, handla, handling
Gesellschaft	välja, beslut, få, förvärv, känna, känsla

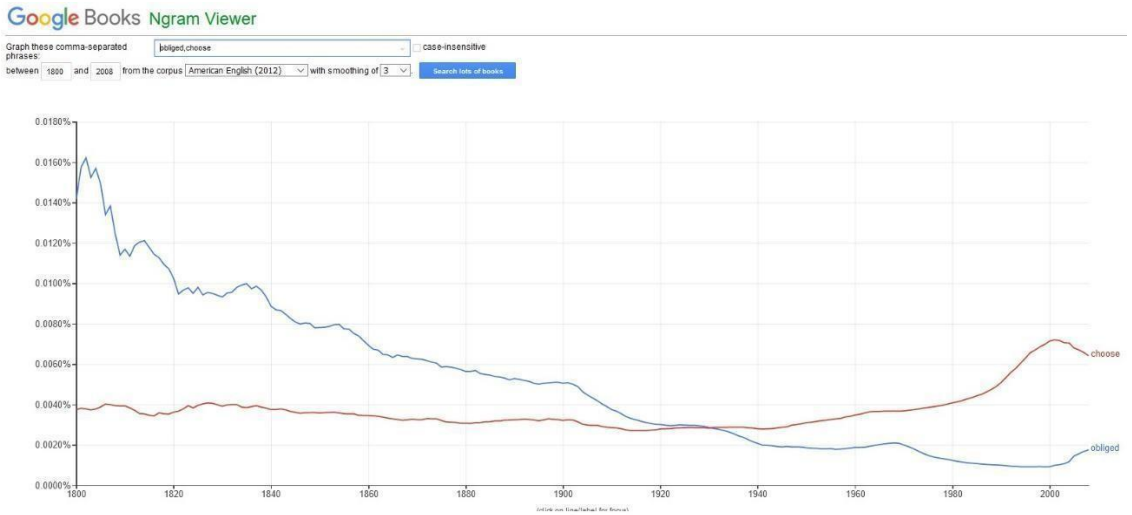
Diskussion

Hur kan vi använda ovanstående förteckning och Google Ngram Viewer för att besvara vår fråga?

Vi nämnde att Google Ngram kan användas för att visa hur ofta ord förekommer i böcker över tid. I vårt exempel innebär det att vi kan använda och jämföra termer som återspeglar motsatta attityder, egenskaper och särdrag och observera hur frekvensen varierar över tid.

Diskussion

I följande figur visas hur ofta orden förpliktad (förknippat med Gemeinschaft) och välja (förknippat med Gesellschaft) förekommer. Hur ofta förekommer de här termerna över tid?



Fler termer och tolkning

I d använda Google Ngrams för att jämföra fler termer som är förknippade med Gemeinschaft och Gesellschaft och observera hur frekvensen har förändrats under de senaste två hundra åren.

Gruppuppgift

Klicka på följande länk för att komma till Google Ngrams (https://books.google.com/ngrams).

1. Använd sökfältet (ta bort söktermer som redan finns i fältet) och skriv in följande två ord med komma mellan: Plikt, Beslut.
2. I rullgardinsmenyn ”from the corpus” väljer du amerikansk engelska (2012) och klickar sedan på knappen ”search lots of books”.
3. Hur ofta förekommer de här termerna över tid?
4. Undersök på samma sätt hur ofta följande ord förekommer (välj minst två):
    a. Ge och Få.
    b. Godhet och Förvärv.
    c. Handla och Känna.

d. Handling och Känsla.

5. Hur varierar frekvensen för dessa ord?

Bekräftar dina observationer att det har skett en övergång från Gemeinschaft till Gesellschaft?

*

Det visar sig verkligen att termer som förpliktad, plikt, ge, godhet, handla och handling, som återspeglar ett Gemeinschaft i landsbygdsmiljö, används allt mer sällan, i linje med teorin om social förändring och mänsklig utveckling. Under samma period har förekomsten av termer som välja, beslut, få, förvärv, känna och känsla, som alla återspeglar ett Gesellschaft i stadsmiljö, ökat.

G[illegible] (valfritt)

I d[illegible]ngick det att de kulturella kännetecknen för Gesellschaft (enligt relevanta ord i amerikanska böcker) ökade i antal, medan de kulturella kännetecknen för Gemeinschaft (enligt relevanta ord i amerikanska böcker) minskade i antal, i takt med att USA utvecklades i riktning mot Gesellschaft. Kan dessa relationer och trender generaliseras för andra delar av världen i linje med teorin om social förändring och mänsklig utveckling?

Gruppuppgift

Upprepa stegen i fas 3, men välj själv litteratur (t.ex. brittisk eller tysk) för att besvara samma fråga för det landet.

Obs: För tysk litteratur kan du använda följande termer (källa: Younes & Reips, 2018):

1. Versprechen och auswählen
2. Pflicht och Entscheidung
3. Geben och bekommen
4. Güte och Kauf
5. Handeln och spüren
6. Handlung och Emotion

Läranderesurser

A Gemeinschaft to Gesellschaft.pdf (valfritt)

B The changing psychology of culture from 1800 through 2000 (valfritt)

C The changing psychology of culture in German-speaking countries (valfritt)

Referenser

Christenson, J.A., (1984). ”Gemeinschaft and gesellschaft: Testing the spatial and communal hypotheses”. *Social Forces*, 63(1), s.160–168.

Greenfield, P.M., (2013). ”The changing psychology of culture from 1800 through 2000”. *Psychological Science*, 24(9), s.1722–1731.

Younes, N. och Reips, U.D., (2018). ”The changing psychology of culture in German-speaking countries: A Google Ngram study”. *International Journal of Psychology*, 53, s. 53–62.

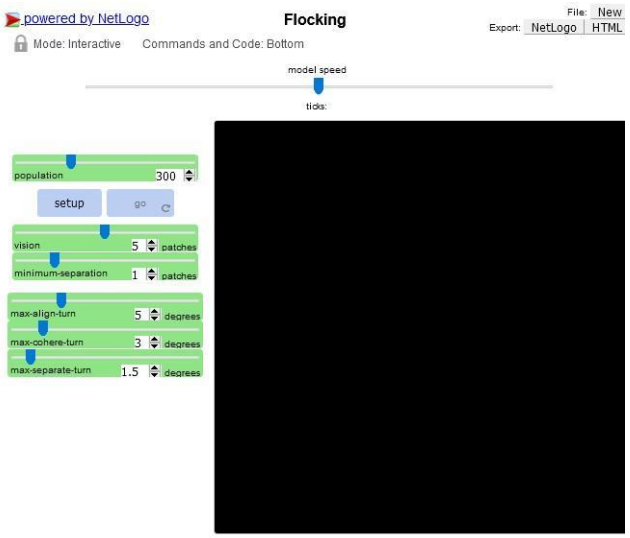
Moment 2.2: Modellering och simulering med NetLogo

Totalt: 2,5–3 timmar

NetLogo är en programmerbar modelleringsmiljö med flera agenter för simulering av naturliga och sociala fenomen och visning av deras utveckling över tid. Här kan du skapa en värld av rektanglar eller fält och parameterisera agenter (eller sköldpaddor) som förflyttar sig och interagerar med varandra och sin miljö. I detta moment ska vi använda NetLogo för en epidemiologisk fallstudie och göra modellsimuleringar för att undersöka hur stora och små förändringar kan påverka en miljö.

U[illegible]liskussion

K[illegible] på följande länk för att få tillgång till verktyget http://www.netlogoweb.org/launch#http://www.netlogoweb.org/assets/modelslib/Sample%20Models/Art/Fireworks.nlogo. I rullgardinsmenyn ”Search the Models Library” väljer du alternativet sample models/Biology/Flocking. Det aktuella exemplet visas i nedanstående figur.



Figur 1. Exempel på flockbeteende i NetLogo

Den här modellen är ett försök att efterlikna fåglars flockbeteende. Varje fågel följer exakt samma regler: ”riktning”, ”avstånd” och ”sammanhållning”. ”Riktning”, som styrs med skjutreglaget maxalign-turn, innebär att fågeln tenderar att röra sig i samma riktning som andra fåglar i närheten. ”Avstånd”, som styrs med skjutreglaget max-separation-turn, innebär att fågeln undviker en annan fågel som kommer för nära. ”Sammanhållning”, som styrs med skjutreglaget maxcoherence-turn, innebär att fågeln rör sig mot andra fåglar i närheten (om inte en annan fågel kommer för nära). När två fåglar är för nära varandra åsidosätter ”avståndsregeln” de andra två reglerna, som avaktiveras tills ett minimiavstånd uppnås. Skjutreglaget för syn (vision) är det avstånd en fågel ser 360 grader runt om.

De gröna skjutreglagen (t.ex. max-fireworks) till vänster används för att parameterisera modellen. Knappen SETUP ställer in modellen enligt värdena på alla skjutreglage. Knappen GO kör modellen.

Uppgift i par

Bestäm först hur många fåglar det ska finnas i simuleringen och ställ in skjutreglaget för population på detta värde. Tryck först på SETUP och sedan på GO för att starta simuleringen (standardinställningarna för skjutreglagen ger ett hyfsat bra flockbeteende, men du kan leka med dem för att skapa variationer). Justera skjutreglagen för att få mer eller mindre sammanhållna flockar eller färre eller fler flockar.

F[illegible]beskrivning

Vi [illegible] epidemiologisk fråga som rör zikaviruset.

Zikaviruset är ett flavivirus som sprids via myggor. Det identifierades för första gången bland apor i Uganda 1947. Inkubationstiden (tiden från det att en individ utsätts för smittan tills

sjukdomen bryter ut) är uppskattningsvis 3–14 dagar. De flesta som utsätts för zikaviruset utvecklar inga symtom. Symtomen är i allmänhet lindriga, däribland feber, hudutslag, ögoninflammation, muskel- och ledsmärtor, illamående och huvudvärk, och varar normalt i 2–7 dagar. Viruset överförs främst genom bett av en infekterad mygga av släktet *Aedes*, huvudsakligen *Aedes aegypti*, i tropiska och subtropiska regioner. Myggor av släktet Aedes biter normalt under dagen, framför allt tidigt på morgonen och sent på eftermiddagen eller på kvällen. Det är samma mygga som sprider denguefeber, chikungunya och gula febern. Viruset överförs också från moder till foster under graviditeten, genom sexuell kontakt, transfusion av blod och blodprodukter samt organtransplantation (källa: Världshälsoorganisationen[1]).

I denna fallstudie ska vi använda ett hypotetiskt epidemiologiskt scenario och försöka förstå och bedöma risken för att viruset sprider sig. Vi använder följande scenario:

I en liten stad i Sydafrika har zikavirus konstaterats hos två personer. Regeringen vill ta reda på hur allvarlig situationen kan bli och vidtar därför nödvändiga förebyggande åtgärder. Staden har totalt 800 invånare, och enligt en första uppskattning är antalet myggor i området 1040 medan antalet predatorer (dvs. organismer som äter myggor) i området är 21.

Frågan är följande:

Hur många personer blir smittade och hur kan spridningen av viruset bäst förhindras?

För att besvara denna fråga ska vi använda NetLogo och köra simuleringar i en modell för zikaviruset.

F ng

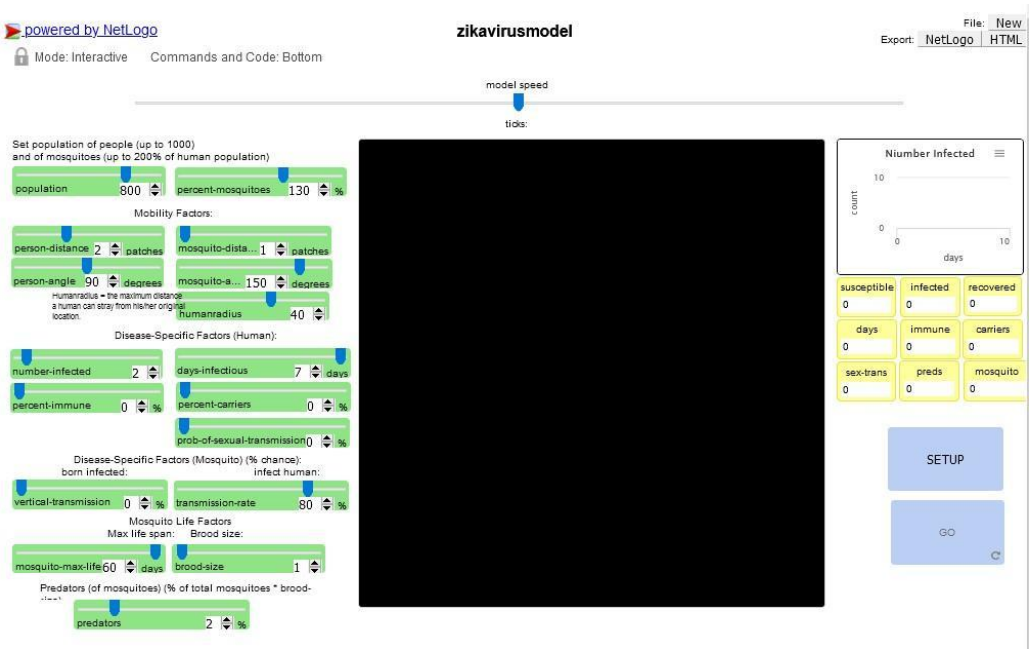
I d s använda en redan genomförd modell som gör det lättare att förstå hur modellering och simulering kan användas för att få ny kunskap om ett fenomen och hur de kan påverka beslutsfattandet.

Klicka på följande länk för att komma till NetLogo:

http://www.netlogoweb.org/launch#http://www.netlogoweb.org/assets/modelslib/Sample%20Models/Art/Fireworks.nlogo

Klicka på ”browse” för att ladda in modellen zikavirusmodel.nlogo. När modellen har laddats in bör följande exempel visas.

[1] https://www.who.int/news-room/fact-sheets/detail/zika-virus



Figur 2. Zikavirus i NetLogo

I modellen skulle myggorna bita personer som redan är smittade och därefter sprida sjukdomen genom att bita andra, friska personer. Det finns många variabler i den här modellen. I detta moment ska vi endast fokusera på följande:

- *person-distance* representerar människors rörlighet – hur långt de kan vända sig och hur många rutor (fält) de kan förflytta sig på en dag
- *number-infected* representerar antalet smittade från början
- *days-infectious* representerar den tid man normalt är smittad
- *percent-immune* avser den andel av befolkningen som är immun (= vaccinerad)
- *predators* representerar antalet predatorer
- *percent-mosquitoes* representerar antalet myggor

Innan modellen körs måste du göra följande i rätt ordning: för det första, ställa in skjutreglagen (om du vill justera dem), för det andra, klicka på SETUP (i skärmens nedre högra hörn) och för det tredje, klicka på GO för att köra modellen.

Diagrammet i övre högra hörnet och kontrollerna direkt nedanför visar hur samhället klarar viruset med dina inställningar.

Uppgift i par

Kör simulatorn med standardvärdena, som är i linje med problembeskrivningen i fas 1. När simuleringen är klar besvarar du följande fråga: Hur många smittades totalt och under hur många dagar?

*

Simulatorn ger dig en uppfattning om hur många som blir smittade under bestämda förhållanden. I detta exempel är antalet någonstans mellan 740 och 775. Med hjälp av vår modell har vi därmed kunnat besvara en del av frågan om hur många som skulle bli smittade av zikaviruset. Vad vi inte har besvarat är hur spridningen av viruset bäst kan förhindras. Detta är nästa fas inriktad på.

undersökning – Tolkning

Fle nna modell kan justeras för att undersöka hur sjukdomen skulle spridas bland befolkningen. För att undersöka hur spridningen av viruset bäst kan förhindras ska vi ta hänsyn till tre variabler: predators, person-distance och percent-immune.

Gruppuppgift

I denna uppgift ska vi undersöka hur följande förändringar påverkar spridningen av viruset och vilken som är mest effektiv.

1. Öka predators till 6 %: Ändra värdet till 6 % och klicka sedan på SETUP och GO. Håll koll på antalet tillfrisknade i kontrollerna i övre högra hörnet. Ställ in skjutreglaget för predatorer på 2 % igen (standardvärde) innan du fortsätter till 2.
2. Öka percent-immune till 6 %: Ändra värdet till 6 % och klicka sedan på SETUP och GO. Håll koll på antalet tillfrisknade i kontrollerna i övre högra hörnet. Ställ in skjutreglaget för immuna på 0 igen (standardvärde) innan du fortsätter till 3.
3. Minska person-distance till 1 fält: Ändra värdet till 1 och klicka sedan på SETUP och GO. Håll koll på antalet tillfrisknade i kontrollerna i övre högra hörnet.

*

Om du minskar person-distance till 1 fält blir antalet smittade 400–680. Om du ökar predators till 6 % blir antalet smittade 140–300, och om du ökar percent-immune till 6 % blir antalet smittade 630–740. Under modellens rådande förhållanden förhindras därför spridningen bäst genom att predators ökas till 6 %.

fattning

I vände vi modellering och simulering i ett hypotetiskt epidemiologiskt scenario för att undersöka spridningen av ett virus genom att ändra variablerna och utvärdera resultaten.

Diskussion

Hur kan modellering och simulering påverka beslutsfattandet i andra vetenskapsgrenar än epidemiologi?

Läranderesurser

zikavirusmodel.nlogo

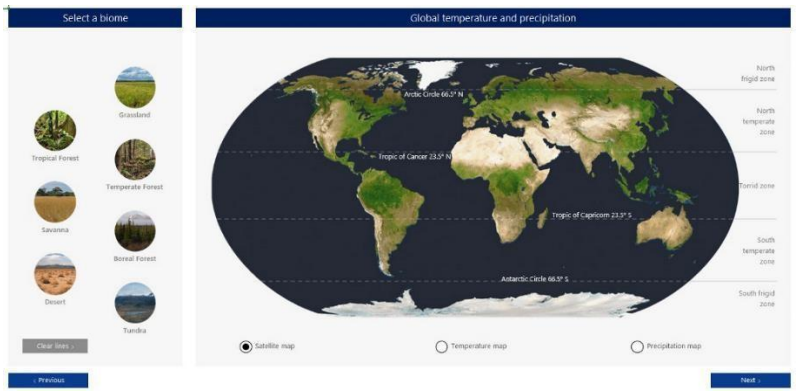
Moment 2.3: Använda Microsoft Excel för att ändra och återge data

Totalt: 2,5–3 timmar

Excel är ett kalkylark för att organisera data, utföra beräkningar med data samt analysera och återge data i tabell- eller diagramform. I denna aktivitet ska vi använda Excel för att analysera och återge data, identifiera trender i diagrammen och använda informationen för att besvara en fråga om klimatskillnader i biom och hur dessa påverkar färgerna på ytan.

U[illegible]diskussion

För[illegible]th colour workbook och går till arbetsbladet Mapping biomes, där det finns en bild av jordytan som sammanställts av satellitbilder (figur 1).



Figur 1. Arbetsbladet för kartläggning av biom

Uppgift i par

Klicka på biomknapparna till vänster om satellitbilden av jorden för att visa olika biom.

- Vilken färg kännetecknar varje biom bäst?
- Varför har de olika färger?

Tips: Välj knappen Temperature map respektive Precipitation map nedanför satellitbilden för att se sambandet mellan färger och fysiska förhållanden.

*

Som du ser är områden med riklig nederbörd grönare, medan högre temperaturer och mindre nederbörd förekommer i områden som är brunare. Polerna har mycket låga genomsnittstemperaturer, vilket ger upphov till det vita på grund av snö och is.

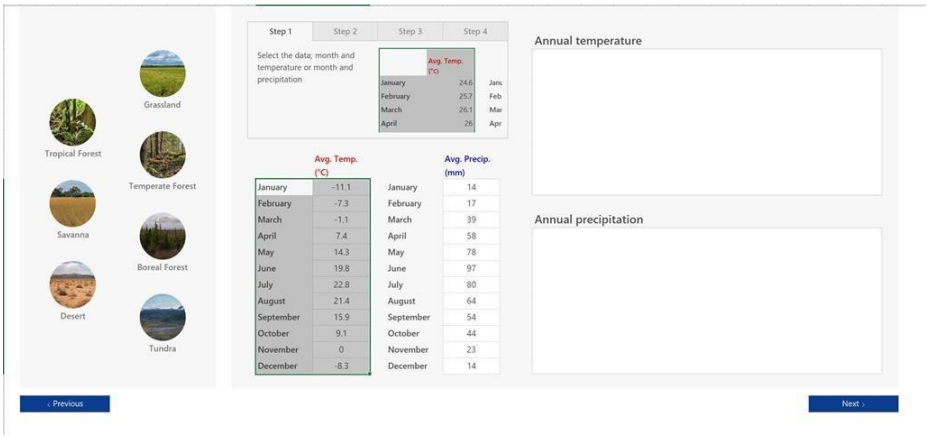
F[illegible]beskrivning

I d[illegible] vi använda data för att bättre förstå varför biomfärgen förändras. Vi ska undersöka sju terrestra biom och deras karakteristiska färger sedda från rymden. Färgerna beror på faktorer som temperatur och nederbörd. Frågan är följande:

Hur förändras temperaturen och nederbörden i olika biom under året och hur återspeglas förändringen i färgerna?

F[illegible]över temperatur och nederbörd

I uppvärmningsuppgiften undersökte du data för genomsnittlig global temperatur och nederbörd och började se sambanden mellan jordytans färger och den genomsnittliga årliga nederbörden och temperaturen. I denna fas ska vi undersöka hur nederbörden och temperaturen i biom förändras under året. För följande uppgift går du till arbetsbladet ”Create a biome climate chart”, som visas i följande figur.



Figur 2. Arbetsblad för att skapa klimatdiagram för biom

Uppgift i par

I denna uppgift väljer du ett biom som du vill titta närmare på. Om du klickar på en biomknapp till vänster visas data om temperatur och nederbörd per månad nedanför ”Avg Temp” och ”Avg. Precip”.

1. Välj uppgifterna om temperatur och klicka på alternativet infoga i Excels menyrad. Klicka på alternativet rekommenderade diagram och välj ett diagram som du vill använda för att visa uppgifterna om temperatur. 2. Upprepa samma steg för uppgifterna om nederbörd.

Hur varierar temperaturen och nederbörden under året för ditt biom? Observera vilka månader temperaturen och nederbörden är högst respektive lägst.

Välj ett annat biom och upprepa uppgiften genom att skapa ett diagram för temperatur och nederbörd. Jämför dessa två biom för att se vilket som uppvisar störst eller minst variation i temperatur respektive nederbörd under året.

Valfri uppgift

Upprepa ovanstående uppgift för alla biom och välj olika typer av diagram: Observera vilka diagram som bäst visar förändringar i temperatur och nederbörd samt trender, skillnader och likheter mellan två eller fler biom.

*

Nu har vi besvarat den första delen av frågan om hur temperaturen och nederbörden förändras under året för ett biom. Med hjälp av diagram för temperatur och nederbörd kunde vi observera dessa variationer och jämföra olika biom. Vad vi måste undersöka ytterligare är dock hur dessa förändringar i temperatur och nederbörd återspeglas genom färger.

F färger med temperatur och nederbörd i biom

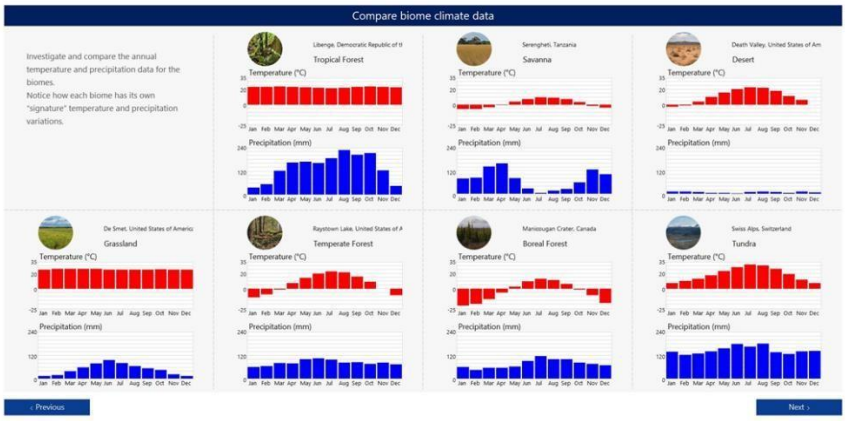
I f vände vi diagram för att visa och jämföra förändringar i temperatur och nederbörd för två biom. I denna fas ska vi korrelera biomegenskaper, temperatur och nederbörd och undersöka hur säsongsbundna vegetationsförändringar återspeglas i biomlandskapets tydligaste färg en viss månad under året.

Uppgift i par

Gå till arbetsbladet ”Information on biomes”, som visas i följande figur. Välj samma biom som i föregående fas och läs informationen om deras vegetation. Vad lägger du märke till när det gäller temperatur, nederbörd och vegetation?

Figur 3. Arbetsblad för biominformation Vad du lägger märke till genom att jämföra ett bioms temperatur, nederbörd och vegetation är att vegetationen är särskilt anpassad till biomets klimatförhållanden. Du lägger också märke till att även om temperaturerna i ett område kan likna dem i ett annat område har nederbörden stor inverkan på typen av vegetation i ett biom och därmed dess färger.

För följande fråga går du till arbetsbladet ”*Compare biome climate data*”, som visas i följande figur.



Figur 4. Arbetsblad för att jämföra biomklimat

Uppgift i par

Med ledning av uppgifterna om temperatur och nederbörd:

- Vilka biom kommer att ha de mest stabila färgerna under året?
- Vilka biom kommer att ha de mest varierande färgerna under året?

Motivera.

*

Färg kan användas för att beskriva biomförändringar under året. Vegetationsförändringarna återspeglas i biomlandskapets tydligaste färg en viss månad under året.

För följande fråga går du till arbetsbladet ”Biome colours through the year”, som visas i följande figur.

Figur 5. Arbetsblad för biomfärger under året

Uppgift i par

Med ledning av uppgifterna om temperatur och nederbörd:

- Vilka biom kommer att ha de mest stabila färgerna under året?
- Vilka biom kommer att ha de mest varierande färgerna under året?

Motivera.

*

För följande fråga går du till arbetsbladet ”Compare biome colour data”, som visas i följande figur.

Figur 6. Arbetsblad för att jämföra biomfärger

Uppgift i par

Hur påverkar temperaturen och nederbörden de tydligaste färgerna för olika biom? Exemplifiera.

Tips: Observera hur färgerna i olika biom står i samband med nederbörd och temperatur.

*

Du kanske har lagt märke till att områden med snö har blågrön färg, medan områden med riklig nederbörd är grönare, och att högre temperaturer och mindre nederbörd förekommer i områden som är brunare.

För att besvara den andra delen av frågan – *hur förändringar i nederbörd och temperatur återspeglas i färgerna* – observerade vi att vegetationsförändringar på grund av varierande nederbörd och temperatur återspeglas i biomlandskapets tydligaste färg en viss månad under året.

Fö a klimatförändring – fördjupningsmoment (valfritt)

I fö besvarade vi frågan hur temperaturen och nederbörden förändras i olika biom under året och hur förändringen återspeglas i färgerna. I denna fas ska du undersöka hur en klimatförändring påverkar ett valfritt biom. I detta syfte ska du samla in och analysera data om nederbörd och temperatur för detta biom och undersöka hur färgerna kan förändras om man inte gör någonting åt klimatförändringen.

Gruppuppgift

Undersök en klimatförändring som påverkar ett valfritt biom.

1. Samla in data om temperatur och nederbörd för varje månad under de senaste 10 åren för att se om det finns trender eller intressanta mönster. Skriv in uppgifterna i lämpligt format i Excel.
2. Skapa diagram över temperatur och nederbörd för att se trender.
3. Fundera över den klimatförändring som berör ditt biom och hur den återspeglas i de data du har samlat in. Diskutera hur färgerna i detta biom kan ha förändrats till följd av klimatförändringen och förutsäg hur färgerna i området kan förändras under de kommande 10 åren om man inte gör någonting åt klimatförändringen.

Läranderesurser

Arbetsböcker: Earth colour workbook.xlsx

AVSNITT 3: Datalogiskt tänkande inom programmering

Totalt: 10 timmar

I detta avsnitt ska du bekanta dig med programmeringens grunder och använda algoritmiskt tänkande inom ramen för historieberättande och spel. Du lär dig också att använda flödesscheman för algoritmer.

Bidrag till läranderesultaten

Läranderesultat

- Utveckla och skriva ned idéer till en berättelse på ett standardiserat sätt
- Fastställa huvudskillnaderna mellan elektroniska interaktiva berättelser och traditionella böcker
- Fastställa och förklara algoritmen för ett befintligt program
- Ändra ett befintligt program
- Felsöka och korrigera ett befintligt program
- Planera och utforma ett nytt program för en interaktiv berättelse
- Skapa och utveckla en interaktiv berättelse med hjälp av programmerbara element
- Använda slingor, variabler, meddelanden, if-satser och sekventiella instruktioner i ett program
- Förstå betydelsen av korrekta instruktioner
- Förstå vad en algoritm är
- Återge en algoritm i ett flödesschema
- Utvärdera en algoritms ändamålsenlighet
- Använda en färdig algoritm eller ett flödesschema i Scratch
- Förklara vad som menas men termen ”variabel”
- Skapa och använda variabler i ditt program
- Förklara vad som menas med termerna ”selektion” och ”slinga”
- Använda selektions- och slingsatser inom en algoritm eller ett program

Moment 3.1: Historieberättande i Scratch

Det digitala berättandet använder digitala medier (bilder, tal, musik, rörelse) för att berätta en historia. Under de senaste åren har det digitala berättandet blivit ett populärt och effektivt sätt att uppnå en rad lärandemål, särskilt för datalogiskt tänkande.

I detta avsnitt ska du öva på digitalt berättande i Scratch. Scratch är en gratis visuell och blockbaserad programmeringsmiljö för utbildning. I Scratch kan studenterna utveckla sina idéer i form av projekt med programmerbara medier som videor, bilder, spel och animationer.

U i

Tit e to Scratch.

Titta på detta exempelprojekt av åttondeklassare om det periodiska systemet.

*

Med Scratch

- formulerar du ett problem för att fastställa hur du ska använda byggstenarna i Scratch för din berättelse -- med handling, miljö, sekvens och perspektiv,
- organiserar och analyserar du logiskt data genom att skapa block av kod för figurer och deras miljö,
- återger du berättelsens innehåll genom sprites -- figurerna i Scratch,
- använder du algoritmiskt tänkande för att utveckla kod som får sprites att röra sig och kommunicera.

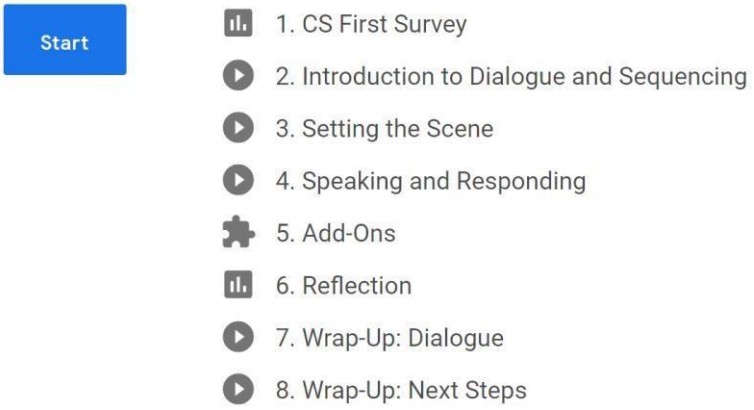
Förberedelser

Vi förutsätter att din lärarutbildare har gett dig användarnamn och lösenord till CS First. Gör följande innan du börjar:

- Öppna ett nytt fönster i webbläsaren och gå till g.co/CSFirst.
- Klicka på ”Sign in” i övre högra hörnet.
- Klicka på ”I am a student”.
- Klicka på ”Sign in with CS First”.
- Klicka på ”Enter class code”.
- Ange klasskoden 3h24s3.
- Ange användarnamn och lösenord.

Dialog

Lär dig använda CS First och skapa sedan en berättelse där två figurer pratar utan att ställa frågor.



U… i… e… rtknappen.

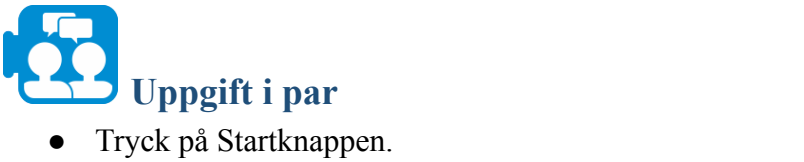
- Hoppa över enkäten och välj 2: Introduction to Dialogue and Sequencing ● Lägg till följande flik i webbläsarens fönster: https://scratch.mit.edu/
- Om du inte har något Scratchkonto får du registrera dig.
  - Tips: Du kan snabbt växla mellan flikarna med hjälp av ctrl tab ● Starta videon vid 2:30, och följ instruktionerna i slutet.
- Tryck på Next.
- Setting the Scene (3)
  - Nu ska du ha skapat ett nytt scratchprojekt, t.ex. med namnet minFörstaHistoria.
- Starta videon och följ instruktionerna i slutet: du måste inte avbryta videon.
- Tryck på Next.
  - Speaking and Responding (4)
- Starta videon och följ instruktionerna i slutet.
- Tryck på Next.
  - Add-Ons (5) ● Välj ”Adding Motion” nederst på sidan.

- Tips: Du måste inte ange koordinaterna själv om du först placerar figuren på önskat ställe innan du väljer motsvarande förflyttningsblock.

- Välj ”Add a Third Character”.

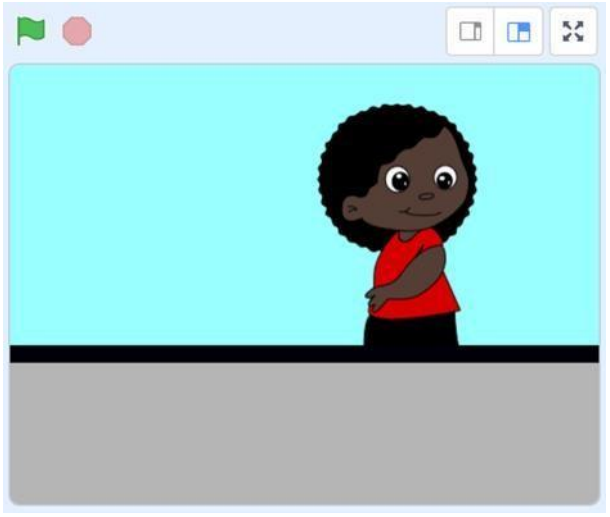
K

Be[illegible] där en figur går genom en miljö och beskriver vad den ser.



Uppgift i par

- Tryck på Startknappen.
  - 0 Du måste inte titta på första videon, utan klicka i stället på länken Starter Project 1. Då öppnas ett startprojekt med färdig kod.



- Tryck på Next.

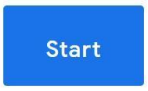
Miljö

Skapa en stormig miljö med regn och blixtar.

Uppgift i par

- Tryck på Startknappen.
- Öppna Rainy Day Starter Projec t.
    - 0 Unexpected Encounter (2)
- Följ instruktionerna i videon.
- Tryck på Next.
    - 0 Add-Ons (3)

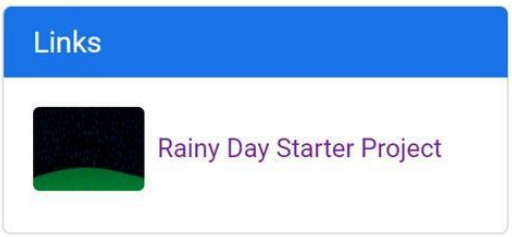
j ”Add Sound” och följ inst

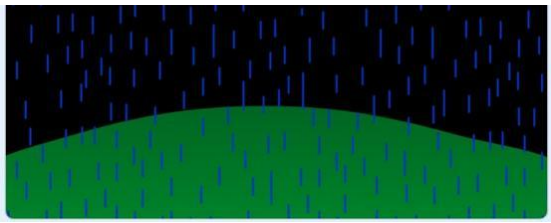


- Gå till
    - 0 Introduction to Setting a
- Titta på videon.

xt.
0 it Rain (2)

- I videon förklaras hur du lägge Du använder en särskild regndroppar, som är programm och ned.
- Följ i
- Trycl
    - 0
- Följ i
- Trycl
    - 0





- Följ instruktionerna i videon.
- Gå till
    0 Add-Ons (6)
- Titta på den korta videon och välj en eller två add-ons nederst på sidan.

Moment 3.2: Skapa ett labyrintspel Introduktion: Algoritmer

En algoritm är rad instruktioner för hur man steg för steg löser en uppgift. Om algoritmen har utformats tillräckligt exakt kan någon annan följa stegen exakt som du har tänkt. Algoritmer som ska utföras av en dator är formulerade med ett mycket exakt språk, programkod, som steg för steg talar om exakt vad datorn ska göra.

Uppgift i par

Titta på följande video: What is an algorithm and why should you care?

Fas 1: Utforska algoritmer med hjälp av flödesscheman

Studera följande algoritm.

1. Rita en vertikal linje.
2. Rita en horisontell linje som korsar den vertikala linjen.
3. Rita en diagonal linje från den vertikala linjens ände till den horisontella linjens ände.
4. Upprepa instruktion 3 för alla kvarvarande hörn.
5. Rita en vågig linje från den nedre änden.

G[illegible]

Rit[illegible]essa instruktioner.

Jämför din bild med de andras bilder. Diskutera varför det var eller inte var lätt och förklara varför.

*

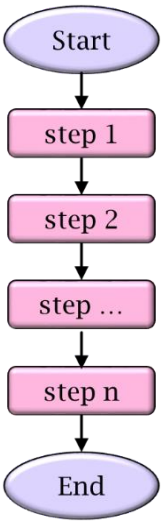
Resultatet av algoritmen skulle föreställa en drake (se bilaga A). Exemplet visar att resultatet kan bli mångtydigt om instruktionerna är oklara eller ofullständiga. Ibland förväntas den som utför instruktionerna själv reda ut eventuella oklarheter. Fundera över vad som skulle hända om vi alltid följde instruktioner precis som de är.

U

Tit e [utmaningen med exakta instruktioner](#).

*

Om vi vill beskriva en algoritm måste vi komma överens om ett språk för detta. Ett sådant språk måste vara tillräckligt exakt: om man följer algoritmen på det språket ska det stå klart vad man ska göra. En algoritm kan återges visuellt i ett flödesschema. Denna översikt gör det enklare att skriva, läsa och analysera instruktionerna.



De enklaste algoritmerna består av en serie instruktioner som verkställs i följd. I ett flödesschema visar vi en sådan sekvens av instruktioner som till höger. Du börjar på ”Start” högst upp. Delen mellan ”Start” och ”"End” (själva flödesschemat) beskriver vad algoritmen faktiskt gör.

Ö

Rit d

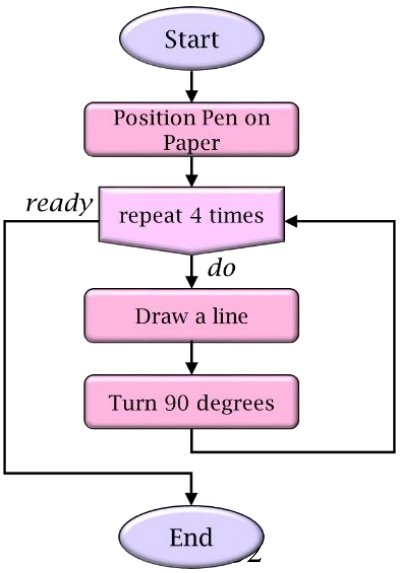
Vi antar att du kan använda följande grundläggande kommandon:

- **Rita en linje.**
- **Vänd 90 grader.**
- **Sätt pennan på papperet.**

Välj lämpliga instruktioner och placera dessa i ett flödesschema för att rita en kvadrat.

*

Din lösning består antagligen av en lång serie instruktioner med upprepningar av kod. Vi kan skriva en sådan serie på ett kompaktare och därmed tydligare sätt genom en upprepningskonstruktion: så länge vi inte har gjort 4 repetitioner följs hoppinstruktionen ”do”. Efter den fjärde iterationen följer vi klar-pilen och algoritmen avslutas.

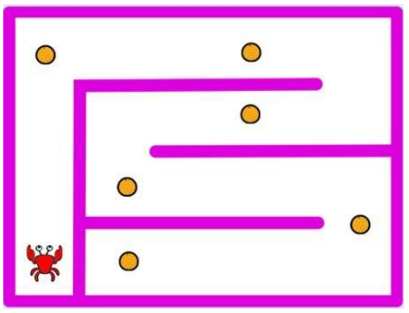


U

- st väljer en bild och skriver instruktioner för att förklara för sin kamrat hur man ritar bilden.
- Kom överens om vilka grundläggande kommandon som får användas i beskrivningen.

- Den andra studenten följer noggrant dessa instruktioner.
- Diskutera lösningen och rätta till misstag.

Fas 2: Skapa ett labyrintspel med hjälp av sprites



I denna fas ska du skapa ett spel i Scratch där spelaren styr en krabba med hjälp av piltangenterna. Målet är att hitta fram till labyrintens andra sida utan att komma i kontakt med väggarna. Det finns apelsiner i labyrinten. Spelaren samlar poäng när krabban plockar apelsinerna.

Läranderesurser

Scratchfil: crab-in-maze-start.sb3

Uppgift

Förbered en bild av spelet. Gör en grovskiss. Fundera över hur spelet ska se ut och hur huvudfiguren styrs av spelaren. Försök formulera villkoren för att spelet ska avslutas. Vilka ”skatter” ska det finnas i spelet? Var finns de och hur plockar spelaren dem?

*

För att programmera spelet måste du lära dig att

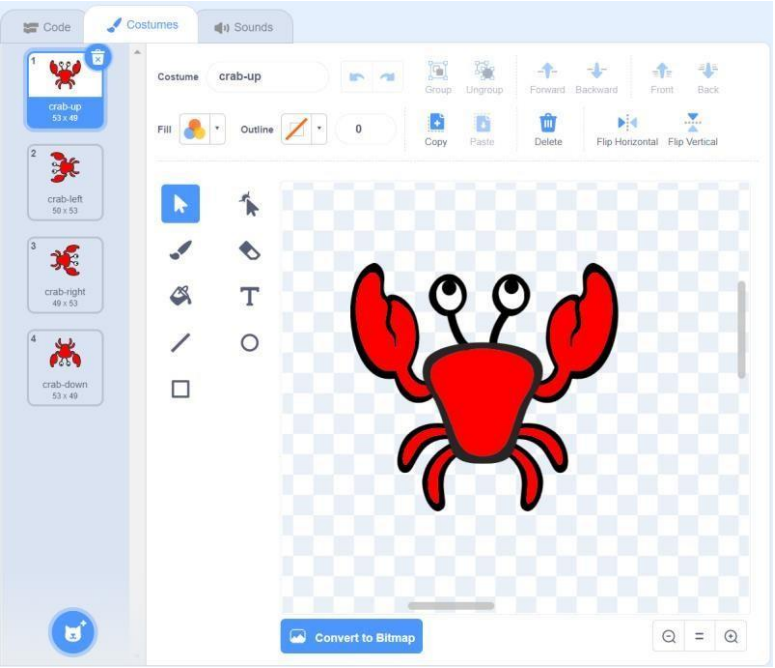
- definiera figurernas utseende, placering och rörelser,
- hantera tangentbordskommandon,
- hantera figurernas interaktion med miljön,
- programmera figurernas interaktion med varandra,
- använda variabler för poängräkning.

Vi börjar med att finjustera utseendet för spelets huvudfigur: krabban (implementerad som en sprite i Scratch).

Uppgift

- Gå till Scratch och öppna filen crab-in-maze-start.sb3 (genom att välja ”load from your computer” i filmenyn).
- Välj krabban och sedan fliken Costumes.
- Ta bort den andra bilden och välj den första bilden. Förminska krabbans klor lite och anpassa hela bilden skalenligt så att krabban lätt får plats i labyrinten (så att den inte kommer i kontakt med väggarna).

- Duplicera bilden 3 gånger (genom att högerklicka på bilden och välja ”duplicera”) och rotera kopiorna av krabban i fågelperspektiv: vänster, höger, upp och ned.

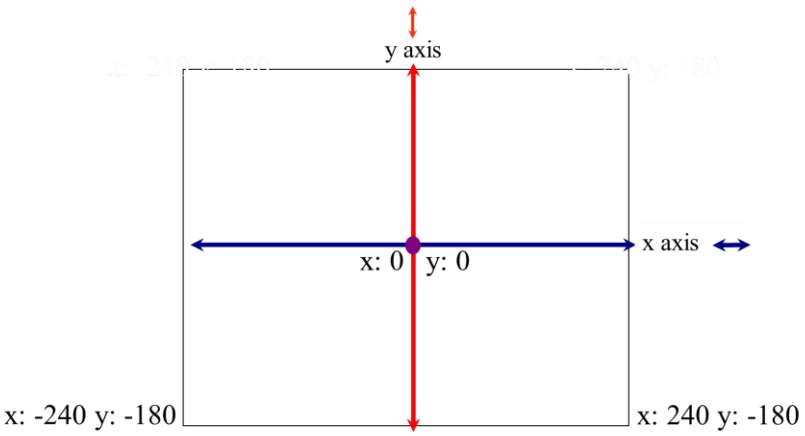


- Ändra namnen därefter.

*

För att programmera krabbans rörelser måste du veta hur en sprite placeras i Scratch. I Scratch betecknas varje punkt på spelplanen med ett nummerpar: det första numret avser det horisontella avståndet från spelplanens mitt och det andra numret det vertikala avståndet.

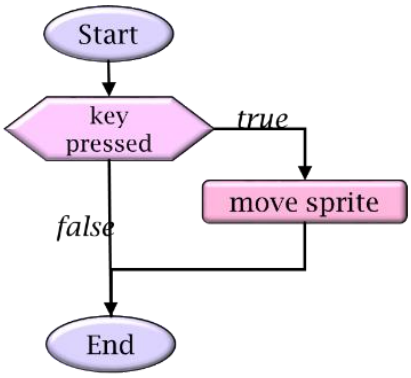
I mer tekniska termer använder Scratch ett så kallat koordinatsystem med nollpunkten i mitten av spelplanen. Varje punkt representeras av ett par (x, y) där x är det horisontella avståndet från nollpunkten och y är det vertikala avståndet.



Spelplanen är totalt 480 pixlar bred och 360 pixlar hög. Det övre vänstra hörnet och det övre högra hörnet har koordinaterna (–240,180) respektive (240,180).

Tillbaka till vårt spel. Spelaren flyttar krabban med hjälp av piltangenterna. Frågan är nu hur programmet vet att spelaren har tryckt på en tangent och hur vi sedan flyttar krabban som spelaren vill.

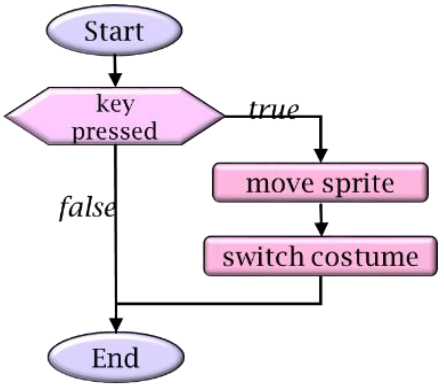
Den algoritm som används när en piltangent trycks ned kan beskrivas med följande flödesschema:



Vi använde en selektion här, även kallad om-så-konstruktion. I en sådan selektion fastställs det på grundval av ett villkor (anges i det sexkantiga blocket) huruvida den villkorliga hoppinstruktionen (hoppinstruktionen med beteckningen ”true”) ska verkställas. Det senare inträffar endast om villkoret är uppfyllt.

- på vilket *händelseblock* du kan använda när piltangenter trycks ned
- Justera krabbans position. Vilket *rörelseblock* behöver du?

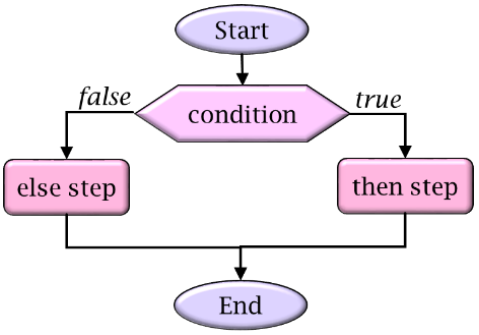
  För att göra spelet mer realistiskt ska vi anpassa bilden av krabban till rörelseriktningen. Det gör vi genom att alltid välja rätt utseende. Uttryckt i ett flödesschema:



- Fastställ vilket *utseendeblock* du behöver för detta. ● Kör och testa koden.

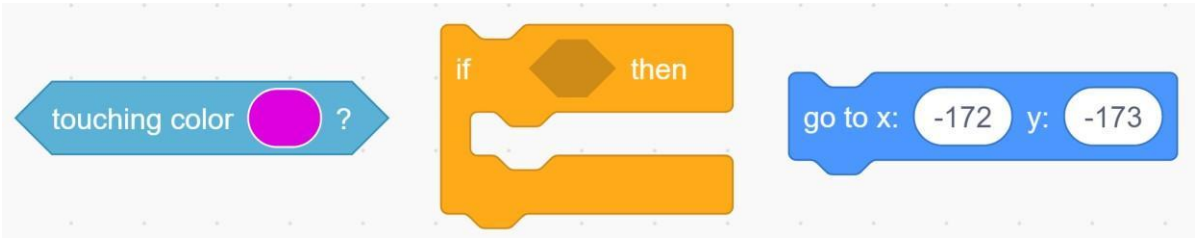
*

Med en allmän selektionssats kan en handling utföras även om villkoret inte är uppfyllt. I föregående exempel hände det inget om villkoret inte var uppfyllt. I ett flödesschema anges en om-så-annars-sats på följande sätt:



Om villkoret är uppfyllt verkställs den villkorliga hoppinstruktionen (med beteckningen ”true”). Annars verkställs hoppinstruktionen ”annars” (med beteckningen ”false”).

Scratch har följande block för att flytta krabban och kontrollera om den kommer i kontakt med väggarna.



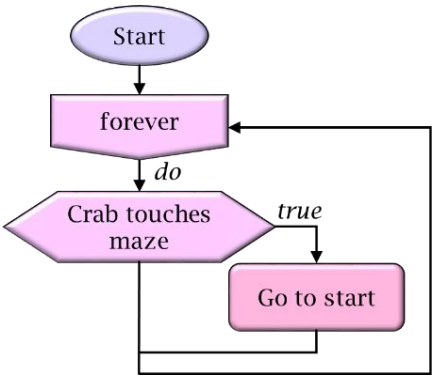
Uppgift i par

Avgör vilka block du behöver och i vilken ordning de ska användas.

- Om krabban kommer i kontakt med väggarna, hur den ska flyttas tillbaka till början.
  - Lägg till lämpliga block i koden. ● Kör och testa koden.
    - Funkar det? Om inte, försök komma på varför och rätta till det.

*

Vi har redan konstaterat att vi med hjälp av ett flödesschema kan ange att en viss del av algoritmen måste upprepas ett visst antal gånger. Ibland kanske vi vill ange att algoritmen alltid ska upprepas och därmed aldrig avslutas. I vårt krabbspel kan vi använda detta för att gång på gång kontrollera om krabban kommer i kontakt med väggarna och måste börja om från början. I ett flödesschema:



Observera att detta flödesschema inte har någon slutpunkt.

U
v hetsslingan (från *styrblocken*) för krabbans rörelser.

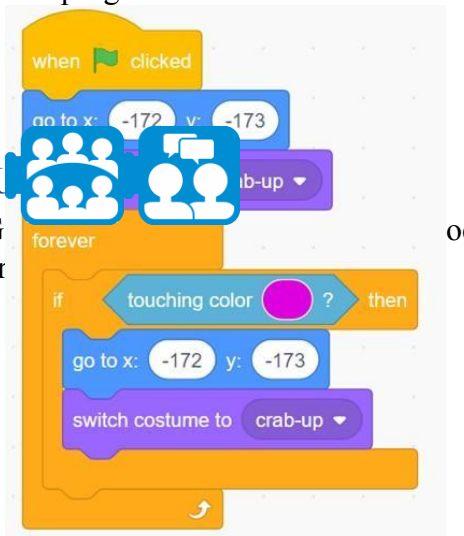
- Tänk ut vad som måste utföras om krabban går in i en av väggarna. ● Kör och testa koden.

*

Ett naturligt sätt att lösa stora problem är att dela upp dem i en rad delproblem, som kan lösas mer eller mindre separat och sedan kombineras för att lösa hela problemet. När en algoritm utformas och problemet delas upp i deluppgifter bör problemlösaren endast ta hänsyn till delalgoritmens funktion för helheten och bortse från detaljerna i deluppgiften. Uppdelningen i problemområden kallas abstraktion. Genom abstraktion kan programmeraren dölja alla detaljer om delalgoritmerna för att minska komplexiteten och därmed göra det lättare att förstå programmet.

Scratch använder abstraktioner för att underlätta programmeringen genom grundläggande byggblock vars komplexa underliggande detaljer förblir dolda. Scratchprogrammeraren kan använda abstraktioner genom att själv lägga till nya block i programmet.

Följande snuttar av programkod kan exempelvis vara en lösning på föregående problem. Till vänster ser du lösningen med alla detaljer. Till höger har abstraktion tillämpats genom ett nytt byggblock (kallat Start) för krabbans rörelse och byte av utseende, som sedan används i huvudprogrammet.

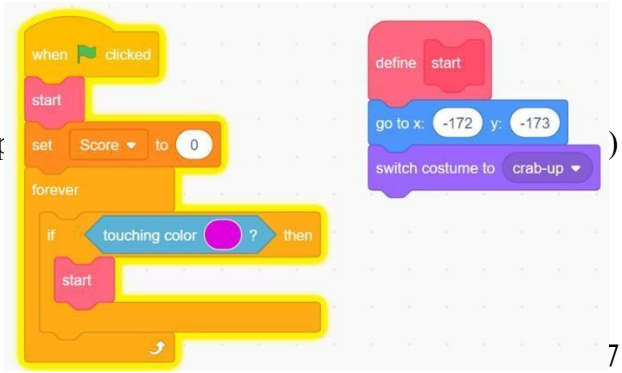


U

G och tillämp)

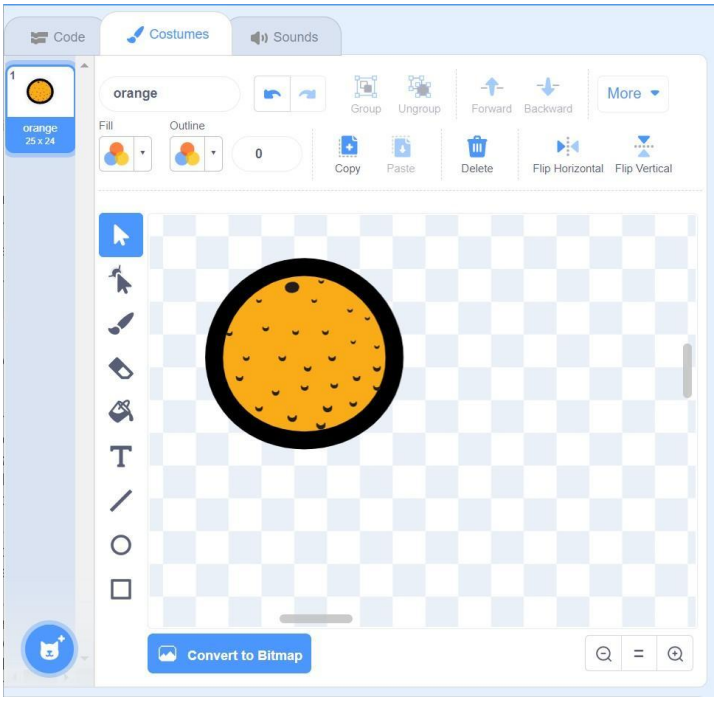
pr

*



En spelare ska nu kunna styra krabban med piltangenterna och krabban ska reagera på rätt sätt när den stöter mot en vägg. Vi ska nu göra spelet intressantare genom att lägga till utmaningar. Till att börja med nöjer vi oss med apelsiner som vi placerar ut i labyrinten så att krabban kan äta dem. Detta sker så snart krabban kommer i kontakt med dem. Varje apelsin krabban äter upp ger en bonuspoäng. Spelets mål är nu att få så många bonuspoäng som möjligt.

- Välj en apelsinsprite och anpassa den skalenligt så att storleken är i proportion till krabban.



Nu står vi inför ett intressant problem: vem ska upptäcka och hantera interaktionen mellan krabban och apelsinen? I verkligheten är detta alltid krabban, men här har vi ett alternativ. Vi kan ge apelsinen större ansvar än i verkligheten. Och nu ska vi använda detta intressanta alternativ: apelsinen ska upptäcka om krabban har hittat den och låta sig ätas. Därför döljer vi apelsinen.

- Här ser du de block du (troligen) behöver för att programmera apelsinerna. Använd dessa block för att uppnå önskat beteende.

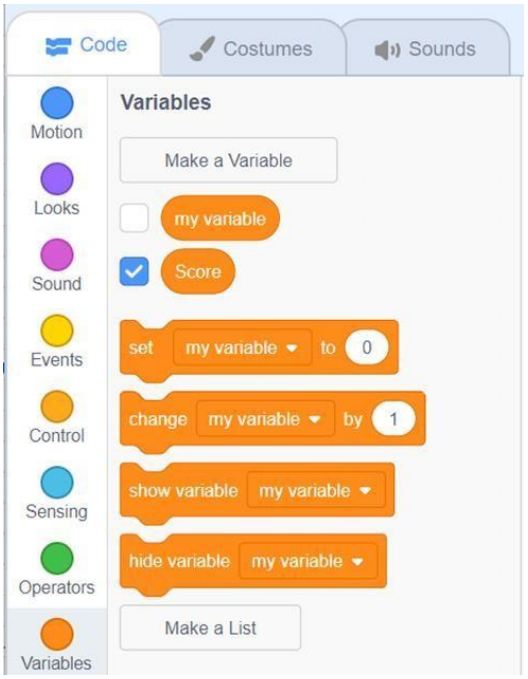
- Kör och testa din kod!

- Bör det hända något mer vid denna punkt?

*

Vi vill också göra det möjligt för programmet att räkna poängen. Vi ska använda en så kallad variabel för detta ändamål. I programmeringsspråk är en variabel ett kärl som kan innehålla en informationsbit i taget, som ett ord eller ett nummer. Det innebär att vi kan hänvisa till och ändra informationsbiten på olika ställen i ett program. Detta gör variabler ytterst användbara.

Hur skapar vi en variabel i Scratch? Innan vi kan använda en variabel måste vi först skapa den med hjälp av knappen ”Make a Variable” i blockpaletten. En variabel i Scratch kan antingen innehålla en text eller ett nummer. Dessutom finns det block som kan ge ett (första) värde till en variabel och ändra värdet när programmet körs. Till höger ser du vilka block du kan använda för att ändra variabler. Värdet i exempelvis blocket ”set” behöver förresten inte vara ett nummer: komplexare uttryck bestående av operatorer från operatorspaletten kan också användas här. Anta exempelvis att Score har värdet 4. Efter exekveringen av



kommer Score att ha värdet 3 * 4 + 2 = 14.

U

g variabel med beteckningen Score i ditt program.

- Fundera på vilket värde variabeln ska ha i början och hur du kan ställa in det värdet.
- Fundera också över när och hur värdet för Score ska justeras och vilket block som kan användas för detta.
- Kör och testa din kod!
- Duplicera apelsinen (i spritefönstret) ett antal gånger (t.ex. 6) och placera dessa kopior någonstans i labyrinten.

AVSNITT 4: Undervisnings- och lärandestrategier för datalogiskt tänkande

Totalt: 8 timmar

Undervisningen och lärandet för datalogiskt tänkande har i huvudsak två former: utan dator och med dator. I detta avsnitt ska du få pröva båda formerna och lära dig principerna för utformningen av undervisningsstrategier för datalogiskt tänkande.

Bidrag till läranderesultaten

Läranderesultat
• Beskriva vad som kännetecknar undervisning och lärandemoment för datalogiskt tänkande med eller utan dator • Beskriva viktiga inslag i undervisningsstrategier för datalogiskt tänkande • Känna igen sådana inslag i konkreta undervisningsmaterial och aktiviteter

Moment 4.1: Databävern Bebras

Databävern är en internationell tävling och utmaning på internet. Tävlingen består av så kallade Bebrasuppgifter om ett eller flera datalogiska begrepp. I detta moment ska du utforska Bebrasuppgifter både som led i tävlingen och som lärandemoment för datalogiskt tänkande utan dator.

Uppgift i par

- Gå till Databäverns lokala webbplats och välj upp till fem exempeluppgifter (försök variera).
- Utför uppgifterna individuellt och jämför och diskutera dem sedan med din kamrat.
- Kan du systematisera uppgifterna enligt de steg och moment du har lärt dig i denna modul?

Bebrauppgifter som lärandemoment

Bebrauppgifter kategoriseras enligt datalogiskt begrepp, och varje uppgift åtföljs av en förklaring av sambandet mellan uppgiften och det informationstekniska området.

Valentina Dagienė (som hittade på Databävern) och Sue Sentance har undersökt hur Bebrasuppgifter används för datalogiskt tänkande: Dagienė, V., & Sentance, S. (2016). ”It's Computational Thinking! Bebras tasks in the curriculum”.

I *International conference on informatics in schools: Situation, evolution, and perspectives* (s. 28–39).

Uppgift i par

• Läs artikeln.

• Gå igenom de begrepp och moment du stötte på i avsnitt 2 och 3. För varje modul ska du hitta ett exempel på ett datalogiskt begrepp och motsvarande Bebrasuppgift.

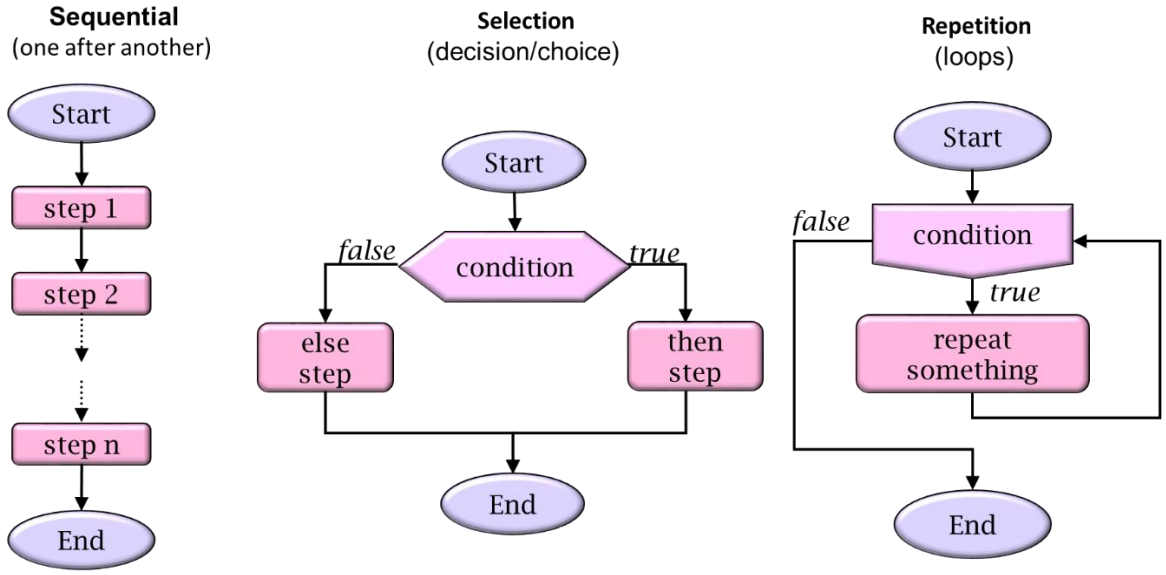
Metoder för att hitta rätt i en labyrint

För att ta sig genom en labyrint måste man ta sig genom labyrinten från start till mål. Vissa sätt att hitta rätt är avsedda att användas inuti labyrinten av någon som inte vet hur den ser ut, medan andra är avsedda att användas av en person (eller ett datorprogram) som kan se hela labyrinten på en gång. Att hitta rätt i labyrinter är en variant av att hitta den kortaste vägen mellan två punkter. När det finns en rad platser som kan vara sammanbundna (t.ex. städer som är sammanbundna genom vägar) söker en algoritm efter den bästa vägen mellan dessa platser (t.ex. den kortaste eller billigaste vägen).

Vårt syfte är att använda dessa problem för att få en första insikt i problemanalys och utformningen av algoritmer utan ingående kunskaper i datorprogrammering och programmeringsspråk. För att kunna formulera möjliga lösningar som är någorlunda exakta ska vid dock först använda flödesscheman igen.

Delar av flödesscheman

Nedan visas tre grundläggande byggblock i flödesscheman.



Varje flödesschema består av följande tre grundläggande delar:

1. **Sekvens**: En ordnad serie instruktioner som verkställs i följd.
2. **Selektion**: På grundval av ett villkor fastställs det huruvida den villkorliga hoppinstruktionen (hoppinstruktionen med beteckningen ”true”) eller hoppinstruktionen ”annars” (med beteckningen ”false”) ska verkställas.
3. **Repetition**: Själva flödesschemat (hoppinstruktionen med beteckningen ”true”) upprepas så länge villkoret är uppfyllt. Därefter följer programmet hoppinstruktionen ”false”.

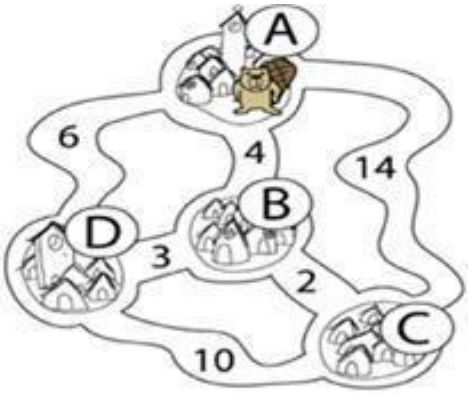
H[illegible]problemet

De[illegible]delsresandeproblemet är ett klassiskt exempel på en uppgift där en optimal lösning eftersträvas.

”Det finns en viss uppsättning städer med bestämda avstånd mellan varje stadspar: vilken är den kortaste möjliga vägen om man vill besöka varje stad och återvända till ursprungsstaden?”

Vi ska nu titta på en lösning på problemet som visserligen inte alltid hittar den bästa vägen, men däremot är intuitiv och lätt att förstå, nämligen *närmaste granne-algoritmen*. I närmaste granne-algoritmen väljer handelsresanden hela tiden närmaste stad som ännu inte har besökts.

Bävrarna simmar från stad A, besöker stad B, C och D och kommer sedan tillbaka till A. De vill hitta den kortaste vägen.



1. Vilken väg skulle de ta om de använder närmaste granne-algoritmen?
2. Ger närmaste granne-algoritmen alltid rätt svar (dvs. för alla uppsättningar av städer och förbindelser mellan dem)? Om inte, ge ett konkret exempel.

*

Även om grannalgoritmen är intuitivt lätt att förstå kan den vara svår att formulera exakt. Vi försöker uttrycka algoritmen i ett flödesschema där vi först måste avgöra vilka primitivinstruktioner vi kan använda. Dessa primitivinstruktioner måste vara kraftfulla men samtidigt abstrakta så att vi inte blir insnärjda i en massa detaljer.

Uppgift i par

Återge grannalgoritmen i ett flödesschema.

Tips: Tänk först på de grundläggande/primitiva instruktioner du får använda.

H[illegible]abyrint

Att [illegible]byrint är ett annat exempel på en uppgift som är lätt att förstå men samtidigt svår att lösa. Det finns flera varianter av det här problemet: hitta rätt ur en labyrint, hitta rätt genom en labyrint eller hitta vägen till en viss plats i labyrinten. För det första kan dessa varianter generaliseras: det handlar om att hitta vägen från punkt A till punkt B.

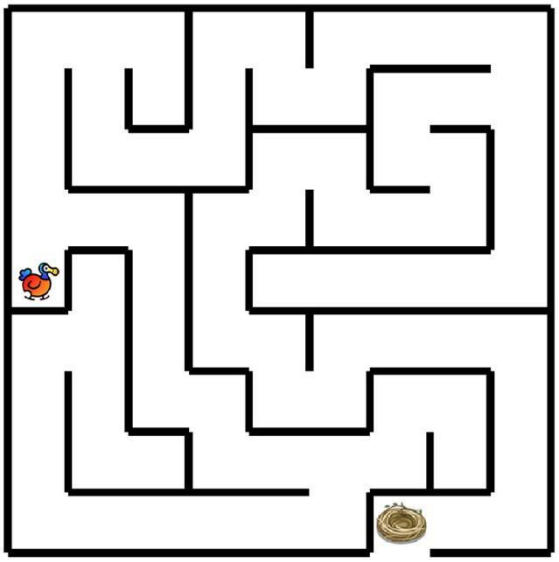
Vad vi vet om labyrinten när vi letar efter målpunkten B är också viktigt. I den här uppgiften antar vi att vi inte vet hur stor labyrinten är eller vad den har för struktur. Vi kan följa labyrintens passager och hela tiden se ett steg framåt. Vi antar därför att vi när som helst kan kontrollera om vi kan gå rakt fram (och därmed inte mötas av en vägg) och eventuellt ändra vår rörelseriktning.

Det mest kända sättet att ta sig igenom en labyrint är att följa en vägg (vänsterhands- eller högerhandsregeln). Om alla labyrintens väggar hänger ihop kan man vara säker på att inte tappa bort sig och komma till utgången genom att hålla handen på ena väggen.

För att konkretisera problemet använder vi följande scenario med en dront som försöker hitta sitt näste.

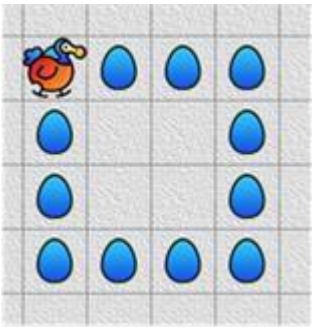
Innan vi kan ta fram en sökalgoritm måste vi på nytt ange med vilka primitivinstruktioner vi kan styra dronten.

Vi skiljer mellan *kommandon* (att få dronten att ta ett steg framåt eller ändra riktning) och *frågor* (genom vilka dronten kan lämna lokal information om sin omgivning).



- Kommandon:
- *vänd åt vänster*: vänder sig 90 grader moturs
- *vänd åt höger*: vänder sig 90 grader medurs
- *gå*: går ett steg framåt
- *lägg ägg*: lägger ett ägg (på den nuvarande platsen) - Frågor/test:
- *kan gå?* kan du ta ett steg framåt?
- *hittat nästet?* har du hittat nästet?

Använd ett flödesschema för den algoritm som gör att dronten lägger ägg i följande mönster. Tips: Använd repetitioner.



Uppgift i par

Återge strategin att följa vänster vägg i ett flödesschema.

Öva på att följa rätt väg, utan dator

Förberedelse

Märk ut en labyrint på golvet med maskeringstejp. Se till att passagerna är breda nog för en person. Labyrinten får vara lite enklare än labyrinten på bilden. Byt flödesscheman från föregående uppgift. Bestäm att en plats i labyrinten ska vara slutmålet. En av er ställer sig någonstans i labyrinten, på tillräckligt avstånd från slutmålet.

Uppgift i par

En av er ställer sig någonstans i labyrinten, på tillräckligt avstånd från slutmålet.

- Den andra personen läser instruktionerna i flödesschemat och personen i labyrinten följer dessa instruktioner noggrant.
- När personen slutmålet? Testa också om personen når målet från andra ställen i labyrinten. Om inte, vad är det för fel på algoritmen? Förbättra algoritmen och kontrollera om det fungerar.
- Återlämna den förbättrade versionen av flödesschemat till ägaren.

Läranderesurser

- Scratchfil: PathThroughMaze.sb3

Hur man programmerar i Scratch

Studenterna ska genomföra algoritmen från föregående uppgift i Scratch. Vi har skapat ett första scenario för detta. Öppna startscenariot PathThroughMaze.sb3.

Här ges en kortfattad förklaring. De primitiva drontinstruktionerna har lagts till i scenariot som separata block. Instruktionen *layEgg* saknas eftersom den inte behövs för uppgiften. Alla andra kommandon fungerar som vanligt. Frågorna har realiserats på ett särskilt sätt, även eftersom egna block i Scratch inte ger resultat. Med hjälp av ett färdigt block från känselpaletten kan det enkelt kontrolleras om dronten har hittat nästet. Vi har infört ett nytt block för testet *canMove*. För att få ett resultat använder vi en variabel som vi har kallat *canMove* precis som blocket. Så efter att du verkställer blocket *canMove* kan du titta på värdet för motsvarande variabel för att se om dronten kan ta ett steg eller inte. Variabeln *canMove* har värdet 0 eller 1. Värdet 0 visar att den inte kan ta ett steg, medan värdet 1 visar att den kan ta ett steg.

Uppgift i par

I startscenariot finns ett block som kallas *tryToFindExit*. Det här blocket gör inte så mycket: det visar bara meddelandet att utgången har hittats på skärmen.

- Realisera detta block genom att omvandla flödesschemat till Scratchkod.
- Kör programmet och testa också om dronten når målet från andra ställen i labyrinten. Justera koden vid behov.

Moment 4.3: Undervisningsstrategier

I detta moment ska du utforska strategier och pedagogiska principer för datalogiskt tänkande.

Tillbakablick Uppgift i par

Diskutera aktiviteten att hitta rätt väg:

- Vilka är de största skillnaderna mellan att använda dator och att inte använda dator?
- Vilka utmaningar medför respektive variant för dina framtida studenter?

Principer för datalogiskt tänkande

I en [illegible]serar Lye och Koh (2014) 27 empiriska studier om datalogiskt tänkande, särskilt studier om klassrumsaktiviteter med programmering. Lye och Koh identifierar bland annat undervisningsstrategier för datalogiskt tänkande.

Lye och Koh grupperade strategierna i fyra kategorier, som ofta förekommer i kombination: *förstärkning av datalogiska begrepp*, *reflektion*, *informationsbehandling*, och *programmera med stöd.*

Uppgift i par

1. Läs sammanfattningen om de fyra kategorierna i dokumentet ”Instructional strategies for computational thinking” (i läranderesurserna).
2. Analysera dina egna aktiviteter för datalogiskt tänkande i denna modul. Vilka strategier känner du igen? Diskutera.

Uppgift i par

1. Läs sammanfattningen om bedömning i datalogiskt tänkande (se dokumentet ”Assessment in CT”).
2. Diskutera lämpliga bedömningsmetoder för lärandemomenten i avsnitt 2 eller avsnitt 3. Välj en möjlig strategi för varje lärandemoment. Hur skulle du använda dessa strategier för att utvärdera studenternas datalogiska tänkande?

Slutsats Diskussion

Jämför resultaten i klassen. Vilka aspekter var enklare och vilka var svårare?
Vad lärde du dig?

Läranderesurser

- A. Instructional strategies for computational thinking (sammanfattning av strategier för undervisning och lärande om datalogiskt tänkande, baserat på Lye, S. Y., & Koh, J. H. L. (2014). ”Review on teaching and learning of computational thinking through programming: What is next for K-12?” *Computers in Human Behavior*, 41, 51–61.)
- B. Assessment in CT

Översikt av läranderesurser i modul O2

- Moment 1.1. – A. Brief introduction to CT
- Moment 1.1. – B. Definitions of CT
- Moment 2.1 – A. Gemeinschaft to Gesellschaft.pdf
- Moment 2.1 – B. The changing psychology of culture from 1800 through 2000
- Moment 2.1 – C. The changing psychology of culture in German-speaking countries
- Moment 2.2 – zikavirusmodel.nlogo
- Moment 2.4 – Earth colour workbook.xlsx
- Moment 3.2 – crab-in-maze-start.sb3
- Moment 4.1 – Dagiene och Sentance 2016 (Dagienė, V., & Sentance, S. (2016). ”It's Computational Thinking! Bebras tasks in the curriculum”. I *International conference on informatics in schools: Situation, evolution, and perspectives* (s. 28–39)).
- Moment 4.2. PathThroughMaze.sb3
- Moment 4.3. – A. Instructional strategies for computational thinking (sammanfattning av strategier för undervisning och lärande om datalogiskt tänkande, baserat på Lye, S. Y., & Koh, J. H. L. (2014). ”Review on teaching and learning of computational thinking through programming: What is next for K-12?” *Computers in Human Behavior*, 41, 51–61.)

  Moment 4.3. – B. Assessment in CT