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

TeaEdu4CT

Module 2

Algemene inleiding tot computationeel denken

Een basismodule die geschikt is voor alle leraren

Auteurs: Radboud University (Nederland)
Maria Kallia,
Sjaak Smetsers,
Erik Barendsen,
Christos Chytas

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

Externe Beoordelaars:
Piret Luik (Estland),
Renate Motschnig (Oostenrijk)

Uitproberen:
Ankara University (Turkije), KTH Royal Institute of Technology (Zweden), Radboud University (Nederland), University of Paderborn (Duitsland), Vienna University of Technology (Oostenrijk)

Vormgeving:
Vaidotas Kinčius (Litouwen)

Module *2* is gebaseerd op het werk binnen het project "Future Teachers Education: Computational Thinking and STEAM" (TeaEdu4CT). Coördinatie: Prof. Valentina Dagienė, Vilnius Universiteit, Litouwen. Partners: Technische Universiteit Wenen (Oostenrijk), CARDET (Cyprus), Tallinn University (Estland), Universiteit van Turku (Finland), Paderborn University (Duitsland), CESIE (Italië), Radboud Universiteit (Nederland), KTH Royal Institute of Technology (Zweden), Ankara University (Turkije). Het project is medegefinancierd door het Erasmus+ programma KA2.

Inhoud

Algemeen overzicht en doel

Het doel van deze module is aanstaande leraren een concreet inzicht te geven in computationeel denken (CT), een praktische kennis van de onderwijs- en leerprincipes voor CT, en hen vertrouwd te maken met computationele hulpmiddelen die het onderwijzen en leren kunnen ondersteunen.

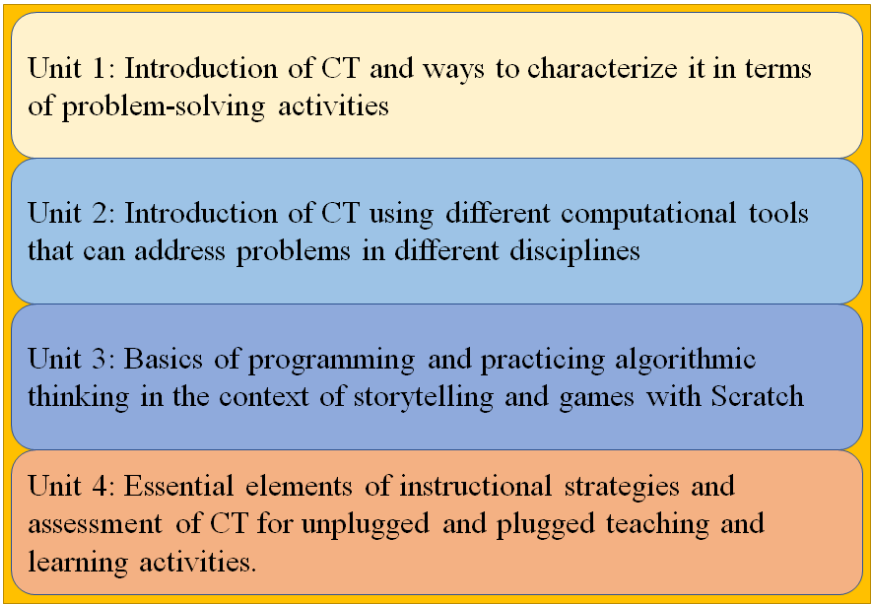
In deze module zullen de lerenden:

- het begrip computationeel denken verkennen en verschillende manieren vergelijken om het te karakteriseren in termen van probleemoplossende activiteiten
- zich bezig houden met computationeel denken-aspecten, -praktijken en -instrumenten, en de rol van computationeel denken binnen de disciplines begrijpen
- kennis maken met de basisbeginselen van programmeren en oefenen met algoritmisch denken in de context van games en storetelling
- ervaring op doen met zowel "unplugged" als "plugged" onderwijs- en leeractiviteiten, en leren over ontwerpbeginselen voor instructiestrategieën voor computationeel denken

De module beschouwt CT als een kader om vakoverschrijdende vaardigheden en competenties te ontwikkelen, geschikt voor elke vakdocent.

De modulestructuur

De module is onderverdeeld in de volgende vier secties: sectie 1 introduceert het concept CT en vergelijkt manieren om het te karakteriseren in termen van probleemoplossende activiteiten; sectie 2 introduceert CT door gebruik te maken van verschillende computationele tools (Ngrams, NetLogo, Excel) en demonstreert hoe deze tools kunnen worden ingezet om problemen in verschillende disciplines (Geschiedenis, Biologie, Aardrijkskunde) aan te pakken; sectie 3 richt zich op de basisprincipes van programmeren en op het oefenen van algoritmisch denken in de context van storytelling en games met Scratch. De module wordt afgesloten met sectie 4, waarin de nadruk ligt op "unplugged" en "plugged" onderwijs- en leeractiviteiten en waarin essentiële elementen van instructiestrategieën en beoordeling van CT worden besproken.



Doelgroepen en vereisten

Deze module is bestemd voor toekomstige leerkrachten die een lerarenopleiding volgen en voor de professionele ontwikkeling van leerkrachten die reeds in dienst zijn en belangstelling hebben voor computationeel denken. De module is ontworpen voor face-to-face leren, maar kan gemakkelijk worden aangepast als een module voor afstandsonderwijs.

Er zijn geen bijzondere voorwaarden voor het volgen van deze module. Het is raadzaam dat studenten een basiskennis hebben van de gebruikte instrumenten en de vorige module "O1: Kaders voor de ondersteuning van de modules: CT&STEM voor de toekomstige lerarenopleiding".

Leerresultaten en beoordelingsmethoden

Een succesvolle leerling die de hele module heeft doorlopen, zal in staat zijn om::

- het begrip computationeel denken uit te leggen en verschillende manieren te vergelijken om het te karakteriseren in termen van probleemoplossende activiteiten;
- probleemoplossingstaken uit te voeren met behulp van computationeel denken-concepten, -praktijken en -instrumenten, en de rol van computationeel denken binnen de vakgebieden uit te leggen;
- eenvoudige programma's in Scratch te bouwen en algoritmisch denken toe te passen in de context van verhalen en spelletjes;
- unplugged en plugged onderwijsstrategieën en leeractiviteiten voor computationeel denken toe te passen en de onderliggende ontwerpprincipes te beschrijven.

Meer specifieke leerdoelen zijn te vinden binnen de structuur van de verschillende secties.

Beoordelingsstrategie

De taken in de modules zijn mogelijkheden voor formatieve beoordeling en feedback. Er is een optionele (algemene) eindbeoordeling die als een apart document kan worden gevonden (zie leermiddelen: module Eindbeoordeling).

Moduleplan en didactische benaderingen

De module bestaat uit 4 secties van face-to-face interactie. Elke sectie bestaat uit verschillende activiteiten die gewoonlijk beginnen met een warming-up activiteit en eindigen met een reflectieve activiteit. De studenten zullen zich bezighouden met veel verschillende leerbenaderingen, waaronder het lezen van artikelen, of de gegeven literatuuroverzichten, groepsdiscussies, probleemoplossende taken, en reflectie-activiteiten.

Delen en activiteiten

Sectie 1: Inleiding tot het computationeel denken

Activiteit 1.1 Inleiding tot CT

- Computationeel denken als een probleemoplossende activiteit
- Het karakteriseren van elementen van computationeel denken
- Conclusie

Totaal: 2 uur

Sectie 2: Hulpmiddelen voor computationeel denken

Activiteit 2.1: Google Ngrams gebruiken

Activiteit 2.2: Modelleren en simuleren met NetLogo

Activiteit 2.3 Gebruik van Microsoft Excel om gegevens te manipuleren en weer te geven

Totaal aantal uren: 8 uur

Sectie 3: Computationeel denken via programmeren

Activiteit 3.1: Verhalen vertellen in Scratch

Activiteit 3.2: Een doolhofspel maken

Totaal aantal uren: 10 uur

Sectie 4: Computationeel denken onderwijzen en aanleren

Activiteit 4.1: Bebras opdrachten

Activiteit 4.2: Pathfinding: Paden in een doolhof

Activiteit 4.3: Instructiestrategieën voor Computationeel Denken

Totaal aantal uren: 8 uur

Sectie 1: Inleiding tot het computationeel denken

In deze sectie zult u het begrip computationeel denken verkennen en verschillende manieren vergelijken om het te karakteriseren in termen van probleemoplossende activiteiten.

Bijdrage aan de leerresultaten

Leerresultaten
• in staat zijn computationeel denken te beschrijven als een brug tussen een vakgebied en IT
• computationeel denken te karakteriseren in termen van probleemoplossende stappen en activiteiten die leiden tot een operationele, uitvoerbare oplossing
• elementen van computationeel denken herkennen in scenario's voor activiteiten van leerlingen

Activiteit 1.1 Inleiding tot computationeel denken

Om de volledige potentie van computers in te zetten bij uiteenlopende vakken en activiteiten, zijn meer digitale vaardigheden vereist dan het kunnen bedienen van een programma als Word of het onderhouden van een social media-account. Om een dergelijke verbinding tussen leerstof en informatietechnologie tot stand te brengen, zijn specifieke probleemoplossende vaardigheden vereist. Die vaardigheden behoren tot het domein van computationeel cenken

Computationeel denken als een probleemoplossende activiteit

Leestaak

Lees het document 'A brief introduction to computational thinking' (Een korte inleiding tot computationeel denken), dat te vinden is in de leermiddelen. Deze tekst onderscheidt drie belangrijke stappen in computationeel denken: de pijlen (1), (2) en (3) in het diagram.

Discussie Opdracht voor tweetallen

Bespreek de twee onderstaande klassenscenario's, overgenomen uit Yadav et al. (2018, p. 381). Welke activiteiten zou je beschouwen als computationeel denken? Hoe verhouden ze zich tot de stappen (1), (2) en (3) in Materiaal A?

Scenario 1:

De Westwood basisschool zal het komende schooljaar beginnen met een 1:1 iPad initiatief. Mr. Nowak heeft besloten om zijn leerlingen van de tweede klas hun iPad te laten gebruiken om het weer (temperatuur, neerslag en wind) voor een week te voorspellen. Elke leerling maakt een tekening van hoe zij denken dat het weer eruit zal zien. Sara, een leerling, wilde ook de temperaturen bijhouden die iedereen voorspeld had. Mr. Nowak startte een Google-spreadsheet waarin elke leerling zijn voorspelde temperaturen kon invoeren. De volgende dag registreerden ze het werkelijke weer door de Accuweather App op hun iPad te gebruiken en de informatie in te voeren in het Google-blad. Olivia wilde ook de werkelijke temperatuur noteren in Sara's spreadsheet, zodat ze konden vergelijken hoe hun voorspellingen zich verhielden tot wat het weer in werkelijkheid was. Na een week projecteerden ze de Google spreadsheet op het smartboard en berekende de verschillen tussen de waargenomen en voorspelde temperaturen. Mr. Nowak demonstreerde hoe je een staafdiagram maakt van die verschillen.

Scenario 2:

Alle klassen van de tweede klas gaan op excursie! De schoolkantine heeft voor iedereen PB&J-lunches ingepakt in identieke papieren zakjes, behalve voor Sara en Olivia die allergisch zijn voor pinda's. De papieren lunchzakjes zijn gelabeld met de namen van alle leerlingen en verdeeld in 10 dozen met 10 lunches per doos. De lunches zijn op alfabetische volgorde van achternaam in de dozen gedaan. Mr. Nowak wil controleren of Sara en Olivia pinda-vrije lunches krijgen. Ze helpen hem met het doorzoeken van de dozen. Olivia Velazquez weet dat haar lunch waarschijnlijk aan het eind zal zijn, dus kijkt ze naar de eerste lunch in elke doos totdat ze er een vindt die begint met een letter dicht bij het eind van het alfabet. Als ze de doos vindt die begint met Jemal Summer's lunch, kijkt ze naar de laatste lunch in die doos. Het is die van Billy Wagner, dus ze weet dat ze er dichtbij moet zijn! Ze kijkt naar de lunch naast die van Billy, en die is van haar. Gelukkig heeft de cafetaria haar een boterham met kaas en wortels meegegeven.

Het karakteriseren van elementen van computationeel denken

De stappen in het computationeel oplossen van problemen kunnen in het algemeen als volgt worden beschreven.

1. decontextualisering: vertaling van het probleem of de vraag in een bepaald vakdomein ("context") in computationele termen;
2. computationeel oplossen van problemen: het construeren van een uitvoerbare oplossing;
3. (her)contextualisering: de oplossing terugvertalen naar het vakdomein.

Definities van Computationeel Denken variëren in de manier waarop ze de activiteiten benadrukken die worden uitgevoerd in de stappen (1), (2) en (3). Selby en Woollard (2013, p. 5), bijvoorbeeld, beschrijven computationeel denken als "een activiteit, vaak productgericht, geassocieerd met, maar niet beperkt tot, probleemoplossing. Het is een cognitief of denkproces dat de volgende elementen weerspiegelt

- het vermogen om in abstracties te denken,
- het vermogen om te denken in termen van decompositie,
- het vermogen om algoritmisch te denken,
- het vermogen om te denken in termen van evaluaties,
- het vermogen om in generalisaties te denken."

We kunnen de bovenstaande elementen globaal in verband brengen met de stappen in ons model. De eerste twee elementen hebben vooral te maken met stap (1): het analyseren van patronen binnen een probleem of situatie (abstractie), en het opdelen van een probleem in kleinere deelproblemen (decompositie). Algoritmisch denken wordt vooral gebruikt binnen stap (2), terwijl het evalueren van een oplossing en het onderzoeken hoe deze kan worden gegeneraliseerd de computationele oplossing verbindt met het vakdomein, hetgeen plaatsvindt in stap (3). We kunnen dit in een tabel samenvatten:

	(1)	(2)	(3)
Selby & Woollard (2013)	abstractie decompositie	algoritmisch denken	evaluatie generalisatie

Opdracht in tweetallen

Drie andere bekende operationaliseringen vindt u in het document "B. Definities van CT". Breid de bovenstaande tabel uit met drie extra rijen. Categoriseer de elementen die je in de drie kolommen vindt. Herkent u elementen in de klassenscenario's 1 en 2?

Conclusie

Algemene Discussie Opdracht

Vergelijk je conclusies met medeleerlingen in de klas. Stel samen een korte praktische definitie op van computationeel denken in termen van stappen en activiteiten.

Leermiddelen

A. Korte inleiding tot CT

B. Definities van CT

Referenties

Selby, C. & Woollard, J. (2013) Computational thinking: the developing definition, available via internet: http://eprints.soton.ac.uk/356481, accessed: 10 February 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.

Sectie 2: Hulpmiddelen voor computationeel denken

In deze sectie krijg je praktische ervaring in computationeel denken leeractiviteiten met behulp van eenvoudige maar krachtige computationele digitale hulpmiddelen bedoeld voor echte probleemoplossingstoepassingen in verschillende disciplines - geen extra programmering is vereist voor dit moment.

De sectie introduceert drie hulpmiddelen, namelijk NetLogo, Microsoft Excel en Google Ngrams, en presenteert activiteiten die laten zien hoe deze hulpmiddelen kunnen worden gebruikt om computationeel denken in verschillende schoolvakken te integreren. De beschrijving van elk hulpmiddel en de toepassingen ervan in verschillende disciplines worden in detail gegeven in de respectieve subsecties. Terwijl u deze activiteiten doorloopt, krijgt u ervaring met computationeel denken-aspecten, -praktijken en -hulpmiddelen en begrijpt u de rol van computationeel denken binnen de verschillende disciplines.

Bijdrage aan de leerresultaten

Leerresultaten
• Computationeel denken in de praktijk brengen en toepassen aan de hand van een reeks voorbeelden van case-scenario's • Classificeren van problemen als computationeel oplosbaar • Computationele hulpmiddelen gebruiken om problemen op te lossen • Gedetailleerde stap-voor-stap oplossingen voor problemen ontwikkelen • Nadenken over gegevens en het interpreteren en visualiseren hiervan • Gebruik maken van gegevensanalyse om het inzicht in natuurlijke systemen te vergroten • Evalueren welke soorten problemen kunnen worden opgelost met behulp van modellering en simulatie • Begrijpen en beschrijven hoe modellering en simulatie kunnen worden gebruikt om een probleem op te lossen • Analyseren van gegevens en identificeren van patronen door modellering en simulatie. • Samenwerken en communiceren met collega's om een probleem op te lossen

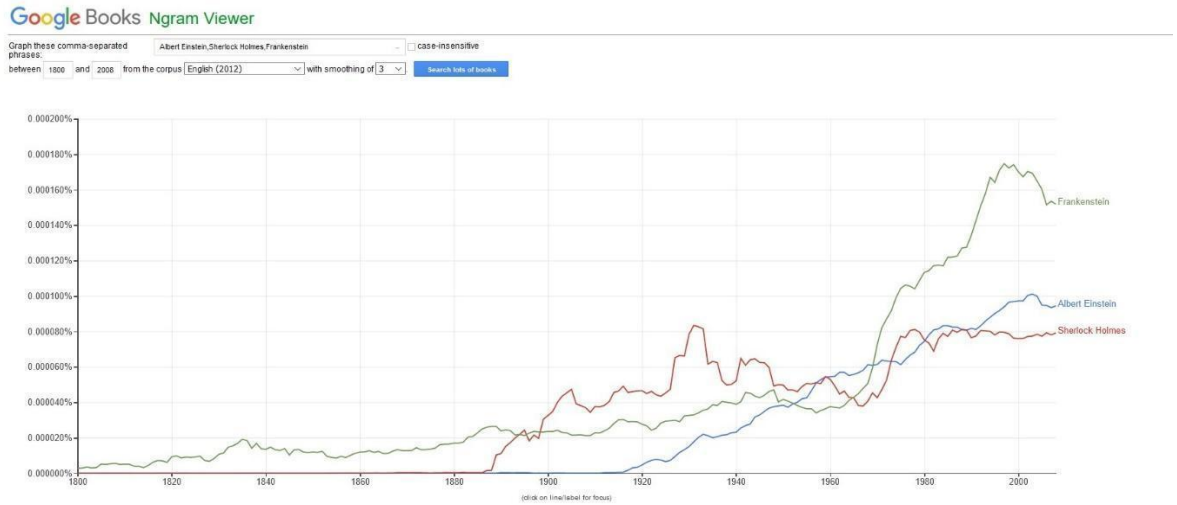
Activiteit 2.1: Google Ngrams gebruiken

Totale tijd: 2 - 3 uur

De Google Ngram viewer is een grafisch hulpmiddel dat woordfrequenties uit een groot corpus van boeken weergeeft, en het gebruik van een term of een zinsdeel in de loop van de tijd visualiseert. De tool gebruikt literatuurbronnen die tussen 1500 en 2008 gedrukt zijn in Amerikaans Engels, Brits Engels, Frans, Duits, Italiaans, Spaans, Russisch, Hebreeuws en Chinees, en kan dus gebruikt worden om veranderingen in taalgebruik door de jaren heen te onderzoeken. Fluctuaties in taalgebruik kunnen sociaal-culturele veranderingen uitbeelden en zo inzicht verschaffen in hoe verschillende sociaal-culturele transformaties zich door de geschiedenis heen ontwikkelen. In deze activiteit gaan we Google Ngrams gebruiken om culturele en sociale veranderingen door de tijd heen te onderzoeken, zoals ze worden weerspiegeld door het gebruik van specifieke termen in boeken.

Warming-up discussie

Voor je aan de slag gaat met deze activiteit, klik je op de volgende link om toegang te krijgen tot de Google Ngram viewer (https://books.google.com/ngrams). Je zult merken dat het programma al een voorbeeld voorstelt dat in de volgende figuur wordt afgebeeld. In dit voorbeeld zien we de resultaten voor drie zinnen, "Albert Einstein", "Sherlock Holmes", en "Frankenstein". Het hulpmiddel identificeert de frequentie van deze zinnen zoals ze voorkomen in boeken tussen 1800-2008. Zoals u kunt zien, komt de term Frankenstein al sinds het begin van de 19e eeuw voor in de literatuur, in vergelijking met de andere twee termen die later in de literatuur verschijnen.



Figuur 1 Google Ngram Voorbeeld voor de termen: "Albert Einstein, Sherlock Holmes, en Frankenstein".

Discussie opdracht

Welke andere trends zie je met betrekking tot de frequentie van de termen in figuur 1?

Subsectie 1: Probleemstelling

De vraag die we in deze activiteit zullen onderzoeken heeft betrekking op culturele veranderingen in de Verenigde Staten gedurende de laatste twee eeuwen. De sociaal-culturele theoretische lens die we gaan gebruiken wordt beschreven in de volgende paragraaf, gevolgd door onze vraag.

De theorie van sociale verandering en menselijke ontwikkeling voorspelt een wereldwijde verschuiving van *Gemeinschaft* naar *Gesellschaft* op basis van sociaal-demografische veranderingen. Deze twee woorden werden in 1887 geïntroduceerd door Toennies en zijn gebruikt voor het bestuderen van sociale veranderingen. Gemeinschaft beschrijft bindende, primaire interactionele relaties gebaseerd op sentiment en wordt gekenmerkt door een landelijke omgeving, eenvoudige face-to-face relaties, een laag technologieniveau, beperkte scholing, en door behoefte in plaats van rijkdom. Gesellschaft daarentegen beschrijft een interactioneel systeem dat wordt gekenmerkt door eigenbelang, concurrentie en onderhandelde accommodatie en wordt gekenmerkt door een stedelijke omgeving, een gemoderniseerde samenleving met hoge niveaus van technologie en rijkdom' (bron: Younes en Reips, 2018, p.1; Christenson, 1984, p.160)

In de afgelopen eeuwen is het tempo van de verstedelijking drastisch en wereldwijd toegenomen. In deze casestudy onderzoeken we of deze verandering overeenkomt met een beweging van *Gemeinschaft* naar *Gesellschaft* systeem. Meer specifiek zullen we de volgende vraag onderzoeken:

"Zijn de Verenigde Staten de laatste twee eeuwen van Gemeinschaft naar Gesellschaft geëvolueerd en stemt dit overeen met de overgang van een boerse naar een stedelijke samenleving?"

Om deze vraag te beantwoorden, gaan we patronen van sociaal-culturele veranderingen onderzoeken die weerspiegeld worden in woordfrequenties van de laatste twee eeuwen in de Verenigde Staten. De Google Ngram viewer is hiervoor bijzonder nuttig, omdat culturele waarden worden weerspiegeld in woorden/termen die in geschriften worden gebruikt, en wij dus bewijs kunnen leveren voor de culturele verandering in waarden op lange termijn door de woordfrequenties in boeken te onderzoeken.

Subsectie 2: Het selecteren van zoektermen

De eerste stap in het beantwoorden van onze vraag is het identificeren van geschikte zoektermen die gekoppeld zijn aan de begrippen Gemeinschaft en Gesellschaft en die de culturele veranderingen in waarden in de Verenigde Staten kunnen weergeven. In dit specifieke voorbeeld zijn de volgende criteria van belang voor het selecteren van

representatieve zoektermen: a. een hoge frequentie van de term en b. de term heeft weinig verschillende betekenissen.

Discussie opdracht

Waarom denk je dat de bovenstaande criteria belangrijk zijn bij deze activiteit?

*

Het selecteren van termen met een hoge frequentie is belangrijk zodat de grafieklijnen daadwerkelijk culturele veranderingen door de tijd heen kunnen weergeven en omdat woorden met een hoge frequentie kenmerken zijn van de cultuur van een land. Even belangrijk in dit voorbeeld is het selecteren van woorden met weinig verschillende betekenissen. Woorden met uiteenlopende betekenissen kunnen immers in verschillende contexten worden gebruikt die niet altijd relevant zijn voor culturele waarden. In de volgende tabel ziet u woorden die aan de bovenstaande criteria voldoen en het systeem Gemeinschaft en Gesellschaft weerspiegelen.

Tabel 1: Termen in verband met Gemeinschaft en Gesellschaft richting

Gemeinschaft	verplicht, plicht, geven, genade, doen, daad
Gesellschaft	kiezen, besluit, krijgen, ontvangst, voelen, emotie

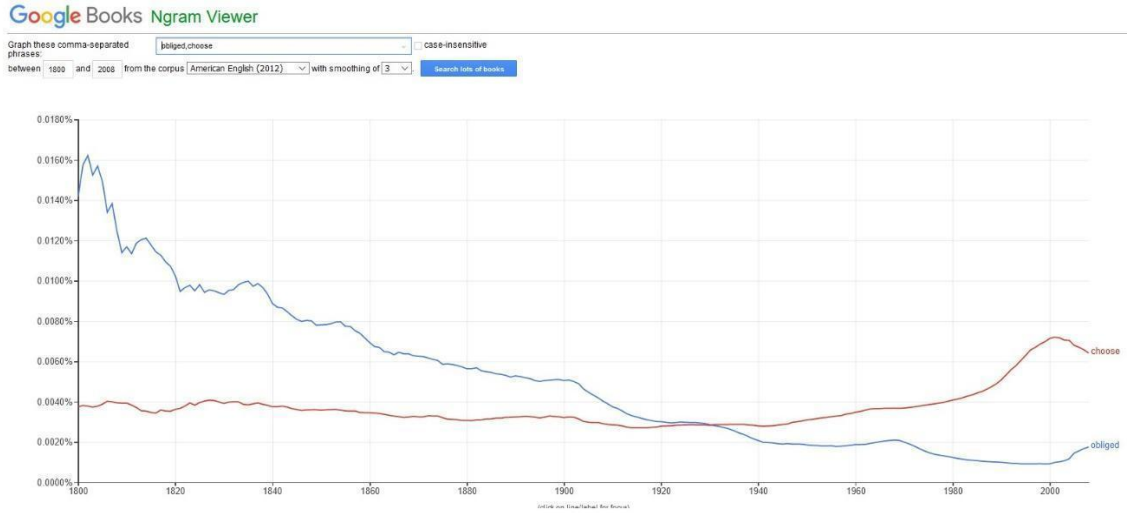
Discussie opdracht

Hoe kunnen we de bovenstaande lijst en de Google Ngram viewer gebruiken om onze vraag te beantwoorden?

We hebben besproken dat Google Ngram kan worden gebruikt om frequenties van woorden in de literatuur door de tijd heen weer te geven. Voor ons voorbeeld betekent dit dat we termen kunnen gebruiken en vergelijken die tegengestelde denkwijzen, karaktertrekken en eigenschappen weerspiegelen en kunnen observeren hoe hun frequentie variëren door de tijd heen.

Discussie opdracht

De volgende figuur toont de frequentie van de woorden verplicht (gekoppeld aan het Gemeinschaft-systeem) en kiezen (gekoppeld aan het Gesellschaft-systeem). Wat valt je op aan de frequentie van deze twee termen in de loop der tijd?



Subsectie 3: Vergelijking van termen en Interpretatie

In deze subsectie gaan we Google Ngrams gebruiken om meer termen te vergelijken die verband houden met Gemeinschaft en Gesellschaft en om te observeren hoe hun frequentie is veranderd in de afgelopen twee eeuwen.

Groepsopdracht

Klik op de volgende link om toegang te krijgen tot Google Ngrams (https://books.google.com/ngrams).

1. Gebruik het zoekveld (wis zoektermen die al in het veld staan) en typ de volgende twee woorden, gescheiden door een komma: Plicht, Besluit.
2. Selecteer uit het drop-down menu "from the corpus" American English (2012) en klik vervolgens op de knop "search lots of books".
3. Wat valt je op aan de frequentie van deze twee termen in de loop der tijd?
4. Onderzoek op dezelfde manier de frequentie van de volgende woorden (kies er ten minste twee):
    a. Geven en krijgen,
    b. genade en ontvangst,
    c. Doen en voelen,
    d. Daad en Emotie
5. Hoe variëren de frequenties van deze woorden?

Bevestigen uw waarnemingen de overgang van Gemeinschaft naar Gesellschaft gedurende de laatste twee eeuwen?

*

Inderdaad, zoals voorspeld door de theorie van sociale verandering en menselijke ontwikkeling, zijn termen als verplicht, plicht, geven doen en daad, die allemaal een plattelands- en een Gemeinschaft-milieu weerspiegelen, in de afgelopen twee eeuwen

verzwakt. In dezelfde periode zijn termen als kiezen, besluit, krijgen, ontvangst, gevoel en emotie, die allemaal een stedelijke en een Gesellschaft-omgeving weerspiegelen, toegenomen.

Generalisatie (optioneel)

Bij deze activiteit bleek duidelijk dat, naarmate de Verenigde Staten zich in de richting van de Gesellschaft bewogen, de culturele kenmerken van de Gesellschaft (weerspiegeld door relevante woorden in het corpus van Amerikaanse boeken) een kwantitatieve toename vertoonden, terwijl de culturele kenmerken van de Gemeinschaft (weerspiegeld door relevante woorden in het corpus van Amerikaanse boeken) een kwantitatieve afname vertoonden. Kunnen deze verbanden en bewegingen worden gegeneraliseerd naar andere delen van de wereld, zoals de theorie van sociale verandering en menselijke ontwikkeling zou voorspellen?

Groepsopdracht

Herhaal de in subsectie 3 beschreven stappen; selecteer deze keer de literatuur van uw keuze (bijv. Brits of Duits) om dezelfde vraag te beantwoorden voor het land van uw keuze.

Opmerking: Voor Duitse literatuur kunt u de volgende termen gebruiken (bron: Younes & Reips, 2018):

1. Versprechen und auswählen
2. Pflicht en Entscheidung
3. Geben en bekommen
4. Güte en Kauf
5. Handeln und spüren
6. Handlung en Emotie

Leermiddelen

A Van Gemeinschaft naar Gesellschaft.pdf (optioneel)

B De veranderende psychologie van de cultuur van 1800 tot 2000 (optioneel)

C De veranderende psychologie van de cultuur in de Duitstalige landen (optioneel)

Referenties

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

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

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

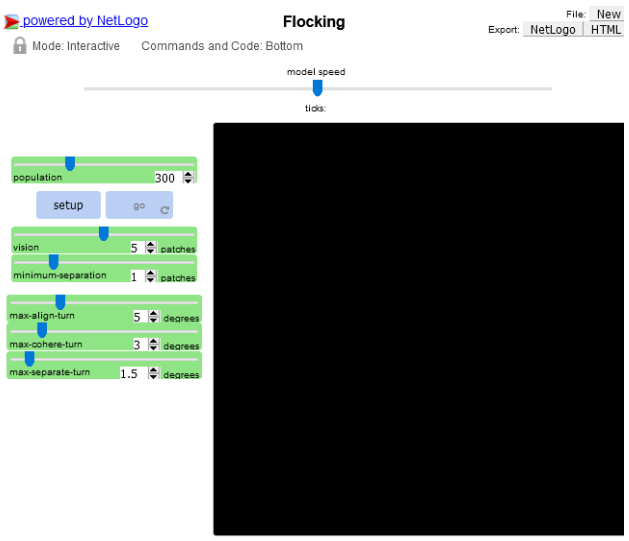
Activiteit 2.2 Modelleren en simuleren met NetLogo

Totale tijd: 2.30 - 3 uur

NetLogo is een multi-agent programmeerbare modelleeromgeving om natuurlijke en sociale fenomenen te simuleren en te demonstreren hoe deze zich in de tijd ontwikkelen. Met deze tool kun je een wereld creëren die bestaat uit rechthoeken of andere stukken en kun je agenten (of schildpadden) parametriseren die zich verplaatsen en met elkaar en hun omgeving communiceren. In deze activiteit gaan we NetLogo gebruiken voor een casestudie over epidemiologie en modelsimulaties uitvoeren om te onderzoeken hoe grote en kleine veranderingen een omgeving kunnen beïnvloeden.

Warming-up discussie

Klik, voordat je met deze activiteit aan de slag gaat, op de volgende link om toegang te krijgen tot het hulpprogramma http://www.netlogoweb.org/launch#http://www.netlogoweb.org/assets/modelslib/Sample%20Models/Art/Fireworks.nlogo. Kies in het uitklapmenu " Search the Models Library " de optie sample models/Biology/Flocking. Het voorbeeld dat wordt geladen, is afgebeeld in de volgende figuur.



Figuur 1 Flocking voorbeeld in NetLogo

Dit model is een poging om het samen vliegen van vogels in zwermen na te bootsen. Elke vogel volgt precies dezelfde set regels: "uitlijnen", "seperatie", en "cohesie". "Uitlijnen", gecontroleerd door de max-align-turn slider, betekent dat een vogel de neiging heeft om zo te draaien dat hij in dezelfde richting beweegt als vogels in de buurt. "Separatie", geregeld door de max-separatiion-turn slider, betekent dat een vogel zal draaien om een andere vogel te ontwijken die te dichtbij komt. "Cohesie", gecontroleerd door de max-coherence-turn slider, betekent dat een vogel naar andere vogels in de buurt zal bewegen (tenzij een andere vogel te dichtbij is). Wanneer twee vogels te dicht bij elkaar zijn, wordt de "separatie" regel de belangrijkst regel en worden de andere twee gedeactiveerd totdat de minimale separatie is bereikt. De zichtregelaar is de afstand die elke vogel 360 graden rondom zich kan zien.

De groene sliders aan de linkerkant (b.v. max-fireworks) worden gebruikt om uw model te parametriseren. De knop SETUP, stelt het model in volgens de waarden aangegeven door alle sliders. De knop GO voert het model uit.

Opdracht in tweetallen

Bepaal eerst het aantal vogels in de simulatie en zet de populatie-slider op die waarde. Druk eerst op SETUP en dan op GO om de simulatie te starten (De standaardinstellingen voor de sliders zullen een redelijk goed gedrag opleveren. Je kunt er echter mee spelen om variaties te krijgen). Pas de sliders aan om te zien hoe u strakkere zwerm, lossere zwerm, minder zwermen of meer zwermen kunt krijgen.

Subsectie 1: Probleemstelling

De vraag die wij in deze activiteit zullen onderzoeken heeft betrekking op de epidemiologie en in het bijzonder op het Zika-virus.

Het Zika-virus is een door muggen overgebracht flavivirus dat in 1947 voor het eerst in Uganda bij apen werd vastgesteld. De incubatieperiode (de tijd tussen blootstelling en symptomen) van de virusziekte wordt geschat op 3-14 dagen. De meerderheid van de mensen die met het Zika-virus besmet zijn, ontwikkelt geen symptomen. De symptomen zijn over het algemeen mild en bestaan uit koorts, huiduitslag, bindvliesontsteking, spier- en gewrichtspijn, malaise en hoofdpijn, en duren meestal 2-7 dagen. Het virus wordt hoofdzakelijk overgedragen door de beet van een besmette mug uit het Aedes-geslacht, voornamelijk *Aedes aegypti*, in tropische en subtropische gebieden. Aedes-muggen bijten gewoonlijk overdag, met een piek in de vroege ochtend en de late namiddag/avond. Dit is dezelfde mug die dengue, chikungunya en gele koorts overbrengt. Het virus wordt ook overgedragen van moeder op foetus tijdens de zwangerschap, door seksueel contact, transfusie van bloed en bloedproducten, en orgaantransplantatie (bron: Wereldgezondheidsorganisatie [1]).

In deze casestudie gaan we werken aan een hypothetisch epidemiologisch scenario en proberen we het risico van verspreiding van het virus te begrijpen en in te schatten. Het scenario waar we aan gaan werken is het volgende:

---

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

In een kleine stad in Zuid-Afrika zijn twee mensen positief getest op het Zika-virus. De regering wil voorspellen hoe ernstig de situatie is en zo de nodige preventiemaatregelen nemen. De totale bevolking van deze stad bedraagt 800 en na een eerste schatting is het aantal muggen in het gebied 1040, terwijl het aantal predatoren in het gebied 21 bedraagt.

De vraag die wij moeten beantwoorden is de volgende:

Hoeveel mensen zullen worden besmet en wat zou de beste strategie zijn om het virus te voorkomen?

Om deze vraag te beantwoorden, gaan we NetLogo gebruiken en simulaties uitvoeren op een model over het Zika-virus.

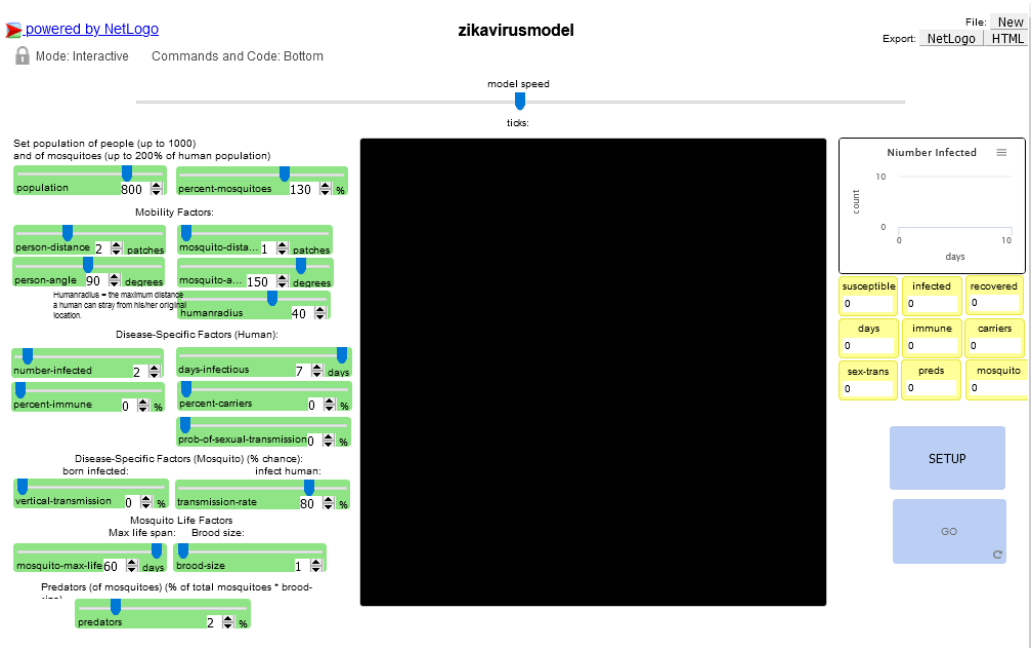
Subsectie 2: Simulatie

In deze subsectie gaan wij werken met een reeds geïmplementeerd model dat ons zal helpen begrijpen hoe modellering en simulatie kunnen worden gebruikt om nieuwe inzichten en kennis te genereren over een verschijnsel en hoe zij de besluitvorming kunnen beïnvloeden.

Klik op de volgende link voor toegang tot NetLogo:

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

Klik op "browse" om het model zikavirusmodel.nlogo te uploaden. Zodra het model succesvol is geladen, zou het volgende voorbeeld te zien moeten zijn.



Figuur 2 Zika-virus in NetLogo

In het model zouden de muggen met Zika besmette mensen bijten en vervolgens de ziekte verspreiden door andere, niet besmette mensen te bijten. Er zijn veel variabelen die je in dit model kunt manipuleren. Voor deze activiteit zullen we ons alleen richten op de volgende:

- *de persoonsverplaatsing (person-distance)*, die de mobiliteit van de mens weergeeft - hoe ver hij kan draaien en hoeveel vierkanten (vakken) hij op een dag kan afleggen
- *aantal geïnfecteerden (number-infected)*, dat het aantal aanvankelijk geïnfecteerden weergeeft
- *dagen-infectieus (days-infectious)*, dat is de tijd dat een gemiddeld persoon besmet is
- *percentage-immuun (percent-immune)* verwijst naar het percentage van de bevolking dat immuun is (= gevaccineerd)
- *roofdieren (predators)* die het aantal roofdieren weergeeft
- *percent-muggen (percent0mosquitoes)* die het aantal muggen weergeeft

Voor elke simulatie van het model moet u achtereenvolgens de volgende drie dingen doen: ten eerste stelt u de sliders in (voor het geval u ze wilt aanpassen), ten tweede klikt u op SETUP (in de rechterbenedenhoek van het scherm) en ten derde klikt u op GO om het model uit te voeren.

De grafiek in de rechterbovenhoek en de monitoren eronder vertellen u hoe het de samenleving vergaat ten opzichte van het virus onder uw sliders instellingen.

Opdracht in tweetallen

Start de simulator met de standaardwaarden die overeenkomen met de in subsectie 1 gepresenteerde probleemstelling. Beantwoord, zodra de simulatie is afgelopen, de volgende vraag: Hoeveel mensen werden in totaal besmet en in hoeveel dagen?

*

Door de simulator uit te voeren, krijgt u een schatting van het aantal mensen dat onder bepaalde omstandigheden met het virus zal worden besmet. In dit voorbeeld ligt dat aantal ergens tussen 740 en 775. Met ons model hebben we dus een deel van onze vraag over het aantal mensen dat met het Zika-virus besmet raakt, kunnen beantwoorden. Wat we nog niet hebben beantwoord is welke strategie het beste zou zijn om de verspreiding te voorkomen. Dit is de focus van de volgende subsectie.

Subsectie 3: Verder onderzoek - Interpretatie

Er zijn verschillende variabelen in dit model die kunnen worden aangepast om na te gaan hoe de ziekte zich onder de bevolking zou verspreiden. Om na te gaan wat de beste strategie is om de verspreiding tegen te gaan, zullen wij drie variabelen in aanmerking nemen: roofdieren (predators), persoonsverplaatsing (person-distance), en percentage immuun (percent-immune).

Groepsopdracht

In deze opdracht gaan we onderzoeken welke invloed de volgende veranderingen hebben op de verspreiding van het virus en welke daarvan het doeltreffendst is.

1. Verhoog de roofdieren (predators) met 6%: Verander de waarde van de predators in 6%, en klik dan op SETUP en dan GO. Houd een notitie bij van het aantal teruggekregen roofdieren in de monitoren in de rechterbovenhoek. Voordat je verder gaat met nummer 2 hieronder, zet je de roofdieren-slider terug op 2% (standaardwaarde).
2. Verhoog het aantal immuun (percent-immune) tot 6%: Verander de waarde van immuun naar 6%, en klik dan op SETUP en dan GO. Houd een notitie bij van het aantal herstelden in de monitors in de rechterbovenhoek. Voordat u doorgaat naar nummer 3 hieronder, zet u de immuun-slider weer op 0 (standaardwaarde).
3. Verlaag de persoonsverplaatsing (person-distance) tot 1 vakje: Verander de waarde van persoonsverplaatsing naar 1%, en klik dan op SETUP en dan GO. Noteer het aantal herstelde personen op de monitors in de rechterbovenhoek.

*

U zult merken dat als u de persoonsverplaatsing met 1 stukje vermindert, het aantal besmette mensen ergens tussen 400-680 ligt. Als u de roofdieren met 6% verhoogt, ligt het aantal besmette mensen tussen 140 en 300, terwijl door het percentage immune mensen met 6% te verhogen, het aantal besmette mensen tussen 630 en 740 ligt. Onder de huidige omstandigheden die in ons model zijn vastgesteld, is de beste strategie om te voorkomen dat de ziekte zich onder de gehele bevolking verspreidt, dus een verhoging van het aantal roofdieren met 6%.

Subsectie 4: Afrondingsvraag

In deze activiteit hebben wij gewerkt aan een hypothetisch scenario in epidemiologie en hebben wij met behulp van modellering en simulatie de dynamiek van een virusverspreiding onderzocht door variabelen te manipuleren en de resultaten te evalueren.

Discussie opdracht

Hoe kunnen modellering en simulatie de besluitvorming in andere disciplines dan de epidemiologie beïnvloeden?

Leermiddelen

zikavirusmodel.nlogo

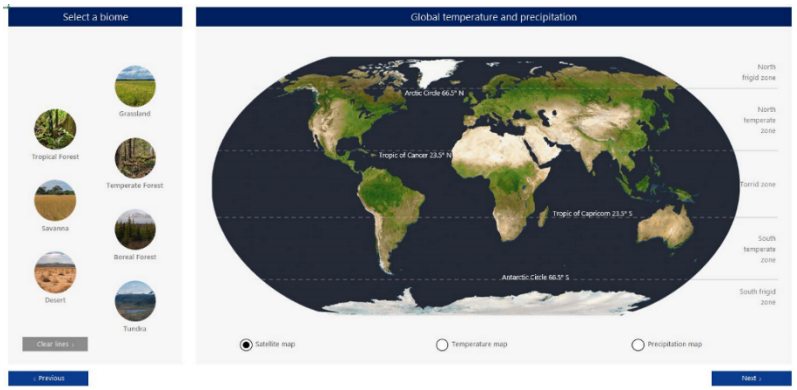
Activiteit 2.3 Microsoft Excel gebruiken om gegevens te manipuleren en weer te geven

Totale tijd: 2.30 - 3 uur

Excel is een spreadsheet-hulpmiddel voor het ordenen en uitvoeren van berekeningen op gegevens, en het analyseren en weergeven van gegevens als een grafiek of een diagram. In deze activiteit gaan we Excel gebruiken voor het analyseren en weergeven van gegevens, het vaststellen van trends die in de grafieken worden weergegeven en het in verband brengen van informatie om een vraag te beantwoorden over klimaatverschillen in biomen en hoe deze van invloed zijn op hun oppervlaktekleuren.

Warming-up discussie

Voordat u met deze activiteit begint, opent u het werkboek over de kleuren van de aarde en navigeert u naar het werkblad Mapping biomes om een afbeelding van de oppervlakken van de aarde te zien die is gemaakt met satellietbeelden (afbeelding 1).



Figuur 1 Werkblad Mapping biomes brengen

Opdracht in tweetallen

Klik op de bioomknoppen links op het satellietbeeld van de aarde om de locaties van de verschillende bioomgebieden te markeren.

- Welke kleur beschrijft volgens jou het best elke bioom?
- Wat verklaart de verschillende kleuren voor de verschillende biomen?

Tip: selecteer de knop Temperatuur- en neerslagkaart (Temperature map en Precipitation map) onder het satellietbeeld om verbanden te leggen tussen de kleuren en de fysieke omstandigheden.

*

Zoals u merkt, zijn regio's met veel neerslag groener, terwijl hogere temperaturen en minder neerslag overeenkomen met regio's die bruiner kunnen lijken. De polen hebben zeer lage gemiddelde temperaturen, wat de witte kleur in verband met sneeuw en ijs verklaart.

Subsectie 1: Probleemstelling

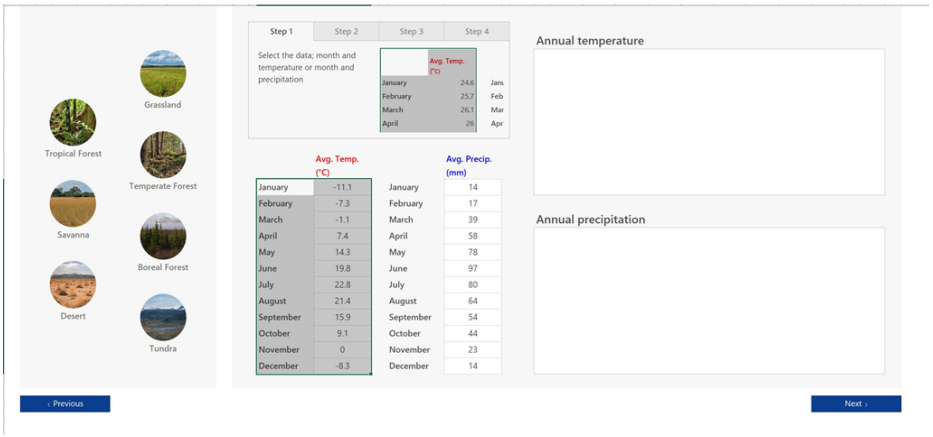
In deze activiteit gaan we gegevens gebruiken als basis om beter te begrijpen waarom kleurveranderingen optreden in biomen. We zullen de zeven aardse biomen onderzoeken en hun karakteristieke kleuren gezien vanuit de ruimte bekijken, die verband houden met factoren als temperatuur en neerslag. De vraag die we zullen behandelen is de volgende:

Hoe verandert de temperatuur en de neerslag in de verschillende biomen in de loop van het jaar en hoe wordt deze verandering weerspiegeld in de kleuren?

Subsectie 2: Grafieken van temperatuur en neerslag

In de warming-up discussie heb je gekeken naar de gemiddelde temperatuur en neerslag op aarde en ben je begonnen verbanden te leggen tussen de kleuren van de aardoppervlakken die op het satellietbeeld te zien zijn en de gemiddelde jaarlijkse neerslag en temperatuur. In deze subsectie gaan we de maandelijkse veranderingen in neerslag en temperatuur onderzoeken die zich in de loop van het jaar in de biomen voordoen.

Navigeer voor de volgende opdracht naar het werkblad "Create a biome climate chart" dat is afgebeeld in de volgende figuur.



Figuur 2 Werkblad "Create a biome climate chart"

Opdracht in tweetallen

Voor deze taak kies je één bioom dat je verder wilt onderzoeken. Als je op de knoppen links van de bioom klikt, zie je de maandelijkse temperatuur- en neerslaggegevens onder de labels "Avg. Temp" en "Avg. Precip." labels.

1. Selecteer de maandtemperatuurgegevens en klik vervolgens op de optie invoegen in de menubalk van Excel. Klik op de optie aanbevolen grafieken en selecteer vervolgens een grafiek die u wilt gebruiken voor de weergave van de maandelijkse temperatuurgegevens van de bioom.
2. Herhaal dezelfde stappen voor de neerslaggegevens

Hoe variëren de temperatuur- en neerslaggegevens in de loop van het jaar voor jouw bioom? Kijk welke maand(en) de hoogste en laagste temperatuur en neerslag heeft.

Kies een ander bioom en herhaal deze taak door een grafiek te maken van de maandelijkse temperatuur en neerslag. Vergelijk de twee biomen door te observeren welke meer of minder schommelingen in temperatuur en neerslag gedurende het jaar vertoont.

Optionele opdracht

Herhaal de bovenstaande taak voor alle biomen en kies verschillende soorten grafieken. Kijk welke diagrammen het nuttigst zijn om de veranderingen in temperatuur en neerslag weer te geven en om trends, verschillen en overeenkomsten tussen twee of meer biomen te onderscheiden.

*

Op dit punt hebben we het eerste deel van onze vraag over de manier waarop temperatuur en neerslag veranderen in de loop van een jaar voor een bioom behandeld. Door gebruik te maken van grafieken om de temperatuur- en neerslaggegevens voor een bioom weer te geven, waren we in staat deze schommelingen waar te nemen en ook biomen met elkaar te vergelijken. Wat we echter verder moeten onderzoeken, is hoe deze veranderingen in temperatuur- en neerslaggegevens worden weergegeven door middel van kleuren.

Subsectie 3: Vergelijking van kleuren met temperatuur en neerslag in biomen

In de vorige subsectie hebben we grafieken gebruikt om de maandelijkse veranderingen in temperatuur- en neerslaggegevens voor twee biomen weer te geven en te vergelijken. In deze subsectie gaan we bioomkenmerken, temperatuur- en neerslaggegevens met elkaar in verband brengen en onderzoeken hoe veranderingen in de seizoensbegroeiing worden weerspiegeld door de meest opvallende kleur die in een bepaalde maand van het jaar in het landschap van een bioom te zien is.

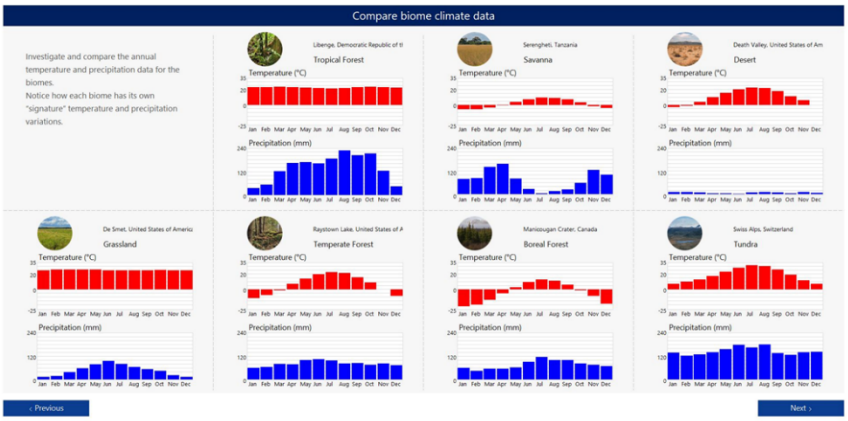
Opdracht in tweetallen

Navigeer naar het werkblad "Information of biomes" dat in de volgende figuur is afgebeeld. Selecteer opnieuw de biomen die u in de vorige subsectie hebt gekozen en lees de verstrekte informatie over hun begroeiing. Wat valt u op betrekkende de temperatuur, de neerslag en de begroeiing van een biome?

Figuur 3 werkblad "Information of biomes"

Wat u zult opvallen door de temperatuur, neerslag en begroeiing van een bioom te vergelijken, is dat de begroeiing van een bepaalde regio specifiek geschikt is voor de klimaatomstandigheden van de regio van de bioom. Je zult ook merken dat, ook al zijn de temperaturen van de ene regio vergelijkbaar met die van de andere, verschillen in neerslag een grote invloed hebben op het type begroeiing in een bioom en dus op de oppervlaktekleuren ervan.

Navigeer voor de volgende vraag naar het werkblad "*Compare biome climate data*" dat in de volgende figuur is afgebeeld.



Figuur 4 Werkblad " Compare biome climate data"

Opdracht in tweetallen

Gebaseerd op de temperatuur- en neerslaggegevens,

- Welke bioomregio's hebben volgens u de meest consistente jaarlijkse oppervlaktekleuren?
- Welke zal volgens u de grootste schommeling in oppervlaktekleuren vertonen gedurende het kalenderjaar?

Leg je redenering uit.

*

Kleur kan worden gebruikt om de veranderingen van een bioom in de loop van het jaar te beschrijven; de veranderende begroeiing van een bioom is verantwoordelijk voor de kleur van het oppervlak en wordt weerspiegeld door de meest opvallende kleur die in een bepaalde maand van het jaar in het landschap van de bioom te zien is.

Navigeer voor de volgende vraag naar het werkblad "Biome colors through the year" dat in de volgende figuur is afgebeeld.

Figuur 5 werkblad "Biome colors through the year"

Opdracht in tweetallen

Gebaseerd op de temperatuur- en neerslaggegevens,

- Welke bioomregio's hebben volgens u de meest consistente jaarlijkse oppervlaktekleuren?
- Welke zal volgens u de grootste schommeling in oppervlaktekleuren vertonen gedurende het kalenderjaar?

Leg je redenering uit.

*

Navigeer voor de volgende vraag naar het werkblad "Compare biome color data" dat in de volgende figuur is afgebeeld.

Figuur 6 Werkblad " Compare biome color data"

Opdracht in tweetallen

Hoe zijn de maandelijkse prominente kleuren voor de verschillende bioomregio's gerelateerd aan temperatuur en neerslag? Geef voorbeelden uit de gegevens.

Tip: Merk op hoe de verschillende kleuren in de verschillende biomen verband houden met de neerslag en de temperatuur.

*

Het is u misschien opgevallen dat de groenblauwe kleur staat voor gebieden met sneeuw, terwijl gebieden met veel neerslag groener zijn, en dat hogere temperaturen en minder neerslag overeenkomen met gebieden die bruiner lijken.

Bij het beantwoorden van het tweede deel van onze vraag - *hoe veranderingen in neerslag en temperatuur worden weerspiegeld in kleuren* - stelden wij vast dat de veranderende begroeiing van een bioom, als gevolg van veranderingen in temperatuur en neerslag, wordt weerspiegeld in de meest prominente kleur die in een bepaalde maand van het jaar in het landschap van het bioom te zien is, en dus de oppervlaktekleur van het bioom bepaalt.

Subsectie 4: Voorspellen van klimaatverandering - Uitgebreide activiteit (optioneel)

De taken en vragen in de vorige subsecties leidden u naar het beantwoorden van de vraag over hoe de temperatuur en neerslag in verschillende biomen door het jaar heen veranderen en hoe deze verandering wordt weerspiegeld in kleuren. In deze sectie gaat u onderzoeken

hoe een probleem van klimaatverandering een bioom regio van uw keuze beïnvloedt. Daarvoor ga je neerslag- en temperatuurgegevens voor dit bioom verzamelen en analyseren en onderzoeken hoe de kenmerkende kleuren ervan zouden kunnen veranderen als het klimaatveranderingsprobleem niet opgelost wordt.

Groepsopdracht

Onderzoek een klimaatveranderingsprobleem dat gevolgen heeft voor een bioom regio van je keuze.

1. Verzamel maandelijkse temperatuur- en neerslaggegevens van de afgelopen 10 jaar om te zien of er trends of interessante patronen zijn. Noteer de gegevens in je Excel-werkgroep in een geschikt formaat.
2. Maak temperatuur- en neerslaggrafieken voor de bioom regio om je te helpen trends te identificeren.
3. Denk na over de bezorgdheid over klimaatverandering voor uw bioomregio en hoe die wordt weerspiegeld in de gegevens die je hebt verzameld. Bespreek hoe de kleuren in dit bioomgebied kunnen zijn veranderd als gevolg van de bezorgdheid over de klimaatverandering en doe voorspellingen over hoe de kleuren in dat gebied de komende 10 jaar kunnen veranderen als de bezorgdheid over de klimaatverandering ongecontroleerd blijft.

Leermiddelen

Werkboeken: Aarde kleur werkboek.xlsx

Sectie 3: Computationeel denken via programmeren

Totale tijd: 10 uur

In deze sectie maak je kennis met de basisprincipes van programmeren en oefen je met algoritmisch denken in de context van storytelling en games. Je leert ook flowcharts gebruiken om algoritmes weer te geven.

Bijdrage aan de leerresultaten

Leerresultaten
• ideeën om een verhaal te genereren en op te schrijven op een gestandaardiseerde manier • de belangrijkste verschillen aan te wijzen tussen de ontwikkeling van elektronische interactieve verhalen en traditionele boeken op papier • het algoritme voor een bestaand programma identificeren en uitleggen • wijzigingen aanbrengen in een bestaand programma • een bestaand programma debuggen en corrigeren

- plan en ontwerp een nieuw programma om een interactief verhaal te produceren
- een interactief verhaal maken en ontwikkelen met behulp van programmeerbare elementen
- loops, variabelen, broadcast-berichten, IF-statements en sequentiële instructies binnen een programma gebruiken
- het belang van correcte instructies begrijpen
- begrijpen wat een algoritme is
- een algoritme in een stroomschema weergeven
- de effectiviteit van een algoritme evalueren
- een vooraf geschreven algoritme of stroomschema in Scratch implementeren
- uitleggen wat wordt bedoeld met de term variabele
- maak en gebruik variabelen in je programma
- uitleggen wat wordt bedoeld met de termen "selectie" en "loop"
- selectie en loop-statements gebruiken binnen een algoritme of programma

Activiteit 3.1: Verhalen vertellen met Scratch

Digitale storytelling maakt gebruik van digitale media (beelden, stem, muziek, beweging) om een verhaal te vertellen. De afgelopen jaren is digital storytelling een steeds populairdere en effectievere leeractiviteit geworden om een reeks leerdoelen te bereiken, in het bijzonder de leerdoelen met betrekking tot computationeel denken.

In deze subsectie ga je oefenen met digitaal verhalen vertellen in Scratch. Scratch is een gratis educatieve, visuele en op blokken gebaseerde programmeeromgeving. In Scratch kunnen leerlingen hun ideeën ontwikkelen in de vorm van projecten met programmeerbare media zoals video's, afbeeldingen, games en animaties.

Opdracht in tweetallen

Bekijk de video de Intro to Scratch

Bekijk dit voorbeeldproject van achtste-klassers die het periodiek systeem bestuderen.

*

Met Scratch, kan je

- een probleem formuleren om te bepalen hoe je de elementen in Scratch kunt gebruiken om je verhaal op te bouwen -- verhaal, omgeving, volgorde en perspectief creëren.
- Logisch organiseren en analyseren van gegevens door blokken code te creëren om personages en hun omgeving te creëren.
- Verhaalinhoud weergeven door middel van de beweging van sprites - de personages in Scratch.
- Gebruik algoritmisch denken om code te ontwikkelen die sprites laat bewegen en communiceren.

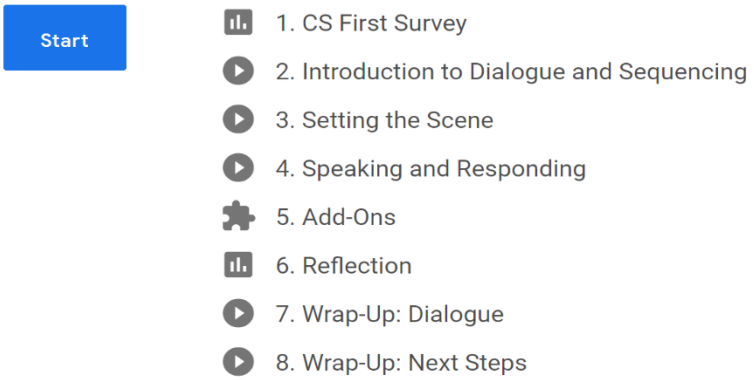
Voorbereiding: Zet alles klaar

We gaan ervan uit dat je docent je CS First gebruikersnamen en wachtwoorden heeft gegeven. Voordat je aan de slag kunt, moet je het volgende doen:

- Open een nieuw venster in uw browser en ga naar g.co/CSFirst
- Klik op "Sign in" in de rechterbovenhoek
- Klik op "I am a student"
- Klik op "Sign in with CS First"
- Klik op "Enter class code"
- Voer klasse code 3h24s3 in
- Voer je gebruikersnaam en wachtwoord in

Dialoog

Leer over CS First en maak dan een verhaal waarin twee personages praten zonder vragen te gebruiken.



Opdracht in tweetallen

- Druk op de Start knop.
- Sla de enquête over en kies 2: Introduction to Dialogue and Sequencing
- Voeg het volgende tabblad toe aan je browservenster: https://scratch.mit.edu/
- Als je nog geen Scratch-account hebt, registreer jezelf dan.
    - Tip: U kunt snel tussen tabbladen wisselen met ctrl tab
- Start de video op 2:30, volg de instructies op het einde.
- Druk op volgende
- De scène opzetten (3)
    - Op dit punt zou je een nieuw scratch project gemaakt moeten hebben, bijvoorbeeld met de naam myFirstStory
- Start de video en volg de instructies aan het eind: u hoeft de video tussendoor niet te onderbreken.
- Druk op volgende
    - Speaking and Responding (4)
- Start de video en volg de instructies aan het eind.
- Druk op volgende
    - Add-Ons (5)

- Kies ‘Adding Motion’ onderaan de pagina.
    - Hint: je hoeft de coördinaten niet zelf in te voeren als je het personage eerst op de gewenste positie zet voordat je het corresponderende move block selecteert.
- Kies ’Add a Third Character’.

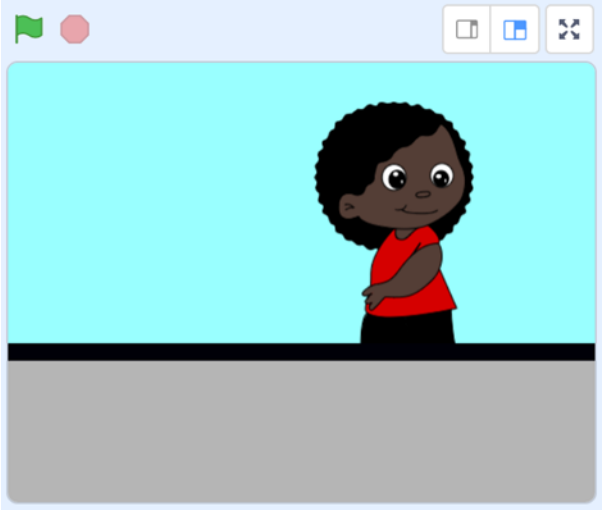
Kijken maar!

Start

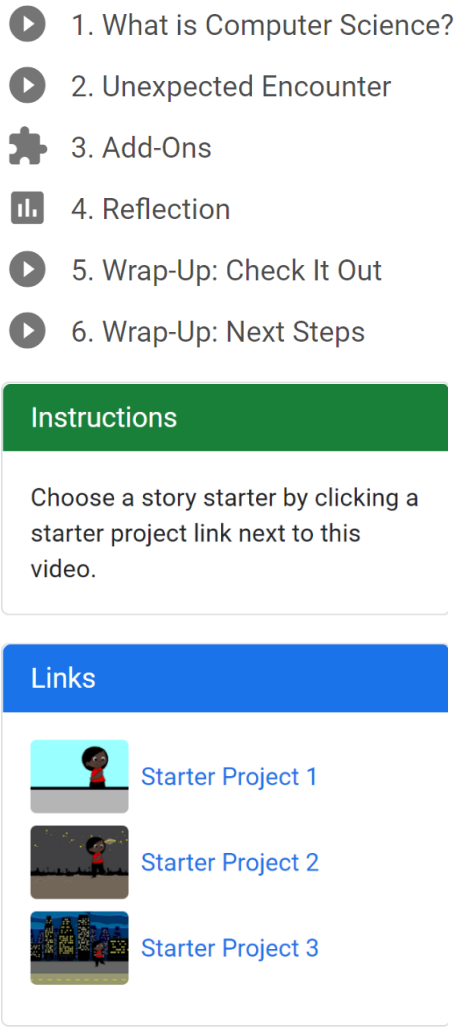
Vertel een verhaal waarin een personage door een scène loopt en beschrijft wat hij ziet.

Opdracht in tweetallen

- Druk op de Start knop.
    - Je hoeft de eerste video niet te bekijken, maar klik in plaats daarvan op de link met Starter Project 1. Het zal Scratch openen met een starter project dat al wat code bevat.



- Druk op volgende
    - Unexpected Encounter (2)
- Volg de instructies in de video
- Druk op volgende
    - Add-Ons (3)
- Kies ‘Add Sound’ en volg de instructies op de video.



Omgeving

Start

Creëer een dynamische stormachtige dag, compleet met regen en bliksem.

1. Introduction to Setting and Randomness
2. Make it Rain
3. Lightning Flash
4. Random Lightning
5. Making Your “Stormy Day” Setting into a Story
6. Add-Ons
7. Reflection
8. Wrap-Up: Setting
9. Wrap-Up: Next Steps

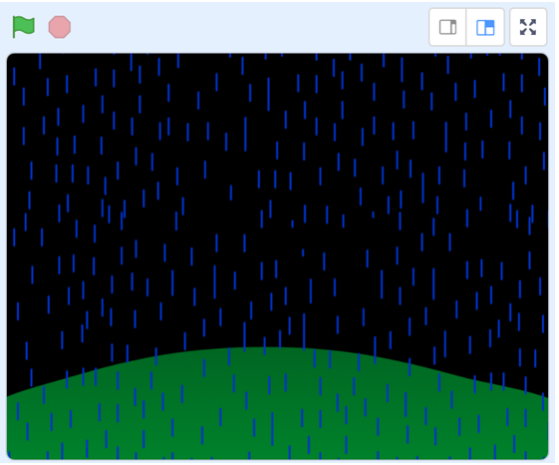
Opdracht in tweetallen

- Druk op de Start knop.
- Open het Rainy Day Starter Project

Links

Rainy Day Starter Project

- Ga naar
  - Introduction to Setting and Randomness (1)
- Bekijk de video
- Druk op volgende
  - Make it Rain (2)
- De video legt uit hoe je regen toevoegt aan je scenario. Dit wordt gedaan door gebruik te maken van een speciale regen sprite, die alle regendruppels bevat, en die geprogrammeerd is om van boven naar beneden te bewegen.
- Volg de instructies in de video
- Druk op volgende
  - Lightning Flash (3)
- Volg de instructies in de video
- Druk op volgende
  - Random Lightning (4)
- Volg de instructies in de video



- Ga naar
    - Add-Ons (6)
- Bekijk de korte video, en kies één of twee van de mogelijke add-ons onderaan de pagina.

 **Activiteit 3.2 Een doolhofspel maken**

 **Inleiding: Algoritmen**

Een algoritme is een lijst van instructies: een stappenplan. Als een algoritme nauwkeurig genoeg is geformuleerd, moet iemand anders in staat zijn de stappen precies zo te volgen als jij het bedoeld hebt. In het bijzonder algoritmen die ontworpen zijn om door een computer te worden uitgevoerd, worden geformuleerd in een zeer precieze taal, programmacode, die de computer stap voor stap precies vertelt wat hij moet doen.

 **Opdracht in tweetallen**

Bekijk de volgende video: What is an algorithm and why should you care?

 **Subsectie 1: Onderzoek van algoritmen met behulp van stroomdiagrammen**

Bekijk het volgende algoritme.

1. Teken een verticale lijn
2. Teken er een horizontale lijn die de vorige lijn kruist
3. Teken een diagonale lijn van de top van de verticale lijn naar de top van de horizontale lijn
4. Herhaal instructie 3 voor alle overige hoeken
5. Trek een golvende lijn vanaf het onderste punt naar beneden

 **Groepsopdracht**

Maak een tekening van deze instructies.

Vergelijk je tekening met die van anderen. Bespreek waarom het wel of niet gemakkelijk was en leg uit waarom.

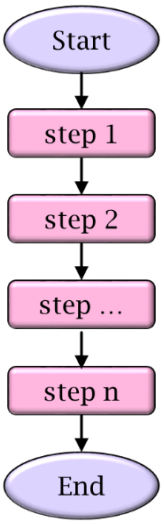
*

Het beoogde resultaat van het algoritme was een tekening van een vlieger (zie bijlage A voor een mogelijke voorstelling). Het voorbeeld toont aan dat wanneer de instructies onduidelijk of onvolledig zijn, het resultaat dubbelzinnig kan zijn. Soms wordt van de uitvoerder verwacht dat hij eventuele onduidelijkheden zelf oplost. Bedenk wat er zou gebeuren als we instructies altijd precies zouden opvolgen zoals ze worden gegeven.

Opdracht in tweetallen

Bekijk de video [the exact instructions challenge].

Als we een algoritme willen beschrijven, moeten we het eens worden over een taal die we daarvoor gebruiken. Zo'n taal moet voldoende precies zijn: het is zeker wat er moet gebeuren als we een algoritme volgen dat in die taal is geschreven. Een algoritme kan visueel worden voorgesteld in een stroomdiagram. Dit helpt het overzicht te bewaren, wat het op zijn beurt gemakkelijker maakt instructies te schrijven, te lezen en te analyseren.



De eenvoudigste algoritmen bestaan uit een reeks instructies die na elkaar worden uitgevoerd. In een stroomdiagram geven we zo'n reeks instructies weer zoals hiernaast is afgebeeld. Je begint bij 'Start' bovenaan. Het deel tussen 'Start' en 'End' (het lichaam van het stroomdiagram) beschrijft wat het eigenlijk doet.

Individuele opdracht

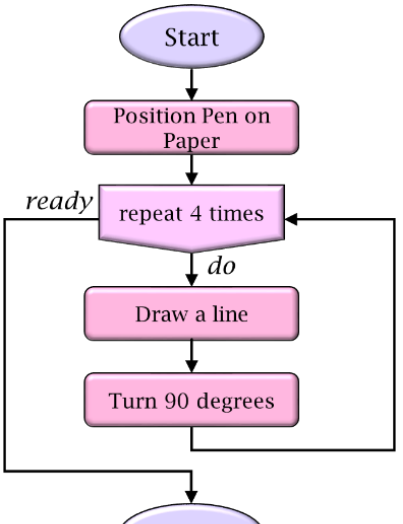
Een vierkant tekenen

Laten we aannemen dat je de volgende basiscommando's kunt gebruiken:

- **Trek een lijn.**
- **90 graden draaien**
- **Plaats pen op papier**

Kies geschikte instructies en plaats deze in een stroomdiagram om een vierkant te tekenen.

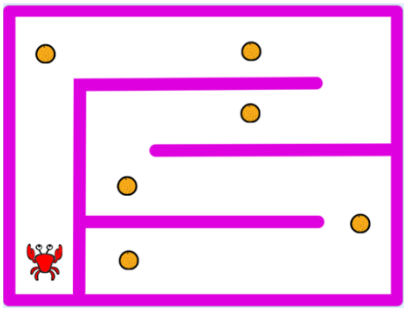
Uw oplossing bestaat waarschijnlijk uit een lange reeks instructies die herhalingen van code bevat. We kunnen zo'n reeks korter en dus duidelijker opschrijven door een repeat/while construct te gebruiken: zolang we nog geen 4 herhalingen hebben uitgevoerd, wordt de do-tak gevolgd. Na de vierde iteratie volgen we de klaar-pijl en eindigt het algoritme.



Opdracht in tweetallen

- Een leerling kiest een tekening en schrijft instructies om zijn partner uit te leggen hoe hij de tekening moet maken
- Bedenk samen een reeks basiscommando's die in de beschrijving kunnen worden gebruikt
- De andere leerling volgt deze instructies precies op
- Als de leerling klaar is, wordt de oplossing besproken en fouten gecorrigeerd

Subsectie 2: Maak een doolhofspel met sprites



In deze subsectie ontwerp je een spel in Scratch waarin de speler een krab bestuurt met behulp van de pijltjestoetsen. Het doel van het spel is om de weg naar de andere kant van het doolhof te vinden zonder de muren te raken. In het doolhof liggen sinaasappels verspreid. De speler verzamelt punten door de krab de sinaasappels te laten oprapen.

Leermiddelen

Scratch bestand: krab-in-maze-start.sb3

Opdracht in tweetallen

Probeer uit te werken hoe je spel zou kunnen lopen. Maak een ruwe schets van je spel. Denk na over hoe het spel eruit ziet en hoe het hoofdpersonage door de speler wordt bestuurd. Probeer voorwaarden te formuleren die aangeven of het spel is afgelopen. Welke 'schatten' voeg je toe aan je spel? Waar liggen ze en hoe pakt de speler ze op?

*

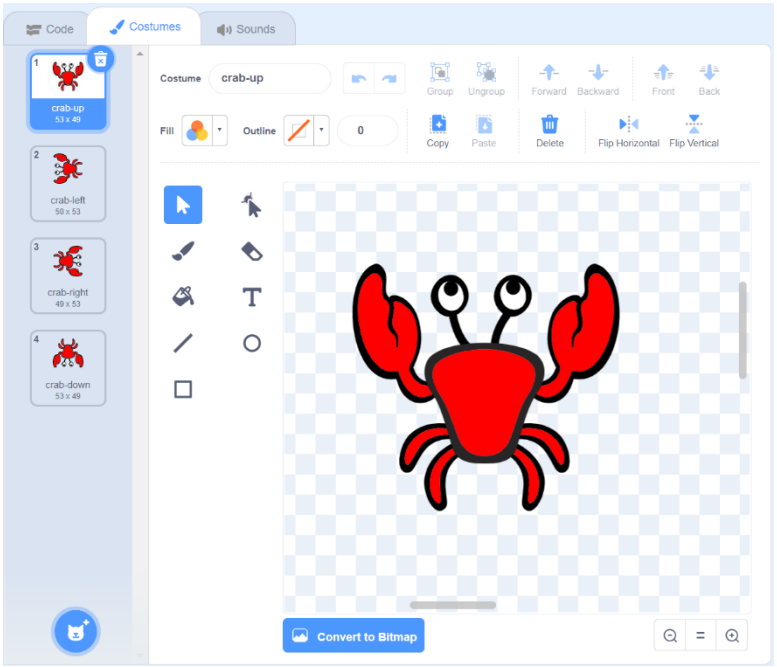
Om dit spel te programmeren, zul je leren hoe je

- sprite kostuums, positionering en beweging bepaald
- toetsenbordinvoer verwerkt
- de sprite interactie met de omgeving behandeld
- de programma sprites met elkaar laat communiceren
- variabelen gebruikt om een score bij te houden

We beginnen met het aanpassen van het uiterlijk van het hoofdpersonage van het spel: de krab (geïmplementeerd in Scratch als een sprite).

Opdracht in tweetallen

- Ga naar Scratch en open het bestand crab-in-maze-start.sb3 (door 'load from your computer' te kiezen in het File menu).
- Selecteer de krab sprite en kies het Costumes tablad.
- Verwijder het tweede plaatje, kies het eerste plaatje, en maak de klauwen van de krab wat kleiner en schaal het hele plaatje zo dat de krab comfortabel in het doolhof past (zodat de muren niet geraakt worden)
- Dupliceer de afbeelding 3 keer (door met de rechtermuisknop op de afbeelding te klikken en 'dupliceren' te selecteren) en draai deze kopieën voor een vogelperspectief van je krab: links, rechts, omhoog en omlaag.

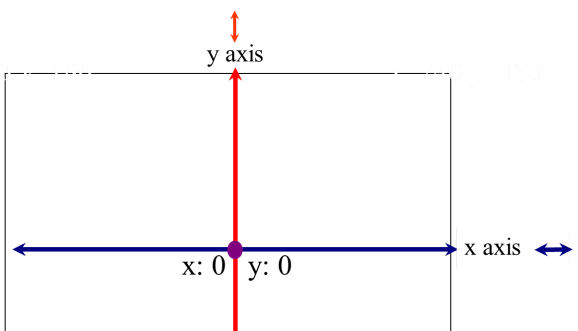


- Verander de namen hiervan op dezelfde manier.

*

Om de beweging van de krab te programmeren is het nodig om te weten hoe sprites worden gepositioneerd in Scratch. Scratch stelt elk punt in het speelveld voor als een paar van twee getallen: het eerste geeft de horizontale afstand vanaf het midden van het veld aan, het tweede de verticale afstand.

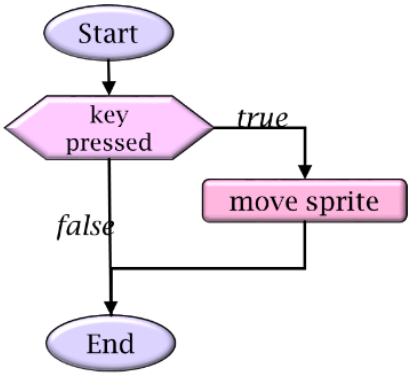
In meer technische termen gebruikt Scratch een zogenaamd coördinatensysteem met de oorsprong in het midden van het speelveld. Elk punt wordt voorgesteld door een paar (x, y) waarbij x de horizontale afstand tot de oorsprong is en y de verticale afstand.



Het totale speelveld is 480 pixels breed en 360 pixels hoog. De linkerbovenhoek en de rechterbovenhoek hebben coördinaten (-240,180) en (240,180), respectievelijk.

Terug naar ons spel. De speler beweegt de krab met behulp van de pijltjestoetsen. De vraag is nu hoe we in ons programma kunnen detecteren of de speler een toets heeft ingedrukt en hoe we dan de positie van de krab kunnen aanpassen aan de ingedrukte toets,

Het algoritme voor de behandeling van ingedrukte pijltjestoetsen kan als volgt in een stroomdiagram worden gespecificeerd:

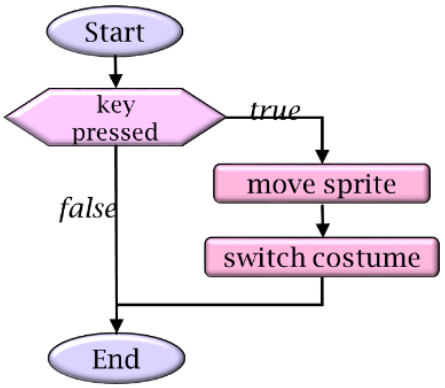


We hebben hier een selectie gebruikt, ook bekend als een if-then construct. In zo'n selectie wordt aan de hand van een voorwaardelijke uitdrukking (aangegeven in het zeshoekige blokje) bepaald of de then-tak (de tak met het label true) moet worden uitgevoerd. Dit laatste gebeurt alleen als de voorwaarde waar is.

Opdracht in tweetallen

- Probeer uit te vinden welk *event blok* je kunt gebruiken om te reageren op ingedrukte pijltjestoetsen
- Pas de positie van de krab correct aan. Welk *motion block* heb je nodig?

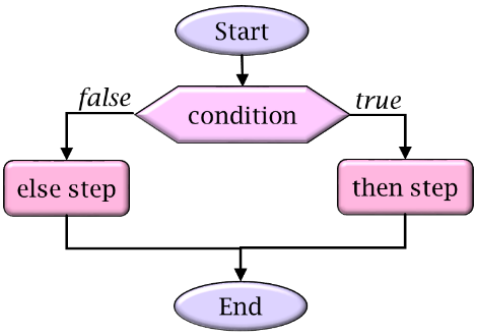
  Om het spel realistischer te maken, zullen we het beeld van de krab laten overeenkomen met de richting van zijn beweging. Dit doen wij door steeds het juiste kostuum te kiezen. Uitgedrukt in een stroomschema:



- Bepaal welk *looks block* je nodig hebt voor dit.
- Voer je code uit en test het.

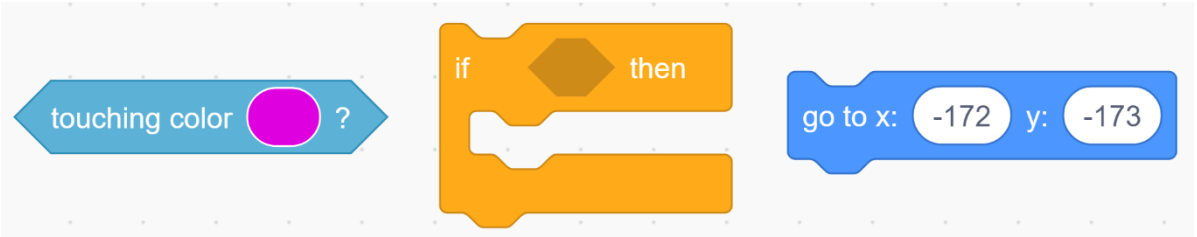
*

De algemene variant van een selectie statement biedt de mogelijkheid om ook een actie uit te voeren als de voorwaarde onwaar is. In het vorige voorbeeld gebeurde er niets bij een false conditie. In een stroomdiagram wordt een if-then-else statement als volgt aangegeven.



Als de voorwaarde waar is, wordt de "then-tak" (met het label true) uitgevoerd. Zo niet, dan wordt de "else tak" (gelabeld false) uitgevoerd.

Scratch heeft de volgende blokken om de muis te bewegen en te controleren of de krab de wanden van het doolhof raakt.



Opdracht in tweetallen

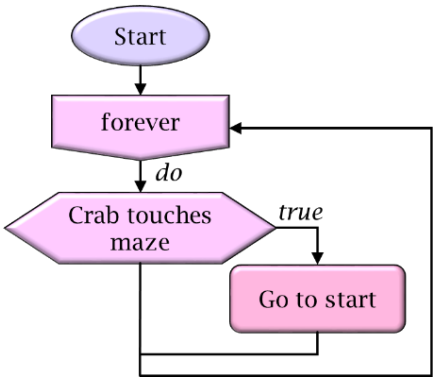
Bepaal welke blokken je nodig hebt en wat de volgorde van de blokken moet zijn.

- Als de krab de zijkanten van het doolhof raakt, hoe beweeg je de krab dan terug naar het begin.
- Voeg de juiste blokken toe aan je code
- Start en test je code.
    - Werkt het? Zo niet, probeer uit te zoeken wat er fout gaat en los het op.

*

We hebben al gezien hoe we in een stroomschema kunnen uitdrukken dat een bepaald deel van een algoritme een aantal keren herhaald moet worden. Het is mogelijk dat we willen aangeven dat een algoritme altijd herhaald moet worden en dus nooit stopt. In ons

krabbenspel kunnen we dit gebruiken om steeds opnieuw te controleren of de krab de muren raakt en zo nodig terug naar het begin moet worden geplaatst. In een stroomdiagram:



Merk op dat dit stroomschema geen eindpunt heeft.

Opdracht in tweetallen

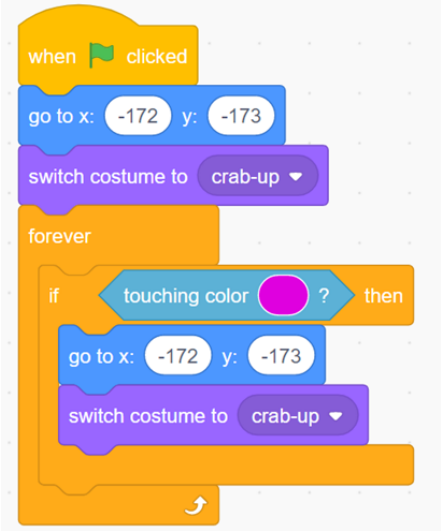
- Gebruik de eeuwige lus (van de *control blocks*) om de beweging van de krab af te programmeren.
- Zoek uit wat er moet gebeuren als de krab een van de muren raakt.
- Start en test je code.

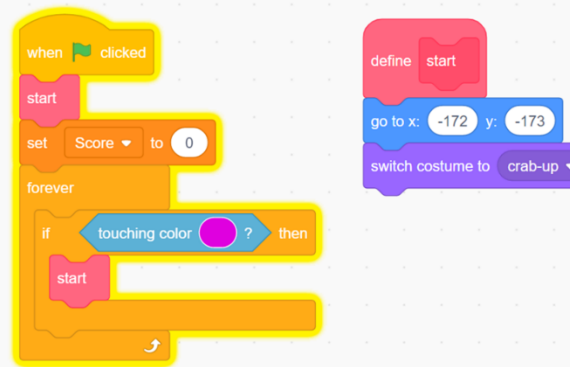
*

Een natuurlijke manier om grote problemen op te lossen is ze op te splitsen in een reeks sub-problemen, die min of meer onafhankelijk van elkaar kunnen worden opgelost en vervolgens gecombineerd om tot een volledige oplossing te komen. Tijdens de ontwerpfase van een algoritme moet de probleemoplosser, wanneer een probleem in sub-taken is onderverdeeld, alleen rekening houden met wat een sub-algoritme globaal moet doen en zich niet bezighouden met de details van die sub-taak. Deze scheiding van zorgen staat bekend als abstractie. Door het proces van abstractie kan een programmeur alle details over sub-algoritmen verbergen om de complexiteit te verminderen en zo het programma begrijpelijker te maken.

Scratch gebruikt abstracties om programmeren gemakkelijker te maken door een veelheid van basisbouwstenen aan te bieden waarvan de complexe onderliggende implementaties verborgen blijven. Scratch-programmeurs kunnen abstracties introduceren door zelf nieuwe blokken aan het programma toe te voegen.

Beschouw bijvoorbeeld de volgende programmafragmenten, die beide een oplossing kunnen zijn voor het vorige probleem. Links zie je de oplossing waar alles rechttoe rechtaan is uitgewerkt. Rechts wordt abstractie toegepast, door het verplaatsen van de krab en het veranderen van het kostuum als een nieuwe bouwsteen in te voeren (met de naam Start) en deze vervolgens in het hoofdprogramma te gebruiken.





Opdracht in tweetallen

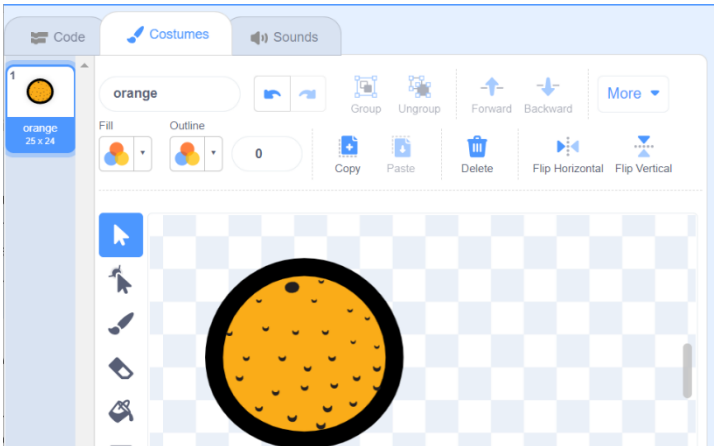
Ga door je Scratch scenario en pas abstractie toe door (gedetailleerde) codefragmenten te vervangen door blokken die je zelf hebt gedefinieerd

*

Een speler van het spel moet nu in staat zijn de krab te besturen met de pijltjestoetsen en de krab zal correct reageren wanneer een muur van het doolhof wordt geraakt. We gaan het spel nu aantrekkelijker maken door uitdagingen toe te voegen. In het begin beperken we ons tot sinaasappels die we over het doolhof verspreiden en die door de krab worden opgegeten. Dit gebeurt zodra de krab ze aanraakt. Het eten van een sinaasappel levert een bonuspunt op. Het doel van het spel is nu om zoveel mogelijk bonuspunten te verdienen.

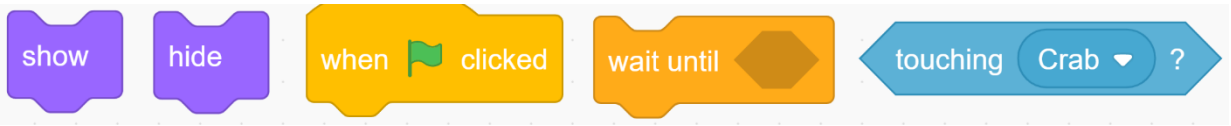
Opdracht in tweetallen

- Selecteer de *orange* sprite en schaal de sinaasappel zodat de grootte in verhouding is met de krab.



We stuiten nu op een interessant probleem: wie maken we verantwoordelijk voor het opsporen en afhandelen van de interactie tussen krab en sinaasappel? In de echte wereld zal dat altijd de krab zijn, maar hier hebben we een keuze. We kunnen de sinaasappel meer verantwoordelijkheden geven dan in werkelijkheid. En we zullen nu gebruik maken van deze interessante optie: we zullen de sinaasappel laten detecteren of hij gevonden is door de krab en zichzelf laten opeten. Dit laatste bereiken we door de sinaasappel te verstoppen.

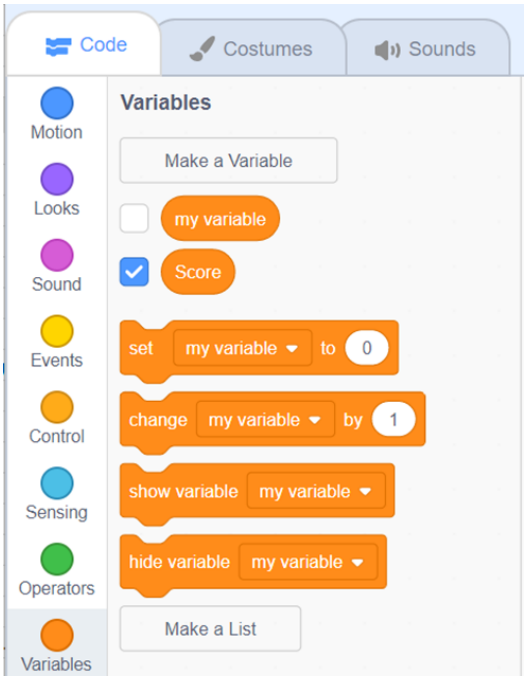
- Hieronder zie je de blokken die je (waarschijnlijk) nodig zult hebben om de sinaasappels te programmeren. Gebruik deze blokken om het gewenste gedrag te krijgen.



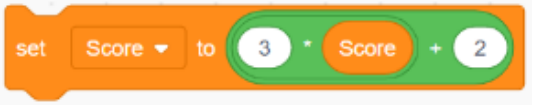
- Start en test je code!
    - Is er nog iets anders dat nu moet gebeuren?

*

Wij willen ook een manier aan het programma toevoegen om de score bij te houden. We zullen hiervoor een zogenaamde variabele gebruiken. In programmeertalen is een variabele een kistje die één stukje informatie tegelijk kan bevatten, zoals een woord of een getal. Doordat we dit stukje informatie kunnen vasthouden, kunnen we er op verschillende plaatsen in een programma naar verwijzen en het manipuleren. Deze mogelijkheid maakt variabelen ongelooflijk nuttig.



Hoe maken we een variabele in Scratch? Voordat we een variabele kunnen gebruiken, moeten we deze eerst maken met de knop "Make a Variable" in het Block Palette. De waarde die een variabele in Scratch kan bevatten is ofwel een tekst ofwel een getal. Verder zijn er blokken om een (initiële) waarde aan een variabele te geven en om de waarde tijdens de uitvoering van het programma te wijzigen. Hiernaast zie je welke blokken je kunt gebruiken voor het manipuleren van variabelen. Overigens hoeft het waardeveld in bijvoorbeeld het set-blok niet per se een getal te zijn: ook complexere expressies samengesteld met operatoren uit het operator-palet zijn hier toegestaan. Stel bijvoorbeeld dat Score de waarde 4 heeft. Na de



uitvoering van

De score zal de waarde 3 * 4 + 2 = 14 hebben.

Opdracht in tweetallen

- Voeg een variabele met de naam Score toe aan je programma.
- Bedenk welke waarde deze variabele in het begin heeft en bedenk hoe je deze waarde kunt instellen.
- Overweeg ook waar en hoe de waarde van Score moet worden aangepast en welk blok daarvoor kan worden gebruikt
- Start en test je code!
- Dupliceer de sinaasappel (in het sprites venster) een aantal keer (bijvoorbeeld 6) en plaats deze kopieën ergens in het doolhof.

*

Sectie 4: Computationeel denken onderwijzen en aanleren

Totale tijd: 8 uur

Onderwijzen en leren over computationeel denken kan op twee manieren: zonder computers ('unplugged') en met computers ('plugged'). In deze sectie ervaar je beide manieren en leer je over ontwerpprincipes voor instructiestrategieën voor computationeel denken.

Bijdrage aan de leerresultaten

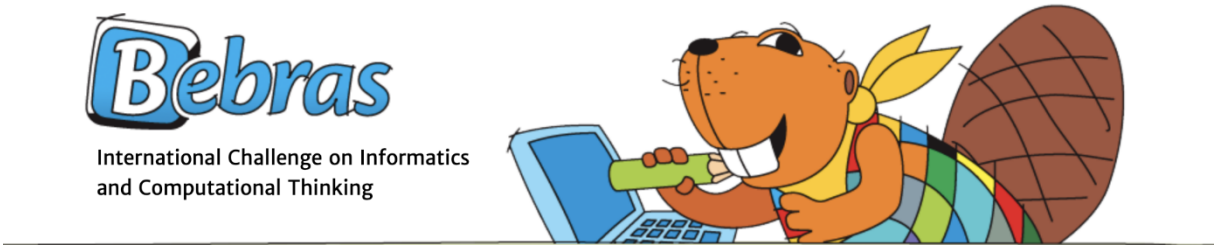
Leerresultaten
• kenmerken van "unplugged" en "plugged" onderwijs- en leeractiviteiten voor computationeel denken te beschrijven • essentiële elementen van instructiestrategieën voor computationeel denken beschrijven • dergelijke elementen te herkennen in concrete leermiddelen en activiteiten

Activiteit 4.1 Bebras opdrachten

Bebras

Bebras is een online wedstrijd en uitdaging, georganiseerd door een internationale gemeenschap. De elementen van de wedstrijden zijn zogenaamde Bebras-opdrachten, die elk één of meer computationele concepten behandelen. In deze activiteit ga je Bebras-opdrachten

verkennen, zowel als onderdelen van de wedstrijd als 'unplugged' leeractiviteiten voor computationeel denken.



Opdracht in tweetallen

- Ga naar de Bebras-website en kies maximaal vijf voorbeeldopdrachten van Bebras (probeer te variëren in de opdrachten).
- Voer de opdrachten individueel uit en vergelijk en bespreek ze daarna met je partner.
- Kun je de opdrachten classificeren in termen van de computationeel denken stappen en activiteiten die je in deze module hebt geleerd?

Bebras opdrachten als leeractiviteiten

Bebras taken zijn gecategoriseerd in termen van de computationele concepten die worden behandeld, elke taak heeft een uitleg over het verband tussen de taak en computerwetenschap.

Valentina Dagienė (de 'moeder van Bebras') en Sue Sentance onderzochten het gebruik van Bebras-opdrachten als CT-leeractiviteiten:

Dagienė, V., & Sentance, S. (2016). It's Computational Thinking! Bebras tasks in the curriculum. In *International conference on informatics in schools: Situation, evolution, and perspectives* (pp. 28–39).

Opdracht in tweetallen

- Lees het artikel.

- Bekijk de concepten en activiteiten die je in de secties 2 en 3 bent tegengekomen. Zoek voor elk van deze modules een voorbeeld van een rekenconcept en een bijpassende Bebras-opdracht.

Activiteit 4.2 Pathfinding: paden in een doolhof

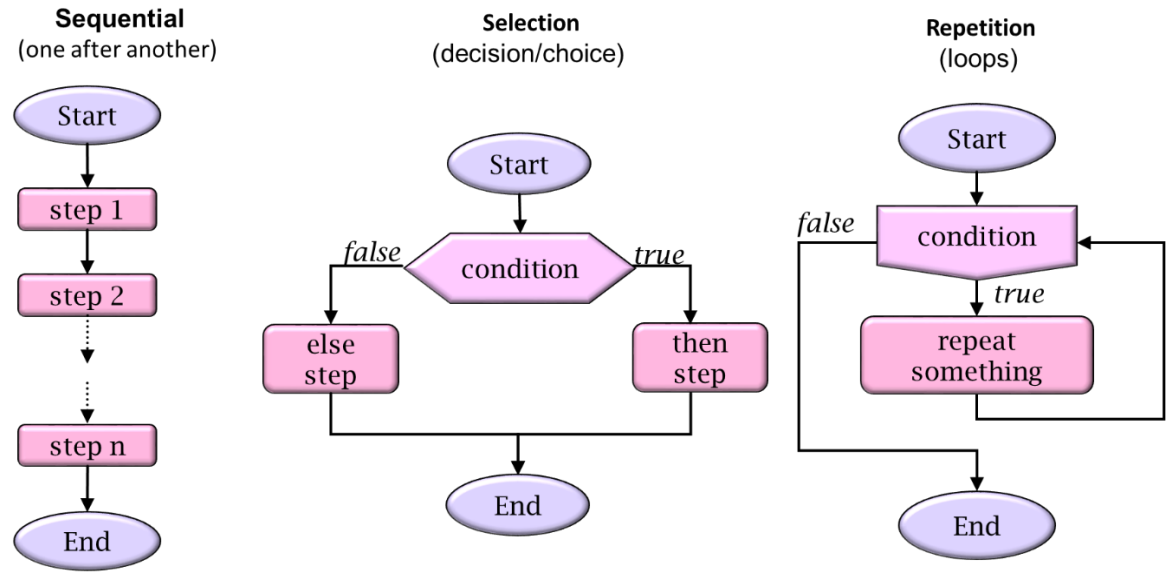
Doolhof oplossen is het proces van het vinden van een pad door het doolhof van begin tot eind. Sommige methoden voor het oplossen van doolhoven zijn ontworpen om binnen het doolhof te worden gebruikt door een reiziger zonder voorkennis van het doolhof, terwijl

andere ontworpen zijn om te worden gebruikt door een persoon (of computerprogramma) die het hele doolhof in één keer kan zien. Het oplossen van doolhoven is een variant van een meer praktische klasse van problemen die bekend is onder de naam pathfinding. Gegeven een aantal locaties die onderling verbonden kunnen zijn (bijvoorbeeld een verzameling steden die door wegen met elkaar verbonden zijn), zoekt een pathfinding-algoritme een route die deze locaties met elkaar verbindt met de bedoeling de beste route te bepalen (b.v. de kortste of goedkoopste route).

Het is onze bedoeling om deze pathfinding-problemen te gebruiken om een eerste inzicht te krijgen in probleemanalyse en algoritme-ontwerp zonder gedetailleerde kennis van computerprogrammering en programmeertalen. Om echter mogelijke oplossingen met voldoende precisie te kunnen formuleren, zullen we eerst weer stroomdiagrammen gebruiken.

Stroomdiagram elementen

Hieronder ziet u de drie basisbouwstenen waarmee stroomdiagrammen kunnen worden samengesteld.



Elk stroomschema is opgebouwd uit de volgende drie basiselementen

1. **Sequentie**: een geordende reeks instructies die na elkaar worden uitgevoerd.
2. **Selectie**: op basis van een voorwaardelijke uitdrukking wordt bepaald of de then-tak (de tak met het label waar) of de else-tak (de tak met het label onwaar) moet worden uitgevoerd.
3. **Herhaling**: de *body* van het element (de tak met het label waar) wordt herhaald zolang de voorwaarde op waar wordt geëvalueerd. Daarna gaat het programma verder door de tak met het label false te volgen.

Reizende verkoper probleem

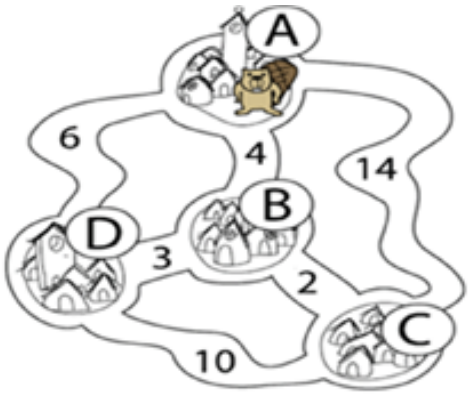
Het zogenaamde reizende-verkopersprobleem is een klassiek voorbeeld van een opgave waarvoor een optimale oplossing wordt gezocht.

"Gegeven een lijst van steden en de afstanden tussen elk paar steden, wat is de kortst mogelijke route die elke stad langsgaat en terugkeert naar de stad van herkomst?"

We gaan nu kijken naar een oplossing voor dit probleem die weliswaar niet altijd de beste route vindt, maar wel intuïtief en gemakkelijk te begrijpen is, namelijk het *nearest neighbor algoritme*. Het nearest neighbour algoritme (NNA) laat de verkoper de dichtstbijzijnde niet-bezochte stad kiezen als zijn volgende zet.

Opdracht in tweetallen

Bevers zwemmen vanuit stad A, ze gaan op bezoek bij stad B, C en D en komen dan terug in A. Ze willen de kortste route vinden.



1. Welke route vinden ze als ze NNA zouden gebruiken?
2. Geeft NNA altijd het juiste antwoord (d.w.z. voor elke lijst van steden en verbindingen daartussen)? Zo niet, geef een concreet tegenvoorbeeld

*

Hoewel de NA intuïtief duidelijk is, kan het moeilijk zijn dit algoritme precies te formuleren. We proberen dit algoritme uit te drukken in een flowchart waarbij we eerst moeten bepalen welke primitieve instructies we hiervoor kunnen gebruiken. Deze primitieven moeten aan de ene kant krachtig genoeg zijn, maar aan de andere kant ook voldoende abstract, zodat we niet in allerlei details verstrikt raken.

Opdracht in tweetallen

Omschrijf de NNA in een stroomschema.

Hint: Denk eerst na over de basisinstructies die u mag gebruiken.

Een pad vinden in een doolhof

Het vinden van een pad in een doolhof is een ander voorbeeld van een opgave die gemakkelijk te begrijpen is, maar ook uitdagend genoeg om op te lossen. Er zijn verschillende varianten van dit probleem: vind een pad uit een doolhof, vind een pad door een doolhof of vind een pad naar een specifieke plaats in een doolhof. Ten eerste kunnen we opmerken dat deze varianten veralgemeend kunnen worden tot: vind een pad van positie A naar positie B.

Het is ook belangrijk wat we weten over het doolhof terwijl we zoeken naar de doelpositie B. In deze opdracht gaan we ervan uit dat we niets weten over de grootte en de structuur van het doolhof. We kunnen door de gangen van het doolhof lopen, waarbij we telkens één stap vooruit kijken. We nemen dus aan dat we op elk punt in het doolhof kunnen controleren of we rechtdoor kunnen lopen (en dus niet voor een muur staan) en eventueel onze bewegingsrichting kunnen aanpassen.

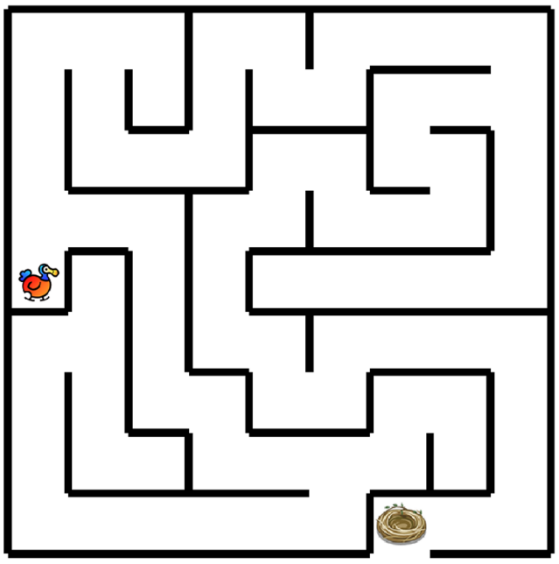
De bekendste regel voor het doorkruisen van een doolhof is de muurvolger (ook bekend als de linkerhandregel of de rechterhandregel). Als alle muren van een doolhof met elkaar verbonden zijn, dan zal de reiziger, door één hand in contact te houden met één muur van het doolhof, gegarandeerd niet verdwalen en de uitgang bereiken.

Om het probleem concreter te maken, gebruiken wij het volgende scenario waarin de doolhofreiziger wordt voorgesteld door een dodo die zich ergens in het doolhof bevindt en zijn elders verborgen nest probeert te vinden.

Voordat wij een zoekalgoritme kunnen specificeren, moeten wij opnieuw aangeven met welke primitieve instructies wij de dodo kunnen besturen.

Wij maken onderscheid tussen *commando's* (om de dodo een stap vooruit te laten doen of de bewegingsrichting te veranderen) en *zoekopdrachten* (waarmee wij de dodo lokale informatie over zijn omgeving kunnen laten verstrekken).

- Commando's:
  - *linksaf*: 90 graden draaien tegen de klok in
  - *rechtsaf*: 90 graden draaien met de klok mee
  - *beweeg*: ga één veld vooruit



    - *ei leggen*: een ei leggen (op de huidige plaats)
- Vragen/testen:
    - *kan bewegen?* kan je een stap voorwaarts zetten?
    - *nest gevonden?* heb je het nest gevonden?

Opdracht in tweetallen

Gebruik een stroomdiagram om een algoritme te laten zien waarmee de dodo het volgende eipatroon produceert. Hint: gebruik herhalingen.

Opdracht in tweetallen

Specificeer de volg-de-linkermuur strategie in een stroomschema.

Pathfinding unplugged

Voorbereiding

Zet bijvoorbeeld met schilderstape een doolhof uit op de vloer. Zorg ervoor dat de gangen breed genoeg zijn zodat een persoon er gemakkelijk doorheen kan lopen. Het doolhof mag iets eenvoudiger zijn dan het doolhof op de foto. Wissel de stroomschema's uit de vorige opdracht om. Wijs een plaats in het doolhof aan als eindbestemming en één van jullie gaat ergens in het doolhof ver genoeg van deze eindbestemming staan.

Opdracht in tweetallen

Een van jullie zal ergens in het doolhof staan, ver genoeg weg van de eindbestemming.

- De andere persoon leest de instructies van het stroomschema voor en degene in het doolhof volgt deze instructies nauwgezet op.
- Is de eindbestemming bereikt? Zo ja, probeer ook uit of dit geldt voor andere startlocaties in het doolhof. Zo nee, wat is er mis met het algoritme? Verbeter het algoritme en ga na of de verbetering het gewenste effect heeft.
- Geef de verbeterde versie van het stroomschema terug aan hun eigenaars.

Leermiddelen

- Scratch bestand: PathThroughMaze.sb3

Pathfinding in Scratch

Tenslotte ga je het algoritme uit de vorige opdracht implementeren in Scratch. We hebben een beginscenario gemaakt dat je hiervoor kunt gebruiken. Open het beginscenario PathThroughMaze.sb3.

Nu volgt een korte uitleg. De primitieve dodo-instructies zijn als afzonderlijke blokken aan het scenario toegevoegd. De *layEgg* instructie ontbreekt omdat deze niet nodig is voor deze opdracht. Alle andere opdrachten werken zoals eerder aangegeven. De query's zijn op een speciale manier gerealiseerd, ook omdat zelf gedefinieerde blokken in Scratch geen resultaten kunnen opleveren. De test om te controleren of de dodo het nest heeft gevonden kan eenvoudig worden gerealiseerd met een voorgedefinieerd blok uit het *sensing palet*. Wij hebben een nieuw blok geïntroduceerd voor de *canMove* test. Om een resultaat terug te geven, gebruiken we een variabele die we, net als het blok zelf, ook *canMove* hebben genoemd. Dus nadat je het *canMove* blok hebt uitgevoerd kun je de waarde van de corresponderende variabele inspecteren om te zien of de dodo een stap kan zetten of niet. De variabele *canMove* bevat ofwel de waarde 0 ofwel 1. De waarde 0 geeft aan dat er geen stap kan worden gezet, terwijl waarde 1 aangeeft dat het wel mogelijk is.

Opdracht in tweetallen

Het startscenario bevat een blok met de naam *tryToFindExit*. Dit blok doet nog weinig: het presenteert alleen de boodschap dat de uitgang gevonden is op het scherm.

- Implementeer dit blok door het stroomschema om te zetten in Scratch-code.
- Draai en test je programma ook voor andere startposities van de dodo. Pas de code aan indien nodig.

Activiteit 4.3 Instructiestrategieën

In deze activiteit zul je strategieën en pedagogische principes voor computationeel denken verkennen.

Terugkijken

Opdracht in tweetallen

Bespreek uw ervaringen in de Pathfinding-activiteit:

- Wat zijn de essentiële verschillen tussen de "unplugged" en de "plugged" variant?
- Welke uitdagingen voor uw toekomstige studenten verwacht u voor elk van deze?

Instructiebeginselen voor CT

In een overzichtsartikel analyseren Lye en Koh (2014) 27 empirische studies naar computationeel denken, in het bijzonder studies naar klassikale activiteiten waarbij programmeren een rol speelt. Lye en Koh identificeren onder andere instructiebenaderingen die computationeel denken bevorderen.

Lye en Koh groepeerden de benaderingen in vier categorieën, die vaak in combinaties voorkomen: *versterking van computationele concepten*, *reflectie*, *informatieverwerking*, en *het construeren van programma's met steigers*.

Opdracht in tweetallen

1. Lees de samenvatting over de vier categorieën van het document ’Instructional strategies for computational thinking " (te vinden in de leermiddelen).
2. Analyseer de computationeel denken activiteiten die je zelf hebt uitgevoerd in deze module. Welke van de strategieën herken je? Bespreek.

Opdracht in tweetallen

1. Lees de samenvatting over “assessment in computational thinking” (zie het document "Assessment in CT").
2. Bespreek welke beoordelingsmethoden geschikt zouden zijn voor de activiteiten die je in sectie 2 of sectie 3 hebt uitgevoerd. Kies één mogelijke aanpak voor elke activiteit. Hoe zou je deze benaderingen gebruiken om de computationele denkvaardigheden van leerlingen te evalueren?

Conclusie

Discussie opdracht

Vergelijk jullie bevindingen in de klas. Welke aspecten vond je gemakkelijk, welke waren moeilijk? Wat heb je geleerd?

Leermiddelen

- A. Instructional strategies for computational thinking (Summary of instructional strategies for CT teaching and learning, based on 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

Overzicht van leermiddelen van module O2

- Activity 1.1. - A. Brief introduction to CT
- Activity 1.1. - B. Definitions of CT
- Activity 2.1 - A. Gemeinschaft to Gesellschaft.pdf
- Activity 2.1 - B. The changing psychology of culture from 1800 through 2000
- Activity 2.1 - C. The changing psychology of culture in German‑speaking countries
- Activity 2.2 - zikavirusmodel.nlogo
- Activity 2.4 - Earth colour workbook.xlsx
- Activity 3.2 - crab-in-maze-start.sb3
- Activity 4.1 - Dagiene and Sentance 2016 (Dagienė, V., & Sentance, S. (2016). It's Computational Thinking! Bebras tasks in the curriculum. In *International conference on informatics in schools: Situation, evolution, and perspectives* (pp. 28–39)).
- Activity 4.2. PathThroughMaze.sb3
- Activity 4.3 - A. Instructional strategies for computational thinking (Summary of instructional strategies for CT teaching and learning, based on 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.)
- Activity 4.3 - B. Assessment in CT