Woord vooraf

Ter afsluiting van mijn opleiding, Bachelor in het Secundair Onderwijs, aan VIVES Brugge schreef ik deze scriptie.

Zonder hulp van anderen is het schrijven van een scriptie een moeilijke opdracht. Daarom wil ik langs deze weg de volgende personen bedanken voor hun bereidwillige medewerking, begeleiding, steun en informatie.
Mevrouw Buffel A., docent informatica.
De heer Carreyn B., leerkracht Sint-Jozef Sint-Pieter
Mevrouw Grymonpon G., docent wiskunde.
De leerlingen uit 4 handel, Sint-Jozef Sint-Pieter.

Ik wil ook mijn ouders bedanken voor de vele moedige woorden en steun gedurende de drie jaar.

Mijn vriend Dieter Wylin mag zeker in dit dankwoord niet ontbreken voor alle steun en motivatie. Ondanks zijn eigen verplichtingen wist hij op meer dan één manier bijdrage te leveren aan deze scriptie.

Tenslotte gaat een woord van dank uit naar alle personen die ik hier onbewust ben vergeten te vernoemen en die op één of andere wijze hebben bijgedragen tot de verwezenlijking van deze scriptie.

Ann-Sophie Fevery, 26 mei 2014

Inhoudsopgave

Inleiding

Naar aanleiding van de nieuwsheisa rond "tablets in het onderwijs" zijn veel uitgevers en scholen bezig met een ontdekkingstocht naar hoe deze apparaten het beste in te zetten zijn. Een visie die ik nog niet veel ben tegengekomen, is het gebruiken van een tablet om te leren programmeren. Er is hier nog zeer weinig tot geen lesmateriaal voor ontworpen.

Daarom heb ik een lessenpakket ontwikkeld om te leren programmeren op de tablet (app Codea). Het lessenpakket is bedoeld voor leerlingen uit de tweede graad van het secundair onderwijs. De cursus die ik heb ontworpen is vooral bedoeld om digitaal te gebruiken op de iPad in combinatie met de applicatie Codea.
Ik heb het lessenpakket dan ook kunnen uittesten tijdens mijn stage in Sint-Jozef Sint-Pieter in Blankenberge. Ik had het geluk mijn lessenpakket te mogen uittesten in de klas 4 handel waar ze 4uur informatica per week hebben. Na deze stage heb ik dan de nodige aanpassingen aan het lessenpakket kunnen doen.

Voor ik het lessenpakket ben beginnen ontwikkelen, heb ik nog een aantal onderzoeken uitgevoerd.
Een eerste onderzoek ging uit naar welke applicatie ik ging gebruiken voor het lessenpakket. Daarvoor heb ik een aantal applicaties geëvalueerd aan de hand van een fiche die ik zelf heb opgesteld.
Een tweede onderzoek ging over hoe ik de leerlingen meer kon aanzetten tot de attitude "eerst denken en dan doen". Hiervoor heb ik een stappenplan opgesteld dat de leerlingen moeten doorlopen bij elke oefening.
Het laatste onderzoek, in functie van het lessenpakket, ging uit naar de cognitieve multimediatheorie van Mayer. Deze theorie bepaalt hoe multimediale informatie wordt verwerkt en hoe we die verschillende leermaterialen moeten ontwerpen (ontwerpprincipes).
Daarna heb ik deze ontwerpprincipes dan toegepast op het lessenpakket. Hoe ik dit heb gedaan, staat uitgebreid beschreven in deze scriptie.

Naast onderzoeken in functie van het lessenpakket heb ik nog een tweetal algemene vragen onderzocht.
Een eerste onderzoek ging uit naar de aanpak van het onderwerp algoritmisch denken. Ik koos ervoor om dit te onderzoeken voor Wallonië, Verenigd Koningrijk en Nederland.
Het tweede onderzoek ging uit naar de aanpak van het lager onderwijs rond het onderwerp algoritmisch denken.

1 Onderzoeksvragen

1.1 Aanpak andere landen

1.1.1 Onderzoek

Mijn interesse ging ook uit naar de aanpak van het onderwerp 'algoritmisch denken' in andere landen.
Hoe ver staan zij daar al mee? Wat moeten de leerlingen kunnen? Wordt er een specifieke aanpak voorop gesteld? Wat zijn de doelstellingen?
Vragen genoeg die interesse opwekken en vooral om op zoek te gaan naar het antwoord.

Ik heb voor dit onderzoek mij toegespitst op de leerplannen van Wallonië, Nederland en het Verenigd Koninkrijk.
Eerst heb ik de werking van het onderwijssysteem nader bekeken om dan zo de (al dan niet) plaatsing van het vak informatica en/of ICT beter te kunnen volgen.
Als het vak gegeven wordt, heb ik dan de vakinhoud nader bekeken om dan zo, op het einde, een vergelijkende tabel te kunnen maken met Vlaanderen.

1.1.2 Wallonië

<u>Onderwijssysteem</u>

Het onderwijssysteem in Wallonië loopt vrij gelijk met het onderwijs in Vlaanderen.
De leerplicht is er van 6 tot 18 jaar en verloopt ook in 3 fasen: kleuter- (école maternelle), basis- (école primaire) en secundair onderwijs (école secondaire). Na het tweede jaar secundair onderwijs, kiezen de leerlingen uit één van de volgende drie filières:
filière générale (vergelijkbaar met het ASO), filière technique (TSO) en filière professionelle (BSO).

Toch enkele opmerkelijke verschillen:

- Op het einde van het Vlaamse basisonderwijs is er geen centraal examen (door de overheid opgesteld). In Wallonië is dat er wel. Scholen krijgen wel nog de kans om de negatieve resultaten van leerlingen bij te sturen.
- In Vlaanderen is er één onderwijsminister, in Wallonië zijn dat er drie. Een minister voor het hoger onderwijs, voor het leerplichtonderwijs en voor het onderwijs voor sociale promotie.
- Als je Vlaamse leerlingen vergelijkt met hun leeftijdsgenoten in de belangrijkste OESO-landen dan scoren ze voor wetenschappen bijzonder goed. De Waalse leerlingen doen het helemaal niet goed in de PISA-rankings. Op geen van de drie domeinen (wetenschappen, wiskunde of leesvaardigheid) raken de Waalse leerlingen in de subtop.
- Hoeveel 'kost' een Waalse leerling?
  Een leerling uit de lagere school 3000 euro (in Vlaanderen 4000 euro). In het secundair onderwijs kost een leerling 6000 euro (in Vlaanderen ruim 7000 euro), een leerling uit het buitengewoon secundair onderwijs ruim 15000 euro (in Vlaanderen 16000 euro). Een belangrijke verklaring voor de lagere kost van Waalse leerlingen ligt bij het loon van de leraren in de Franse gemeenschap. Zij verdienen ongeveer tien procent minder dan leraren in Vlaanderen.

- Een school die twintig tot honderd procent van de schoolvakken in een andere taal geeft dan de officiële taal, zou een immersieschool zijn. Behalve enkele proefprojecten tref je zulke immersiescholen niet echt aan in Vlaanderen. In Wallonië en Brussel bestaan er 300 immersiescholen.

Fig. 1 overzicht onderwijssysteem Wallonië (uit Westenwind Ryckevelde, 2014)

<u>"Programme d'informatique"</u>

In zowel de eerste, tweede als derde graad van het secundair onderwijs wordt, afhankelijk van de specifieke richting waarin men zit, informatica gegeven.

In de eerste graad omvat dit:

- algemene basisvaardigheden;
- produceren en exploiteren van documenten;
- van digitale informatiebronnen gebruikmaken;
- communiceren via e-mail.

In de tweede graad:

- tekstverwerking;
- werken met spreadsheets;
- gebruik van computer en software bij het geven van presentaties.

In de derde graad:

- presenteren en versturen van een simpele brief in tekstverwerking;
- aanwenden van de fundamenten van een spreadsheet;
- fotobewerking;
- integreren van kantoorsoftware.

In het eerste jaar van het secundair onderwijs (ASO) van de derde graad wordt het vak 'informatique de gestion' of 'IT management' onderwezen. Daarin ziet men meer de nieuwe technologieën bv. hoe men spreadsheets en gegevensbestanden opstelt.

In het technisch secundair onderwijs, specifiek de studierichting elektriciteit – electronica, bestaat het vak 'technicien en informatique' of 'informatica technicus'. Daarin wordt meer toegespitst op de technische, professionele kant van informatica. Zoals het aanleggen van een computernetwerk, de beveiliging en het monitoren van computers,…

Bij de toegepaste wetenschappen (derde graad) bestaat het vak 'science informatique' of 'informatica wetenschappen'.

Daar wordt specifiek aandacht besteed aan:

- analyseren en gebruik van Office software;
- het gebruik van een compleet informatica systeem;
- gebruik internet;
- het maken van websites;
- het creëren van multimedia;
- het ontwikkelen en implementeren van een netwerk;
- object-georiënteerd programmeren;
- beheersen van de binaire rekenkunde en Booleaanse algebra;
- maken van een algoritme en programmeren.

Vergelijking van de vakinhouden

België - Vlaanderen	België - Wallonië
Algoritmisch denken wordt enkel gegeven in de 2de graad. Inzien wat een algoritme is. Het verschil tussen een algoritme en een programma kennen. ▪ Het begrip algoritme kaderen in het dagelijks leven. ▪ Omschrijven wat een algoritme en een programma is. ▪ De verschillende stappen in het oplossen van een probleem kennen en continu toepassen bij het oplossen van problemen nl. probleemdefinitie, analyse, algoritme, programma, testen en documenteren Een probleemstelling omzetten in een werkend programma. De verschillende controlestructuren kennen en gebruiken ▪ Ongeacht de eenvoud of de complexiteit van een probleem, een analyse maken en vooraleer tot het gebruik van de computer over te gaan, minimaal voor zichzelf het principe van een oplossing formuleren. ▪ De verschillende elementen en mogelijkheden van de gebruikte ontwikkelomgeving doelgericht aanwenden. ▪ Met variabelen en constanten werken. ▪ De toekenningsoperator gebruiken. ▪ Rekenkundige-, vergelijkings- en logische operatoren integreren. ▪ De controlestructuren met hun kenmerken kennen en toepassen waaronder de sequentie, de selectie en de iteratie. ▪ De eenzijdige, tweezijdige en geneste selectie of keuze toepassen. ▪ De voorwaardelijke en begrensde herhaling gebruiken.	Algoritmisch denken wordt enkel gegeven in de 3de graad, richting toegepaste wetenschappen. ▪ object-georiënteerd programmeren; ▪ maken van een algoritme en programmeren. Concretere lesinhouden zijn niet uitgeschreven voor dit vak.

1.1.3 Verenigd Koninkrijk

Onderwijssysteem

In het Verenigd Koninkrijk is de algemene leerplicht van toepassing op alle kinderen. De leerplicht duurt elf jaar.

Het onderwijs wordt verdeeld in vier sleutelfasen:

- Sleutelfase 1 (5 tot 7 jaar);
- Sleutelfase 2 (7 tot 11 jaar);
- Sleutelfase 3 (11 tot 14 jaar);
- Sleutelfase 4 (14 tot 16 jaar).

De kinderen volgen de 1ste en 2de sleutelfase in de basisschool. De sleutelfasen 3 en 4 worden in de secundaire scholen gevolgd.

Er zijn staatsscholen, waar de kinderen geen schoolgeld hoeven te betalen en bijzondere scholen (onafhankelijke scholen).
Alle staatsscholen zijn verplicht het Nationaal Curriculum te volgen. In het Nationaal Curriculum staat welke onderwerpen moeten onderwezen worden.

Aan het einde van de 4de sleutelfase (op 16-jarige leeftijd) moeten de leerlingen het General Certificate of Secondary Education (algemeen secundair onderwijscertificaat) examen afleggen. Het GCSE examen wordt in afzonderlijke vakken afgelegd en dat door onafhankelijk exameninstituten wordt verbeterd.

Punten worden niet gegeven wel letters: A*, A, B, C, D, E, F en G. (A* is de hoogste graad die men kan behalen.)

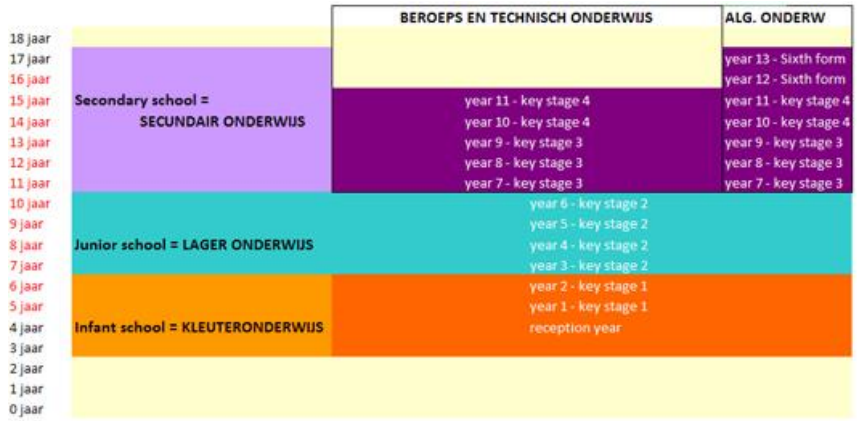


Fig. 2 overzicht onderwijssysteem Verenigd Koninkrijk (uit Westenwind Ryckevelde, 2014)

"Computing programmes"

De vakken informatica en ICT worden in het Verenigd Koninkrijk met de term "computing programmes" omschreven. Het vak wordt gedurende alle sleutelfases gegeven.

Vergelijking van de vakinhouden

België - Vlaanderen	Verenigd Koninkrijk
Algoritmisch denken wordt enkel gegeven in de 2de graad. Inzien wat een algoritme is. Het verschil tussen een algoritme en een programma kennen. ▪ Het begrip algoritme kaderen in het dagelijks leven. ▪ Omschrijven wat een algoritme en een programma is. ▪ De verschillende stappen in het oplossen van een probleem kennen en continu toepassen bij het oplossen van problemen nl. probleemdefinitie, analyse, algoritme, programma, testen en documenteren Een probleemstelling omzetten in een werkend programma. De verschillende controlestructuren kennen en gebruiken ▪ Ongeacht de eenvoud of de complexiteit van een probleem, een analyse maken en vooraleer tot het gebruik van de computer over te gaan, minimaal voor zichzelf het principe van een oplossing formuleren. ▪ De verschillende elementen en mogelijkheden van de gebruikte ontwikkelomgeving doelgericht aanwenden. ▪ Met variabelen en constanten werken. ▪ De toekenningsoperator gebruiken. ▪ Rekenkundige-, vergelijkings- en logische operatoren integreren. ▪ De controlestructuren met hun kenmerken kennen en toepassen waaronder de sequentie, de selectie en de iteratie. ▪ De eenzijdige, tweezijdige en geneste selectie of keuze toepassen. De voorwaardelijke en begrensde herhaling gebruiken.	<u>Sleutelfase 1</u> ▪ Het begrip algoritme kennen, hoe die worden uitgevoerd als programma's op digitale apparaten. ▪ Het creëren en opsporen van fouten in eenvoudige programma's. ▪ Logisch redeneren (om het gedrag van eenvoudige programma's te kunnen voorspellen). <u>Sleutelfase 2</u> ▪ Leren ontwerpen, schrijven en debuggen van programma's. ▪ Sequentie, selectie en herhaling leren gebruiken in programma's. ▪ Gebruikmaken van een logische redenering om uit te leggen hoe eenvoudige algoritmen werken. ▪ Het opsporen en corrigeren van fouten in algoritmen en programma's. <u>Sleutelfase 3</u> ▪ Het begrijpen van een aantal belangrijke algoritmen die het computer-denken weerspiegelen (bv. die voor het sorteren en zoeken); gebruik van logisch redeneren om het nut van alternatieve algoritmen voor hetzelfde probleem te vergelijken. ▪ Gebruik van twee of meer programmeertalen, ten minste één tekstuele, om een verscheidenheid van computer problemen op te lossen; gebruik maken van passende datastructuren (bv. lijsten, tabellen of arrays); ontwerpen van modulaire programma's die procedures gebruiken of functies ontwikkelen. <u>Sleutelfase 4</u> Alle leerlingen moeten de kans krijgen aspecten van informatietechnologie en informatica met voldoende diepgang te studeren om hen zo in staat te stellen hun studies naar een hoger niveau te brengen of om een carrière op te bouwen. ▪ Ontwikkelen van hun vermogen, creativiteit en kennis in de informatica, digitale media en informatietechnologie. ▪ Ontwikkelen en toepassen van hun vaardigheden om analytisch, computer gericht en probleemoplossend te denken.

1.1.4 Nederland

Onderwijssysteem

Ook het Nederlandse onderwijssysteem loopt vrij gelijk aan dit van Vlaanderen.
Er is een leerplicht voor kinderen van 5 tot 16 jaar. Daarna is men gedeeltelijk leerplichtig tot 18 jaar.
Het onderwijs verloopt er ook in 3 fasen: peuterspeelzaal (kleuteronderwijs), basisonderwijs en het voortgezet onderwijs (secundair).

Het voortgezet onderwijs richt zich vooral op het voorbereiden van de leerlingen op verschillende soorten vervolgonderwijs. Die zijn: een beroepsopleiding binnen het Middelbaar Beroeps Onderwijs (MBO), het Hoger Beroeps Onderwijs (HBO) of een wetenschappelijke opleiding aan de universiteit. Het voortgezet onderwijs is dus geen garantie op succes op de arbeidsmarkt. Daarvoor is verder studeren noodzakelijk.

Het voorgezet onderwijs wordt dan verder opgesplitst in Voorbereidend Wetenschappelijk Onderwijs (VWO) = Vlaamse ASO. Het Hoger Algemeen Voortgezet Onderwijs (HAVO) = TSO in Vlaanderen. Het Voorbereidend Middelbaar Beroeps Onderwijs (VMBO) kan met het Vlaamse BSO vergeleken worden.

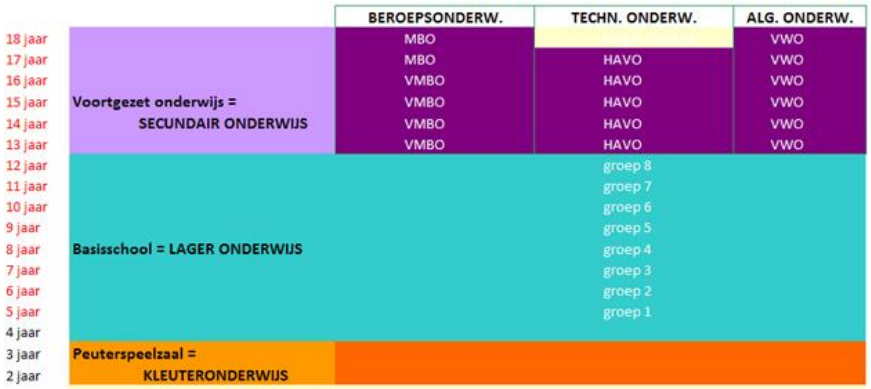


Fig. 3 overzicht onderwijssysteem Nederland (uit Westenwind Ryckevelde, 2014)

Informatica

Het vak informatica is in het Nederlandse schoolsysteem een keuzevak en wordt slechts door 60% van de scholen aangeboden.
Voor informatica wordt geen centraal examen georganiseerd, maar wordt wel afgesloten met een schoolexamen. Informatica is dan ook geen vak dat voorbereidt op een informaticastudie in het wetenschappelijk onderwijs of een ict-opleiding in het hoger beroepsonderwijs, maar een algemeen vormend vak.

Bij het maken van het eindexamen voor informatica in Nederland worden vooral de bredere onderwerpen getoetst.

In 2007 werden de beschikbare uren voor informatica uitgebreid en de eindtermen werden aangepast aan de andere (buur)landen.

Algemeen zijn de doelstellingen van een schoolvak in het voortgezet onderwijs in meer of mindere mate afgeleid van de inhoud van het betreffende vak in het hoger onderwijs. Dat geldt ook voor het vak informatica in de bovenbouw van het Nederlandse voortgezet onderwijs.

Sinds 2007, toen informatica meer lesuren kreeg, is algoritmisch denken een belangrijk onderdeel van het vak.
Maar in de eindtermen vind je nog altijd maar weinig terug over algoritmisch denken. Leerlingen kunnen op meerdere onderdelen de leerstof niet goed toepassen. Ze hebben geleerd *over* programmeren (= het kennen), maar ze hebben niet geleerd *om* te programmeren (= het kunnen).

Vergelijking van de vakinhouden

België - Vlaanderen	Nederland
Algoritmisch denken wordt enkel gegeven in de 2de graad. Inzien wat een algoritme is. Het verschil tussen een algoritme en een programma kennen. ▪ Het begrip algoritme kaderen in het dagelijks leven. ▪ Omschrijven wat een algoritme en een programma is. ▪ De verschillende stappen in het oplossen van een probleem kennen en continu toepassen bij het oplossen van problemen nl. probleemdefinitie, analyse, algoritme, programma, testen en documenteren Een probleemstelling omzetten in een werkend programma. De verschillende controlestructuren kennen en gebruiken ▪ Ongeacht de eenvoud of de complexiteit van een probleem, een analyse maken en vooraleer tot het gebruik van de computer over te gaan, minimaal voor zichzelf het principe van een oplossing formuleren. ▪ De verschillende elementen en mogelijkheden van de gebruikte ontwikkelomgeving doelgericht aanwenden. ▪ Met variabelen en constanten werken. ▪ De toekenningsoperator gebruiken. ▪ Rekenkundige-, vergelijkings- en logische operatoren integreren. ▪ De controlestructuren met hun kenmerken kennen en toepassen waaronder de sequentie, de selectie en de iteratie. ▪ De eenzijdige, tweezijdige en geneste selectie of keuze toepassen. ▪ De voorwaardelijke en begrensde herhaling gebruiken.	▪ Beheersen van eenvoudige datatypen, programmastructuren en programmeertechnieken. ▪ Beheersen van de kernbegrippen in het objectgericht programmeren (object, objectklasse, attributen en methodes). ▪ Toepassingen op het objectgericht programmeren of visueel programma. ▪ Leren de computerprogramma's lezen en beschrijven in eigen woorden. ▪ Leren de verschillende programma's en hun programmeertaal herkennen. ▪ Een programma schrijven en verbeteren. ▪ Zelf code schrijven.

1.2 Lager onderwijs

1.2.1 Onderzoek

Na mijn onderzoek naar de aanpak van het onderwerp 'algoritmisch denken' in andere landen, ben ik ook benieuwd of al enige kennis wordt opgedaan rond algoritmisch denken in het lager onderwijs.

Voor een leerkracht ICT-informatica in het secundair onderwijs is het belangrijk om te weten welke beginsituatie de leerlingen hebben uit het basisonderwijs. Hieraan koppel ik dan ook de vraag: welke kennis hebben de leerlingen die uit het lager onderwijs komen in verband met algoritmisch denken?

Ik ben mijn onderzoek gestart door de eindtermen ICT te bekijken in het lager onderwijs. Aangezien er geen sprake is van algoritmisch denken in de eindtermen (zie verder) heb ik daarna een enquête opgesteld om een aantal zaken te peilen bij leerkrachten lager onderwijs (zie 'enquête').

1.2.2 Eindtermen

Hieronder de eindtermen ICT voor het basisonderwijs.

De leerlingen…

- hebben een positieve houding tegenover ict en zijn bereid ict te gebruiken om hen te ondersteunen bij het leren;
- gebruiken ict op een veilige, verantwoorde en doelmatige manier;
- kunnen zelfstandig oefenen in en door ict ondersteunde leeromgeving;
- kunnen zelfstandig leren in en door ict ondersteunde leeromgeving;
- kunnen ict gebruiken om eigen ideeën creatief vorm te geven;
- kunnen met behulp van ict voor hen bestemde digitale informatie opzoeken, verwerken en bewaren;
- kunnen ict gebruiken bij het voorstellen van informatie aan anderen;
- kunnen ict gebruiken om op een veilige, verantwoorde en doelmatige manier te communiceren.

In de eindtermen van het basisonderwijs wordt geen algoritmisch denken vermeld.
Ook in het raamplan van de eerste graad secundair onderwijs wordt het niet vermeld.

Er wordt dus niet verwacht dat de leerlingen kennis hebben van of kennis maken met algoritmisch denken.

1.2.3 Enquête

Omdat in de eindtermen niet gesproken wordt over algoritmisch denken, was ik wel eens benieuwd of leerkrachten dat effectief dan niet geven aan de kinderen.
Beide enquêtes vind je in de bijlagen van deze scriptie.

Voor deze onderzoekvraag heb ik twee enquêtes opgesteld:

- "Programmeren in de klas": Deze enquête is gericht op leerlingen secundair onderwijs en studenten lerarenopleiding met als onderwijsvak informatica.

  De bedoeling van deze enquête is om te peilen wat de (voor)kennis van de leerlingen en studenten is in verband met algoritmisch denken. Wat vinden zij van de aanpak van dit onderwerp? Krijgen de leerlingen deze lessen effectief? …

- "Programmeren in de klas (leerkrachten)": Deze enquête is gericht op leerkrachten basis- en secundair onderwijs.

  De bedoeling van deze enquête is om na te gaan of de leerkrachten genoeg kennis hebben om dit aan te brengen? Waar hebben de leerkrachten deze kennis opgedaan? Welke programmeeromgeving wordt er gebruikt in hun lessen? Hoe zit het met de kennis van de leerkrachten in het lager onderwijs? Zou het mogelijk zijn om algoritmisch denken al te geven in het lager onderwijs? …

Beide enquêtes bestaan uit drie grote delen:

- Wat algemene informatie over de persoon die de enquête invult.
- Wat is de voorkennis en/of kennis op het vlak van informatica en programmeren?
- Wat specifiekere vragen in verband met het onderdeel programmeren.

1.2.4 Verwerking enquête: leerkrachten

De enquête kan je vinden op volgende link:
https://limesurvey.vives.be/surveystudent/index.php/491556/lang-nl

Samenvatting respons	
Volledige respons:	37
Onvolledige respons:	3
Aantal respons:	40

Fig. 4 respons enquête 1 (LimeSurvey VIVES, 2014)

Ter informatie: ik werk in het vervolg van de resultaten enkel verder met de volledige responsen (37).

- Geslacht

Er vulden ongeveer evenveel mannelijke als vrouwelijke leerkrachten de enquête in.

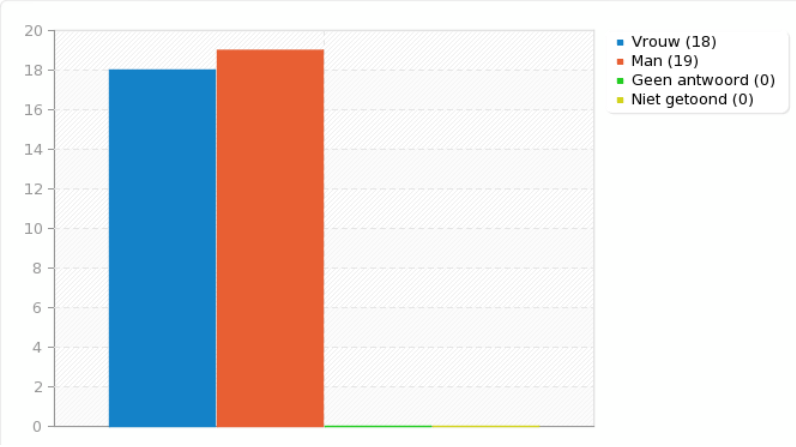


Fig. 5 diagram geslacht (LimeSurvey VIVES, 2014)

- Leeftijd

De gemiddelde leeftijd is 34 jaar van de leerkrachten die de enquête invulden. De jongste deelnemer was 28 jaar en de oudste deelnemer 56 jaar.
Besluit: de populatie die deelnam aan de enquête was eerder de jongere generatie maar met al wat enige ervaring.

Berekening	Resultaat
Telling	37
Som	1274.0000000000
Standaardafwijking	11.35
Gemiddelde	34.43
Minimum	1.0000000000
1e kwartiel (K1)	28
2e kwartiel (mediaan)	33
3e kwartiel (K3)	42.5
Maximum	56.0000000000

Fig. 6 samenvatting leeftijd (LimeSurvey VIVES, 2014)

- Verdeling

Het is ook belangrijke om de verdeling tussen de leerkrachten secundair- en basisonderwijs binnen de enquête te weten.
Er is een mooie spreiding tussen beide. Er zijn 15 leerkrachten lager onderwijs en 26 leerkrachten secundair onderwijs waar de meerderheid in de tweede graad lesgeeft, waar ook vooral programmeren gegeven wordt.
De 14 leerkrachten binnen de categorie "andere" zijn leerkrachten die bovenop het lesgeven in het lager- of secundair onderwijs ook zorgcoördinator of leerkracht in het volwassen onderwijs zijn.

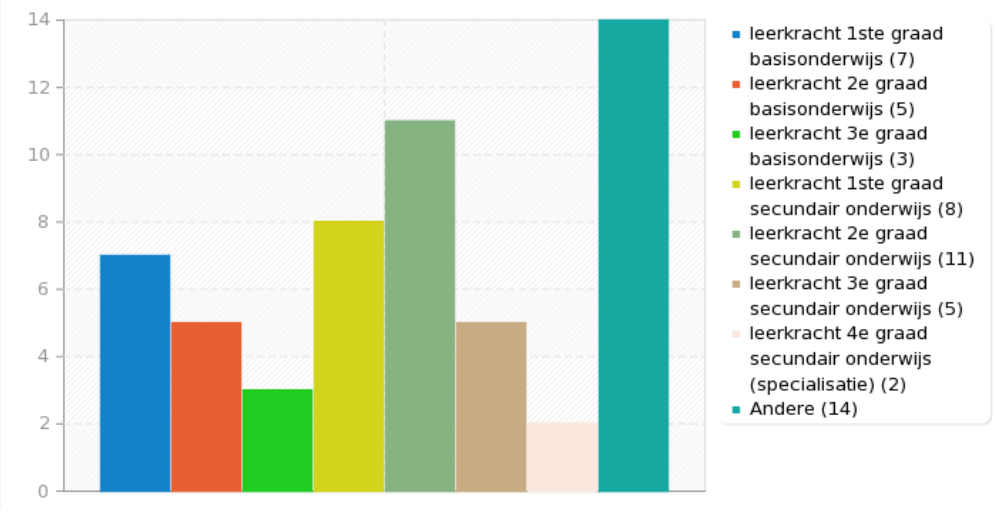


Fig. 7 diagram verdeling (LimeSurvey VIVES, 2014)

- Ben je vertrouwd met ICT en/of informatica?

De meerderheid van de leerkrachten gebruikt echt wel informatica. De periode waarin de drempel tot het gebruik van informatica groot was, is voorgoed voorbij. Het is dan ook niet meer weg te denken uit het onderwijs. Elke leerkracht heeft dus wel enige kennis van informatica.

Antwoord	Telling	Percentage
Ja, ik ben expert want ik maak vaak gebruik van tekstverwerkingsprogramma's, rekenbladen, presentaties, leerpaden, iPads, interactief bord, ... (A1)	20	54.05%
Ja, ik bezit vooral de basiskennis. Ik maak vaak gebruik van tekstverwerkingsprogramma's (bv. Word), rekenbladen (bv. Excel), presentaties (bv. PowerPoint). (A2)	14	37.84%
Nee, ik gebruik enkel de computer als het moet (als ik geen andere keuze heb). (A3)	3	8.11%
Nee, ik gebruik zelden tot nooit de computer. (A4)	0	0.00%
Geen antwoord	0	0.00%
Niet getoond	0	0.00%

Fig. 8 samenvatting gebruik ICT/informatica (LimeSurvey VIVES, 2014)

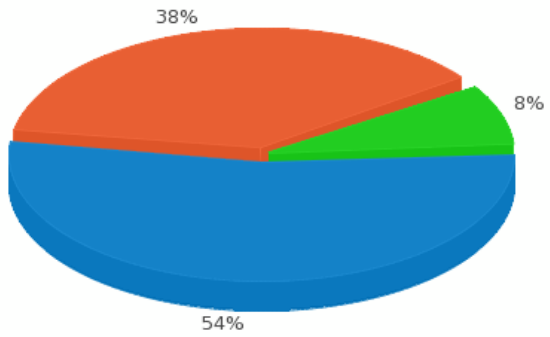


Fig. 9 diagram gebruik ICT/informatica (LimeSurvey VIVES, 2014)

Ik heb ook eens de vergelijking gemaakt tussen het gebruik van informatica tussen de leerkrachten basis- en secundair onderwijs.
De leerkrachten uit het secundair onderwijs zijn vooral experts en gebruiken ook minder gekende software. Ik denk dat dit vooral wijst omdat de leerkrachten uit het secundair onderwijs, die de enquête hebben ingevuld, vooral leerkrachten informatica zijn en die zijn dus meer gespecialiseerd. Leerkrachten uit het lager onderwijs moeten ook meer basiskennis informatica/ICT aanleren aan de leerlingen en zo is dat wel logisch.

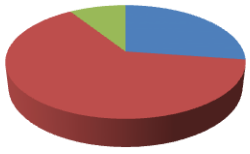


Fig. 10 diagram lager onderwijs gebruik (LimeSurvey VIVES, 2014)

Legende:

- blauw: expert
- rood: basiskennis
- groen: enkel als het moet

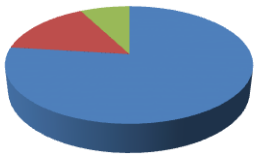


Fig. 11 diagram secundair onderwijs gebruik (LimeSurvey VIVES, 2014)

- Ben je vertrouwd met ICT en/of informatica? (Geef jezelf punten op 10.)

Deze vraag peilt naar de kennis van de leerkrachten. In de diagrammen hieronder maakte ik een onderscheid tussen leerkrachten basis- en secundair onderwijs. De leerkrachten basisonderwijs staan aangeduid met een blauwe balk en de leerkrachten secundair onderwijs met een rode balk. Op de horizontale as (x-as) zie je de puntenschaal van 1 t.e.m.10. Op de verticale as (y-as) zie je het aantal leerkrachten in procenten.

In de diagrammen zie je dat de kennis bij beide groepen (leerkrachten basis- en secundair onderwijs) een stuk lager ligt bij programmeren ten opzichte van de andere onderwerpen. Er is wel te zien dat de kennis van de leerkrachten uit het secundair onderwijs meer gespreid is rond het onderwerp programmeren, de kennis over dit onderwerp van de leerkrachten uit het basisonderwijs is bij een lage score gegroepeerd.

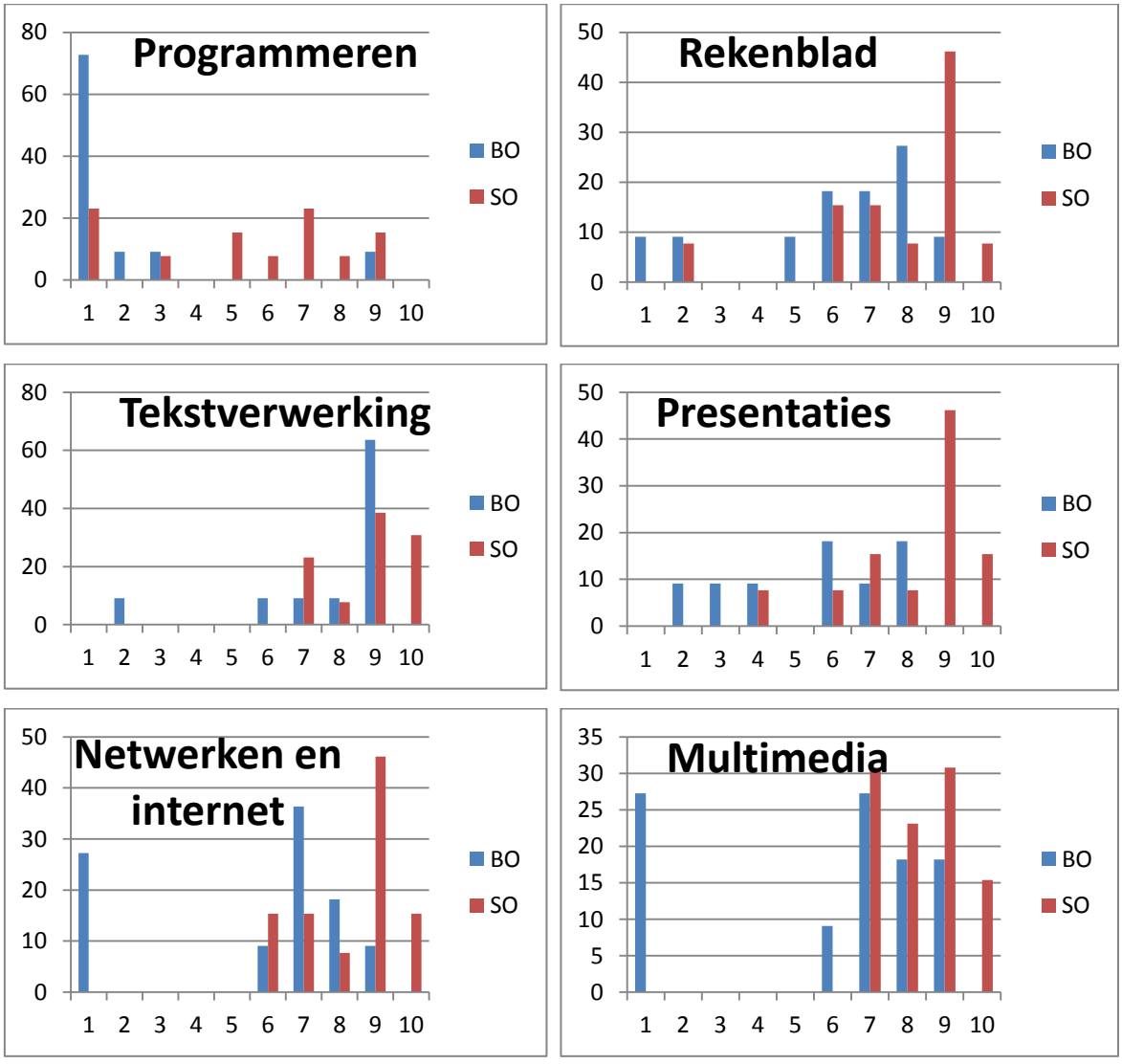


Fig. 12 diagrammen kennis (LimeSurvey VIVES, 2014)

- Weet je wat de term 'programmeren' inhoudt?
- Gebruik je deze materie (programmeren) in de klas?

De leerkrachten uit het lager onderwijs weten amper wat de term programmeren betekent. Er zijn slechts 3 van de 11 leerkrachten die een programma en een programmeertaal kennen. De 3 leerkrachten die er kennis van hebben gebruiken deze materie niet in de klas. Een initiatie van programmeren wordt dus niet echt onderwezen in de lagere school.

Antwoord	Telling
Ja (ik programmeer vaak in meerdere programmeertalen) (A1)	0
Ongeveer (ik weet wat het begrip betekent en heb kennis van één programma en één programmeertaal) (A2)	3
Nee (ik weet niet wat het betekent/vaag wat het betekent) (A4)	8
Geen antwoord	0
Niet getoond	0

Fig. 13 samenvatting term programmeren (LimeSurvey VIVES, 2014)

Antwoord	Telling	Percentage
Ja (Y)	0	0.00%
Nee (N)	3	27.27%

Fig. 14 samenvatting gebruik in klas (LimeSurvey VIVES, 2014)

- Vind je dat de leerlingen in het basisonderwijs reeds deze leerstof moeten kennen?

De leerkrachten in beide groepen hebben geen unaniem antwoord op deze vraag. Ongeveer de helft vindt het een goed idee om een basis te leggen in het basisonderwijs, de andere helft vindt het goed zoals het nu is (met andere woorden de leerstof alleen in het secundair onderwijs onderwijzen). Drie leerkrachten vinden dat het onderwerp totaal niet moet gegeven worden.

Antwoord	Telling	Percentage
Ja, een basis leggen in het basisonderwijs is een goed idee. (A1)	16	43.24%
Nee, beter in het secundair onderwijs. (A2)	18	48.65%
Nee, totaal niet nuttig in zowel het secundair onderwijs als in het basisonderwijs. (A3)	3	8.11%

Fig. 15 samenvatting lkr basis- en secundair onderwijs (LimeSurvey VIVES, 2014)

Ik was wel eens benieuwd of het vooral leerkrachten in het basisonderwijs zijn die vinden dat de leerstof alleen moet gegeven worden in het secundair onderwijs. Zoals je kan zien in de diagrammen hieronder kiezen deze leerkrachten ervoor om de leerstof te onderwijzen in het secundair onderwijs. Ik denk dat we hiervoor de verklaring moeten gaan zoeken bij de kennis van deze leerkrachten over dit onderwerp (zie vorige bladzijden).

Antwoord	Telling	Percentage
Ja, een basis leggen in het basisonderwijs is een goed idee. (A1)	3	27.27%
Nee, beter in het secundair onderwijs. (A2)	6	54.55%
Nee, totaal niet nuttig in zowel het secundair onderwijs als in het basisonderwijs. (A3)	2	18.18%

Fig. 16 samenvatting lkr basisonderwijs (LimeSurvey VIVES, 2014)

- Vind je dat leerlingen in het basisonderwijs reeds deze leerstof moeten kennen voor later?
- Vind je dat leerlingen in het secundair onderwijs deze leerstof moeten kennen als een deel van hun algemene basiskennis?

Met beide bovenstaande vragen is de meerderheid van de leerkrachten van beide groepen het eens. Deze antwoorden zijn toch echt wel in tegenstrijd met de vorige vraag. Nogmaals een bevestiging dat het niet onderwijzen van het onderwerp echt wel te maken heeft met de kennis die de leerkrachten niet hebben. Het ligt zeker niet aan het feit dat de leerkrachten het niet willen geven omdat ze het nut er niet van inzien.

Antwoord	Telling	Percentage
Ja (Y)	26	70.27%
Nee (N)	11	29.73%

Fig. 17 samenvatting basisonderwijs (LimeSurvey VIVES, 2014)

Antwoord	Telling	Percentage
Ja (Y)	24	64.86%
Nee (N)	13	35.14%

Fig. 18 samenvatting secundair onderwijs (LimeSurvey VIVES, 2014)

1.2.5 Verwerking enquête: leerlingen en studenten

De enquête kan je vinden op volgende link:
https://limesurvey.vives.be/surveystudent/index.php/212223/lang-nl

Samenvatting respons	
Volledige respons:	37
Onvolledige respons:	4
Aantal respons:	41

Fig. 19 respons enquête 2 (LimeSurvey VIVES, 2014)

Ter informatie: ik werkte in het vervolg van de resultaten enkel verder met de volledige responsen (37).

- Geslacht

Antwoord	Telling	Percentage
Vrouw (F)	14	37.84%
Man (M)	23	62.16%

Fig. 20 samenvatting geslacht (LimeSurvey VIVES, 2014)

- Leeftijd

De gemiddelde leeftijd is 16 jaar. De jongste deelnemer was 12 jaar en de oudste deelnemer 24 jaar.

Berekening	Resultaat
Telling	37
Som	606.0000000000
Standaardafwijking	2.77
Gemiddelde	16.38
Minimum	12.0000000000
1e kwartiel (K1)	15
2e kwartiel (mediaan)	15
3e kwartiel (K3)	17
Maximum	24.0000000000

Fig. 21 samenvatting leeftijd (LimeSurvey VIVES, 2014)

- Verdeling

Het is ook belangrijke om de verdeling tussen de leerlingen en studenten binnen de enquête te weten.
Er is een niet evenwichtige spreiding tussen beide. Er vulden 30 leerlingen en 7 studenten de enquête in.

Antwoord	Telling	Percentage
leerling secundair onderwijs (A1)	30	81.08%
student lerarenopleiding (onderwijsvak: informatica) (A2)	7	18.92%

Fig. 22 samenvatting verdeling (LimeSurvey VIVES, 2014)

Ter informatie: In het vervolg van de enquête werk ik verder met de volledige resultaten van leerlingen. De bevraagde populatie (7) is te klein om besluiten te nemen op basis van de studenten.

- Kennis (Geef jezelf punten op 10.)

In de diagrammen zie je op de horizontale as (x-as) de puntenschaal van 1 t.e.m.10. Op de verticale as (y-as) zie je het aantal leerlingen in procenten.

De kennis van de leerlingen in het secundair onderwijs is relatief hoog. Je zou verwachten dat de kennis van de leerlingen rond het onderwerp programmeren ook gegroepeerd zou zijn in de lagere scores. Zoals je kan zien op de diagrammen is dat niet het geval. De scores liggen wel meer verdeeld maar toch nog vrij hoog. Blijkbaar leggen de leerkrachten uit het secundair onderwijs echt wel nog de klemtoon op programmeren zodat de leerlingen dit als minder moeilijk ervaren.

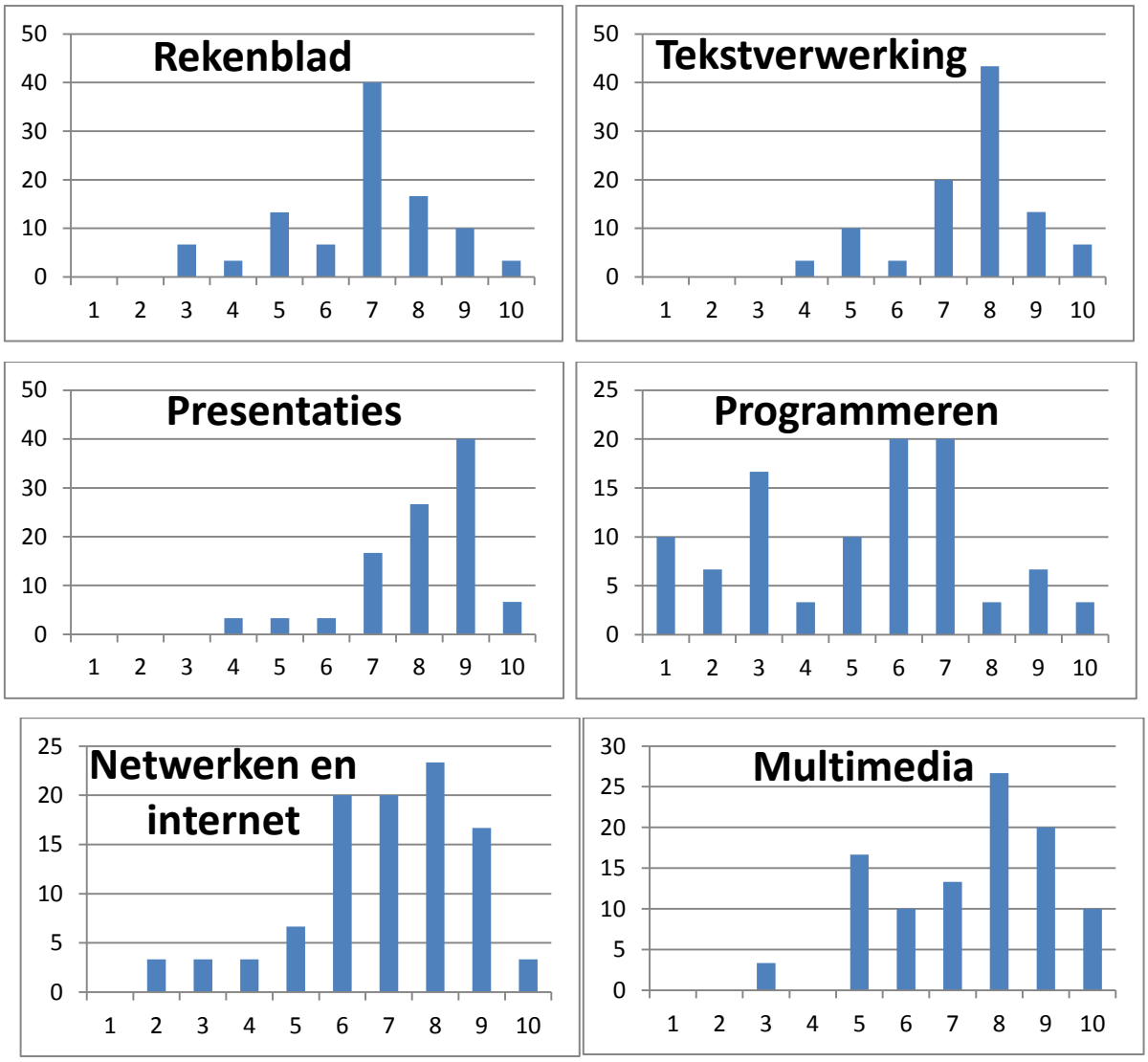


Fig. 23 diagrammen kennis leerlingen (LimeSurvey VIVES, 2014)

- Vink de begrippen aan die je kent.

Hieruit kunnen we afleiden dat de leerlingen de meeste begrippen onthouden. Het valt op dat functies niet in het leerplan staat maar toch wordt gegeven aangezien de leerlingen het begrip herkennen/kennen ofwel denken de leerlingen hier aan functies in een rekenblad…

Klassen is ook een voorbeeld die niet in het leerplan vermeld staat maar dat de leerlingen toch herkennen. Het is vreemd dat maar 10% van de leerlingen iteratie kent, nochtans moeten de leerlingen dat eigenlijk kennen volgens het leerplan.

Antwoord	Telling	Percentage
variabelen (SQ001)	14	46.67%
sequentie (SQ002)	7	23.33%
selectie (SQ003)	20	66.67%
iteratie (SQ004)	3	10.00%
functies (SQ005)	22	73.33%
arrays (SQ006)	6	20.00%
overerving (SQ007)	3	10.00%
klassen (SQ008)	16	53.33%

Fig. 24 samenvatting begrippen (LimeSurvey VIVES, 2014)

- Krijg je lessen over/rond programmeren?

De meerderheid van de leerlingen krijgt lessen rond programmeren. 17% van de leerlingen krijgen geen lessen rond programmeren. Dit percentage is relatief hoog, maar dit zou kunnen zijn omdat ook leerlingen uit de eerste graad de enquête hebben ingevuld.

Antwoord	Telling	Percentage
ja (A1)	19	63.33%
nee (A2)	5	16.67%

Fig. 25 samenvatting lessen (LimeSurvey VIVES, 2014)

- Zou het handig zijn geweest, als je deze kennis reeds vroeger zou opgedaan hebben?

Ongeveer 69% van de leerlingen wil vroeger kennismaken met programmeren. Er is wel enige onenigheid wanneer dan het programmeren zou moeten gestart worden. 27% van de leerlingen vindt dat programmeren totaal niet past binnen het secundair onderwijs wat toch ook wel veel is. Een verklaring hiervoor zou kunnen zijn dat programmeren niet goed aangepakt wordt binnen de lessen. De leerlingen zien soms het nut er niet van in of vinden het echt wel moeilijk.

Antwoord	Telling	Percentage
Ja, van mij mag men reeds leren programmeren in het basisonderwijs (A1)	10	33.33%
Ja, 'programmeren' zou moeten verplicht zijn vanaf 1ste graad secundair onderwijs (A2)	11	36.67%
Nee, deze kennis had ik al (A3)	1	3.33%
Nee, 'programmeren' heb je slechts nodig na het secundair onderwijs (A4)	8	26.67%

Fig. 26 samenvatting kennis vroeger (LimeSurvey VIVES, 2014)

1.2.6 Initiatieven

Aangezien algoritmisch denken in het lager onderwijs en in de eerste graad secundair onderwijs niet tot amper wordt gegeven, heb ik een aantal initiatieven/mogelijkheden opgezocht waar deze leerlingen terecht kunnen aangezien toch 69% van de leerlingen vroeger wil kennismaken met programmeren.

Codekinderen

Fig. 27 logo Codekinderen (Kennisnet, 2013)

Codekinderen is een organisatie die leerlingen van het basisonderwijs de kans geven om hun digitale talenten te ontdekken. De kinderen maken kennis met digitale spelen tot echt programmeren en code schrijven.

CoderDojo

Fig. 28 logo CoderDojo (CoderDojo, 2011)

CoderDojo is een organisatie die wereldwijd gratis bijeenkomsten organiseert voor jongeren tussen de 7 en 18 jaar.
Deze Ierse organisatie werd opgericht door James Whelton en Bill Liao.
Tijdens deze Dojo's (=de lessen) leren ze er programmeren. Op die bijeenkomsten leren de jongeren programmeren, websites maken, applicaties en games ontwikkelen. Ze ontmoeten zo andere jongeren en laten elkaar zien waaraan ze gewerkt hebben. Deze organisatie maakt van ontwikkelen en programmeren een leuke, gezellige en toffe leerervaring.
De werking van deze Dojo's berust volledig op vrijwilligers.

Software

Er worden steeds meer software ontworpen om kinderen warm te maken voor algoritmisch denken.
Sommige software wordt zelfs zo ontworpen zodat kinderen alleen aan de slag kunnen (spelenderwijs leren).

Kodu Game Lab is een voorbeeld van zo'n software.
Het is een spel waarbij niet het spelen maar het 'maken van spelen' voorop staat. Je ontwerpt je eigen spelwereld en spelregels. In de eerste niveaus leer je hoe je zelf uitdagende spelen kan opbouwen. Eens deze niveaus voltooid zijn, kunnen de kinderen zelf leuke spelen en werelden ontwerpen.

Fig. 29 spelomgeving Kodu (Microsoft)

1.2.7 Estse basisonderwijs

In het artikel hieronder kun je lezen dat de leerlingen in het basisonderwijs in Estland kennismaken met algoritmisch denken. Ze gaan zelfs nog een stapje verder en geven de kleuters al enkele basislessen informatica. Estland staat dus al een heel eind verder.

vandaag.be 24/7

Kinderen leren programmeren in Estse basisonderwijs

woensdag 13 maart 2013

In Estland krijgen kinderen al van in de lagere school lessen in computervaardigheid. Het aandeel dat informatica er opeist wordt stilaan gelijkwaardig aan dat van rekenen en taal. Het onderwijs werkt er toekomstgericht. Websites en applicaties bouwen wordt een basisvaardigheid. Dat weet Knack.

Aan het Pelgulinna Gümnaasium in de hoofstad Talinn krijgen kleuters al één keer per week een basisles informatica. Vanaf het vierde leerjaar krijgen ze te maken met Robotics en enkele jaren later leren ze al websites en mobiele applicaties bouwen.

Beter gewapend

Tientallen scholen in Estland gebruiken lesprogramma van de Tiger Leap Foundation. Ook basisschoolleraren worden door hen opgeleid. In de Baltische staat, waar de bevolking elektronisch stemt en zijn belastingaangifte invult, is men er rotsvast van overtuigd dat wie technologie in de vingers heeft, de toekomst beter gewapend tegemoet gaat.

Daarom heeft de Estse regering het fiscaal aantrekkelijk gemaakt om een ict-bedrijfje op te richten. Uit een andere hoek weerklinkt de kritiek dat de digitaal onderlegde kinderen wel eens de hackers van morgen zouden kunnen worden. Ave Lauringson, projectleider bij de Tiger Leap Foundation doet het af als onzin. De kinderen worden ook de normen en waarden van de maatschappij bijgebracht.

Discussie

In Vlaanderen woedde onlangs nog een discussie over het afschaffen van de traditionele informaticalessen. Ook Facebook-oprichter Mark Zuckerberg pleitte er al voor om jonge kinderen vertrouwd te maken met programmeren.

MCe

Fig. 30 artikel Estse basisonderwijs (Vandaag.be, 13 maart 2013)

1.2.8 Besluit

Om op de onderzoeksvraag "Welke kennis hebben de leerlingen die uit het lager onderwijs komen in verband met algoritmisch denken?" te kunnen antwoorden is de enquête vrij unaniem. Aangezien de leerkrachten uit het basisonderwijs zeer weinig kennis hebben over het onderwerp, wordt programmeren ook niet gegeven in het basisonderwijs. De leerlingen die uit het basisonderwijs komen hebben dus niet echt een voorkennis over dit onderwerp.

In het secundair onderwijs wordt het onderwerp toch vrij veel gegeven. De leerlingen herinneren zich heel wat termen die met het onderwerp te maken hebben. Het is zelfs opmerkelijk dat de leerlingen een aantal termen, zoals klassen en functies, kennen die niet in het leerplan vermeld staan. Daar tegenover herkent maar 10% het begrip iteratie, wat toch een cruciaal begrip is. Daaruit zou je kunnen afleiden dat de invulling van het onderwerp toch niet is zoals het leerplan voorlegt.

Een kleine meerderheid van de leerkrachten en een groep leerlingen (33%) vinden dat programmeren in het basisonderwijs zou moeten gegeven worden. Terwijl de andere groep leerlingen (37%) vinden dat programmeren zou moeten gegeven worden vanaf de eerste graad secundair onderwijs.
27% van de leerlingen vinden dat programmeren niet thuishoort in het secundair onderwijs met de gedachte "je hebt het toch maar na het secundair onderwijs nodig". Om ook deze groep leerlingen te kunnen motiveren in de lessen moet de aanpak van programmeren nog wat bijgesleuteld worden. Blijkbaar is het voor deze leerlingen niet duidelijk waarom programmeren belangrijk is, ook in het secundair onderwijs.

1.3 Applicaties

1.3.1 Onderzoek

Om een lessenpakket, met de iPad, te creëren rond algoritmisch denken, heb je uiteraard een programmeeromgeving nodig. Een moeilijke keuze die ik meteen in een onderzoeksvraag gegoten heb. Welke applicatie is er het meest geschikt om dit lessenpakket rond op te bouwen?

Ik heb dus een aantal applicaties uitgekozen en deze geëvalueerd. Om deze applicaties te evalueren, heb ik een evaluatiefiche gemaakt.
In de volgende delen wordt zowel de uitleg van de evaluatiefiche gegeven als de evaluatie van de applicaties.

1.3.2 Leren

Wat is leren?
Leren is een constructief, cumulatief, zelfgestuurd, doelgericht, contextgebonden, coöperatief en individueel verschillend proces van kennisverwerving, betekenisgeving en vaardigheidsontwikkeling. (De Corte, 1996)

1.3.3 Kenmerken krachtige leeromgeving

We leren in een krachtige leeromgeving wanneer aan onderstaande kenmerken is voldaan.

Constructief
De leerling denkt actief en voert cognitieve verwerkingsactiviteiten uit. Ze leert zelf kennis op te bouwen.
Voorbeeld: memoriseren, structureren, analyseren, …

Cumulatief
De leerling bouwt verder op wat ze al weten (voorkennis).

Zelfgestuurd
De leerling is actief in het beheren van hun eigen leerproces (metacognitieve activiteiten).
Voorbeeld: bijsturen, vooruitgang in het oog houden, …

Doelgericht (intentioneel leren)
De leerling leert om een doel te bereiken.
Voorbeeld: vooraf de doelstellingen formuleren, vragen wat hun doel is, …

Contextgebonden (gesitueerd leren)
Het zijn niet alleen de denk- en leerprocessen in ons hoofd. De denk- en leerprocessen zijn verbonden met specifieke activiteiten en de context of cultuur waarin de leerling zich bevindt.
Voorbeeld: reële situaties gebruiken, toepassen in de praktijk, …

Coöperatief
De leerling leert in een bepaalde sociale en culturele context.
Voorbeeld: peer tutoring, gemengde groepen, …

Individueel verschillend
Dezelfde instructie leidt niet noodzakelijk tot hetzelfde leren bij verschillende leerlingen.
Voorbeeld: omgaan met de verschillende leerstijlen, leerniveaus, …

1.3.4 Evaluatiecriteria op basis van de krachtige leeromgeving

Om de applicaties te evalueren heb ik een evaluatiefiche gemaakt, die ik opstel per applicatie. In deze evaluatiefiche heb ik ook rekening gehouden met de kenmerken van de krachtige leeromgeving.

Constructief
Dit kenmerk heb ik letterlijk opgenomen in de evaluatiefiche (evaluatiecriteria: "constructief").
Maakt de applicatie gebruik van vragen van een hoog denkniveau?

Cumulatief
Dit kenmerk heb ik letterlijk opgenomen in de evaluatiefiche (evaluatiecriteria: "cumulatief").
Biedt de applicatie flexibiliteit om de inhoud en instellingen aan te passen aan de leerlingen?
Kunnen de instellingen worden aangepast aan de beginsituatie van elke leerling?
Kunnen de leerlingen terug inpikken waar ze zijn gekomen?

Zelfgestuurd
Dit kenmerk heb ik letterlijk opgenomen in de evaluatiefiche (evaluatiecriteria: "zelfgestuurd").
Krijgen de leerling feedback op wat ze gemaakt hebben? Wordt specifieke feedback gegeven, zodat leerlingen weten waar hun fout precies zit?
Is er een zelfcontrole voorzien? Kunnen de leerlingen hun oefening verbeteren zonder dat de leerkracht hierin moet tussenkomen?

Doelgericht
Dit kenmerk heb ik letterlijk opgenomen in de evaluatiefiche (evaluatiecriteria: "doelgericht").
Is de applicatie geschikt om de doelen te bereiken?
Dit kenmerk komt ook terug in het lessenpakket zelf. De leerlingen krijgen bij het begin van elk hoofdstuk de doelenlijst te zien. Het is de bedoeling dat de leerlingen zo het doel (per hoofdstuk) voor ogen krijgen.

Contextgebonden
Dit kenmerk werd niet opgenomen in de evaluatiefiche door onderstaande reden.
Elk programma heeft linken met de leefwereld en de realiteit van de jongeren.
Voorbeelden: robot/auto simuleren, berekening van BMI, …

Coöperatief
Dit kenmerk komt in alle applicaties zeker voor.
De leerlingen leren aan de hand van interactie. Zowel met de andere leerlingen als in interactie met de iPad/applicatie zelf.
Dit kenmerk komt ook terug in het lessenpakket zelf. Leerlingen moeten vaak met elkaar samenwerken, resultaten vergelijken, …

Individueel verschillend
Dit kenmerk komt onder andere terug op de evaluatiefiche bij het evaluatiecriteria "delen". Kunnen de leerlingen hun projecten delen met de leerkrachten? Heeft de leerkracht een zicht op wat de leerlingen al bereikt hebben en op wat ze nog moeten bereiken (prestaties)?

1.3.5 Andere evaluatiecriteria

Gebruiksvriendelijkheid

Het is belangrijk dat de leerkracht niet veel tijd moet besteden aan hoe ze de applicatie precies moeten gebruiken. De klemtoon moet kunnen gelegd worden op de inhoud zonder dat dit gehinderd wordt door 'hoe werk ik met deze applicatie'.

Taal

Dit aspect zou van groot belang geweest zijn een aantal jaren geleden. Aangezien men nu de talenkennis van leerlingen wil integreren in bepaalde vakken zou het een voordeel kunnen zijn dat de applicatie Frans- of Engelstalig is.

Tot een op de vijf lessen in secundair in andere taal

8 november 2 reacties

Vind ik leuk 42 Tweeten 2 Pin it

Secundaire scholen mogen vanaf volgend schooljaar tot 20 procent van de niet-taalvakken in het Frans, Engels of Duits geven. Het decreet daarover werd al in de zomervakantie goedgekeurd maar kwam deze week weer in de actualiteit. Wat houdt het systeem concreet in?

Officieel heet het systeem Content and Language Integrated Learning (CLIL). Negen proeftuinscholen experimenteren er al mee sinds 2007. Vanaf 1 september kunnen àlle secundaire scholen ermee aan de slag.

Fig. 31 lessen in andere taal (Klasse, 8 november 2013)

Compatibiliteit

Het zou handig zijn als de applicatie niet afhankelijk is aan een bepaalde store. Zo kunnen alle leerlingen die thuis een tablet hebben oefenen.
De school heeft dan ook de keuze welke tablet ze kopen.

Helpfunctie

Dit criteria heb ik erin gestopt omdat ik het belangrijk vind dat leerlingen ook zelf problemen moeten kunnen oplossen zonder hulp van de leerkracht. Ze moeten leren een handleiding doornemen of vragen stellen op discussiefora om tot een oplossing te komen. De leerkracht moet tijdens het oefenmoment als hulplijn/coach ingeschakeld worden. Hiermee werk ik aan het aspect 'zelfredzaamheid' binnen het vak informatica.

Om de leerlingen op een aanvaardbare wijze door één leraar effectief te begeleiden bij het gebruik van systeem- of toepassingssoftware of bij het probleemoplossend werken, hen te stimuleren tot zelfwerkzaamheid, hen permanent te evalueren en een goede veiligheidssituatie te garanderen moet de school absoluut streven naar een situatie waarbij niet meer dan 16 pc's gelijktijdig in gebruik zijn.

Fig. 32 doelstelling leerplan 2011/045 (VVKSO, 2011)

Prijs

Ik vond het ook belangrijk om de prijs te bekijken per applicatie. Er wordt meer en meer nadruk gelegd op de kostprijs van het onderwijs, zo wil men nu ook de maximumfactuur opleggen in het secundair onderwijs. Ik denk dat dit zowel voor de school als voor de leerling (ouders) een belangrijk aspect is.
Want alle kleine beetjes helpen, toch?

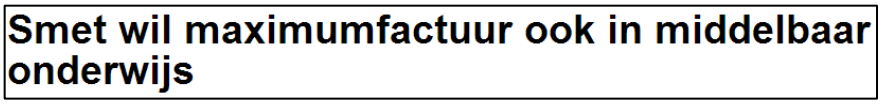


Fig. 33 maximumfactuur (HLN, 3 september 2013)

Oefeningen

Oefening 1:

Deze oefening is een eenvoudige oefening om de ontwikkelomgeving te verkennen. Het is de bedoeling dat deze oefening in de eerste lessen over algoritmisch denken zou kunnen gebruikt worden. Er wordt onder andere gebruik gemaakt van een variabele en de controlestructuur sequentie.

Oefening 2:

Deze oefening is een complexere oefening en bevat meerdere controlestructuren. Het is de bedoeling dat deze oefening na een aantal lessen algoritmisch denken gebruikt zou kunnen worden.

1.3.6 Evaluatiefiche

Sterren

Bij elk evaluatiecriteria worden een aantal beschrijvingen gekoppeld. In deze beschrijvingen wordt beschreven in welke mate de applicatie moet voldoen om 1, 2, 3 of 4 sterren te krijgen.

Opmerking

Tijdens het uittesten van de applicaties ondervond ik nog een aantal dingen. Opmerkelijke voordelen en nadelen plaatste ik bij opmerkingen.

Samenvatting

Ik noteerde bij elke applicatie een besluit. Onder andere waarvoor ik het wel en niet bruikbaar vind, wat de doorslag heeft gegeven, …

Score

Ik plaatste bij elke applicatie een algemene score

--	De applicatie bevat een aantal cruciale knelpunten waardoor de doelen niet kunnen bereikt worden.
-	De applicatie bevat een aantal werkpunten waardoor de doelen nu nog niet kunnen bereikt worden. Indien de applicatie werkt aan deze werkpunten kan hij gebruikt worden om de doelen te bereiken.
+/-	De applicatie is voldoende om de doelen te bereiken.
+	De applicatie is kwalitatief goed om de doelen te bereiken.
++	De applicatie is kwalitatief zeer goed om de doelen te bereiken.

1.3.7 Evaluatie

Ik heb de volgende applicaties geëvalueerd:

- Hopscotch

- Daisy the dinosaur

- Kodable

- Cargo-Bot

- Codea

<u>Verantwoording keuze applicaties</u>
Mijn mentor (waar ik mijn lessenpakket heb uitgetest) stelde voor om deze applicaties uit te testen.

Applicaties worden regelmatig geüpdatet, daarom wil ik hierbij vermelden dat de evaluatie van de applicaties is gemaakt op 12 oktober 2013.

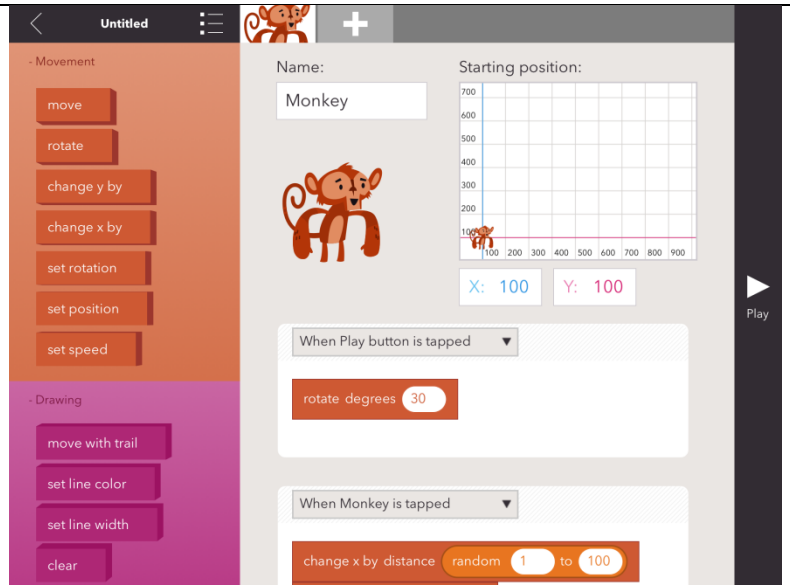
Evaluatiefiche

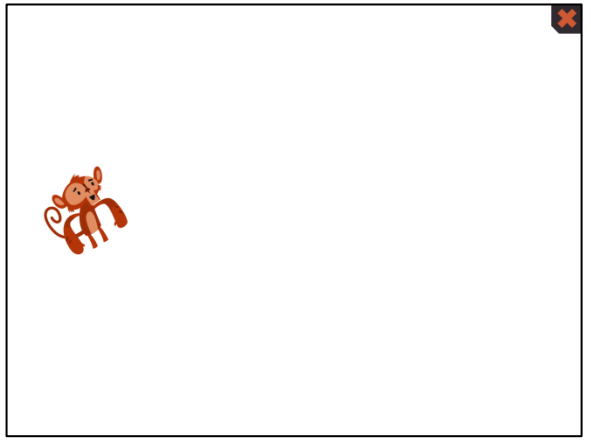
Score: -

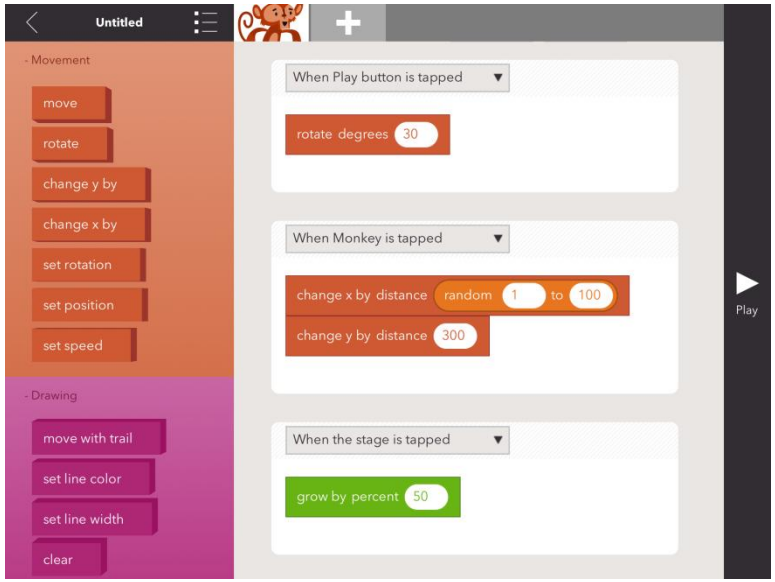
App: Hopscotch

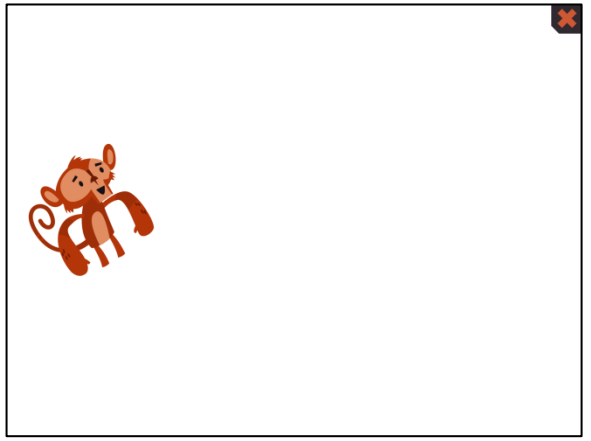
	★	★★	★★★	★★★★
Constructief	De app is beperkt tot het gebruik van hogere orde vaardigheden waaronder evalueren, analyseren en toepassen.	De app maakt meestal gebruik van hogere orde vaardigheden waaronder evalueren, analyseren en toepassen.	De app maakt gebruik van hogere orde vaardigheden waaronder evalueren, analyseren en toepassen.	De app stimuleert het gebruik van hogere orde vaardigheden waaronder evalueren, analyseren en toepassen.
Cumulatief	De app biedt geen flexibiliteit om te voldoen aan de behoeften van de leerlingen.	De app biedt beperkte flexibiliteit om de inhoud en instellingen aan te passen om zo te voldoen aan de behoeften van de leerlingen.	De app biedt enige flexibiliteit om de inhoud en instellingen aan te passen om zo te voldoen aan de behoeften van de leerlingen.	De app biedt volledige flexibiliteit om de inhoud en instellingen aan te passen om zo te voldoen aan de behoeften van de leerlingen.
Zelfgestuurd	Er wordt geen feedback aan de leerlingen gegeven.	Er wordt beperkte feedback aan de leerlingen gegeven.	Er wordt feedback aan de leerlingen gegeven.	Er wordt specifieke feedback aan de leerlingen gegeven.
Doelgericht	De app heeft geen verbinding met het doel en is niet geschikt voor leerlingen.	De app heeft een beperkte verbinding met het doel en is niet geschikt voor leerlingen.	De app is gerelateerd aan het doel en is vooral geschikt voor leerlingen.	De app heeft een sterke verbinding met het doel en is geschikt voor leerlingen.
Gebruiksvriendelijkheid	De app is moeilijk te gebruiken door leerlingen.	De leerkracht moet de leerlingen steeds helpen om de app te gebruiken.	De leerlingen hebben instructie van de leerkracht nodig om de app te gebruiken.	De leerlingen kunnen de app zelfstandig gebruiken.
Delen	Er is geen samenvatting van de prestaties per leerling. Het 'product' van de leerling is niet opgeslagen.	Er is een beperkte samenvatting van de prestaties per leerling. Het 'product' van de leerling is niet opgeslagen.	Er is een samenvatting van de prestaties per leerling. Het 'product' van de leerling is opgeslagen, maar het verdelen is beperkt.	Er is een samenvatting van de prestaties per leerling. Het 'product' van de leerling kan worden opgeslagen en verdeeld naar de leerkracht of naar de andere leerlingen.
Taal	Engels			
Compatibiliteit	Enkel in de AppStore (Apple) te verkrijgen.			
Helpfunctie	Geïntegreerd in de applicatie (Engelstalig). Uitleg van de objecten waarmee de leerlingen kunnen programmeren (handleiding).			
Prijs	Gratis			

Oefening 1

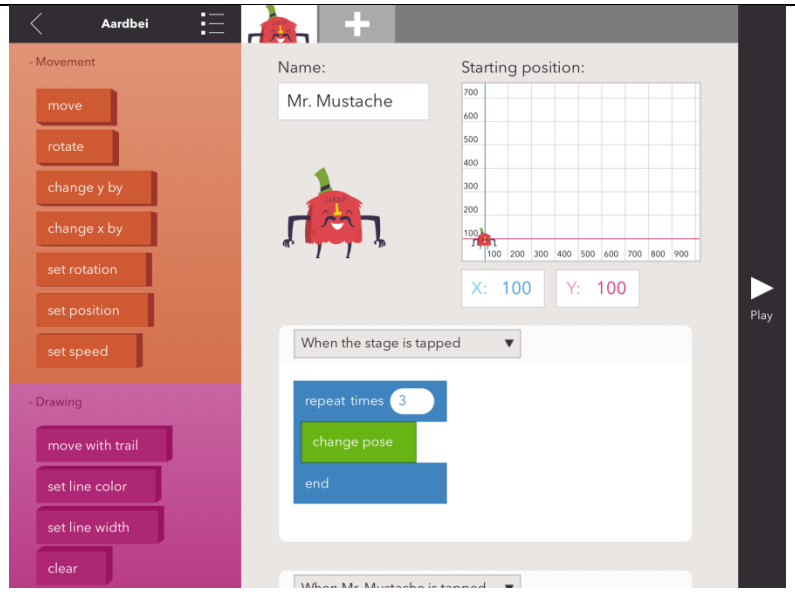

Untitled
Movement
move
rotate
change y by
change x by
set rotation
set position
set speed
Drawing
move with trail
set line color
set line width
clear
Name:
Monkey
Starting position:
X: 100
Y: 100
When Play button is tapped
rotate degrees 30
When Monkey is tapped
change x by distance random 1 to 100
Play

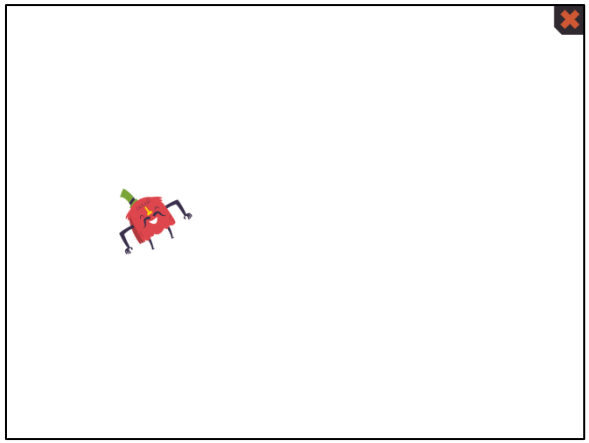


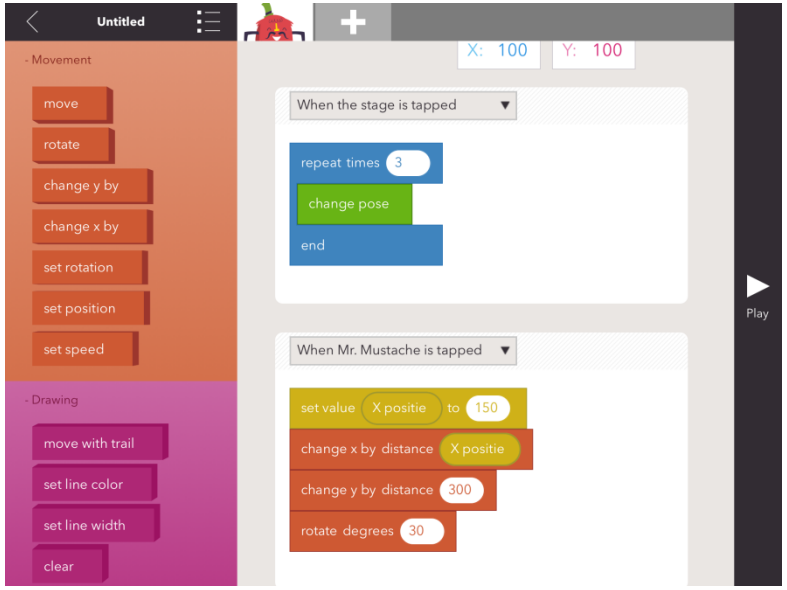

Untitled
Movement
move
rotate
change y by
change x by
set rotation
set position
set speed
Drawing
move with trail
set line color
set line width
clear
When Play button is tapped
rotate degrees 30
When Monkey is tapped
change x by distance random 1 to 100
change y by distance 300
When the stage is tapped
grow by percent 50
Play

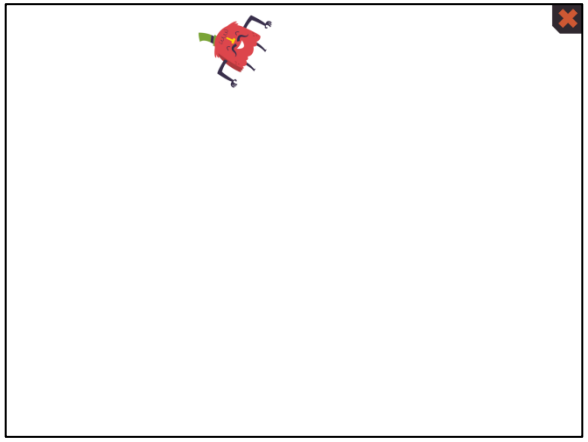


Oefening 2


Aardbei
- Movement
move
rotate
change y by
change x by
set rotation
set position
set speed
- Drawing
move with trail
set line color
set line width
clear
Name:
Mr. Mustache
Starting position:
X: 100
Y: 100
Play
When the stage is tapped
repeat times 3
change pose
end

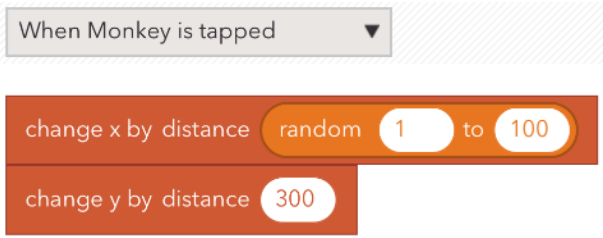



Untitled
- Movement
move
rotate
change y by
change x by
set rotation
set position
set speed
- Drawing
move with trail
set line color
set line width
clear
X: 100
Y: 100
Play
When the stage is tapped
repeat times 3
change pose
end
When Mr. Mustache is tapped
set value X positie to 150
change x by distance X positie
change y by distance 300
rotate degrees 30



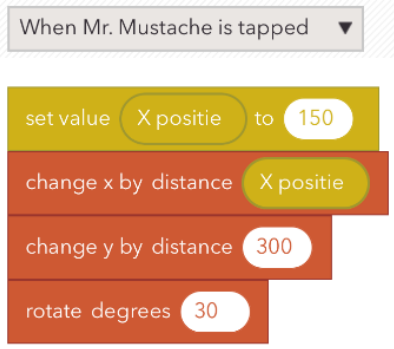
Opmerkingen

Oefening 1
Er kan niet gewerkt worden zonder iteraties te gebruiken. De eenzijdige selectie en de iteratie worden gecombineerd en automatisch gebruikt (zonder dat je het ziet in de programmastructuur/programmacode).
Voorbeeld:



Telkens wanneer de aap (Monkey wordt aangeraakt, zal de aap zich verplaatsen. De x-waarde van de aap zal telkens verhogen met een waarde tussen 1 en 100. De y-waarde zal telkens verhogen met 300.
Indien de aap de randen van het scherm bereikt, blijft deze maximumwaarde behouden.

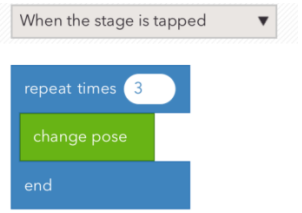
Er kunnen variabelen worden gedeclareerd via "value".
Voorbeeld:



Oefening 2

De controlestructuren zijn zeer beperkt.

- Je kan gebruik maken van de iteratie.
- De enkelvoudige selectie kan gebruikt worden maar de tweevoudige selectie niet.
- De sequentie kan gebruikt worden.



Andere opmerkingen

- De applicatie is vrij kinderlijk.
- De figuren die kunnen gebruikt worden ogen nogal kinderlijk. Je kan ook geen andere figuren importeren.
- De applicatie is makkelijk te begrijpen.
- De leerlingen zullen er snel mee weg zijn, de programmeerlijnen moeten gesleept worden. Er kunnen dus geen typfouten gebeuren in de programmacode enkel denkfouten.
- Een nadeel is wel dat de handleiding in het Engels is.
- De controlestructuren zijn zeer beperkt.
  De controlestructuur sequentie kan niet echt worden aangeleerd, er wordt meteen gewerkt met een combinatie van een eenzijdige selectie met een iteratie.
- Hopscotch is vergelijkbaar met Scratch.
- Deze applicatie is vrij vergelijkbaar met de applicatie Scratch. Deze applicatie is wel makkelijker en er zijn minder mogelijkheden dan in Scratch.
- Het heeft een handige deel- en feedbackfunctie.
- Je kan elk project een eigen naam geven en die dan delen via een url die verkregen wordt. Zowel de leerkracht als de leerlingen kunnen eigenlijk het project bekijken.

Samenvatting

Deze applicatie is in principe bruikbaar in het lager onderwijs, maar aangezien het een Engelstalige applicatie is niet.
Het is vrij gebruiksvriendelijk en heeft een zeer goede deelfunctie. Op het gebied van 'leren in een krachtige leeromgeving' scoort deze applicatie minder: er is geen feedbackfunctie, beperkte controlestructuren, je kan geen programmacode zelf intypen, …
Deze applicatie is niet bruikbaar in het secundair onderwijs. De doelen kunnen niet bereikt worden omdat er te weinig kan ingegaan worden op de controlestructuren vanwege de beperkte functionaliteit.
Indien de applicatie zou aangepast worden zodat alle controlestructuren kunnen gebruikt worden, dan is deze applicatie bruikbaar in het secundair onderwijs.

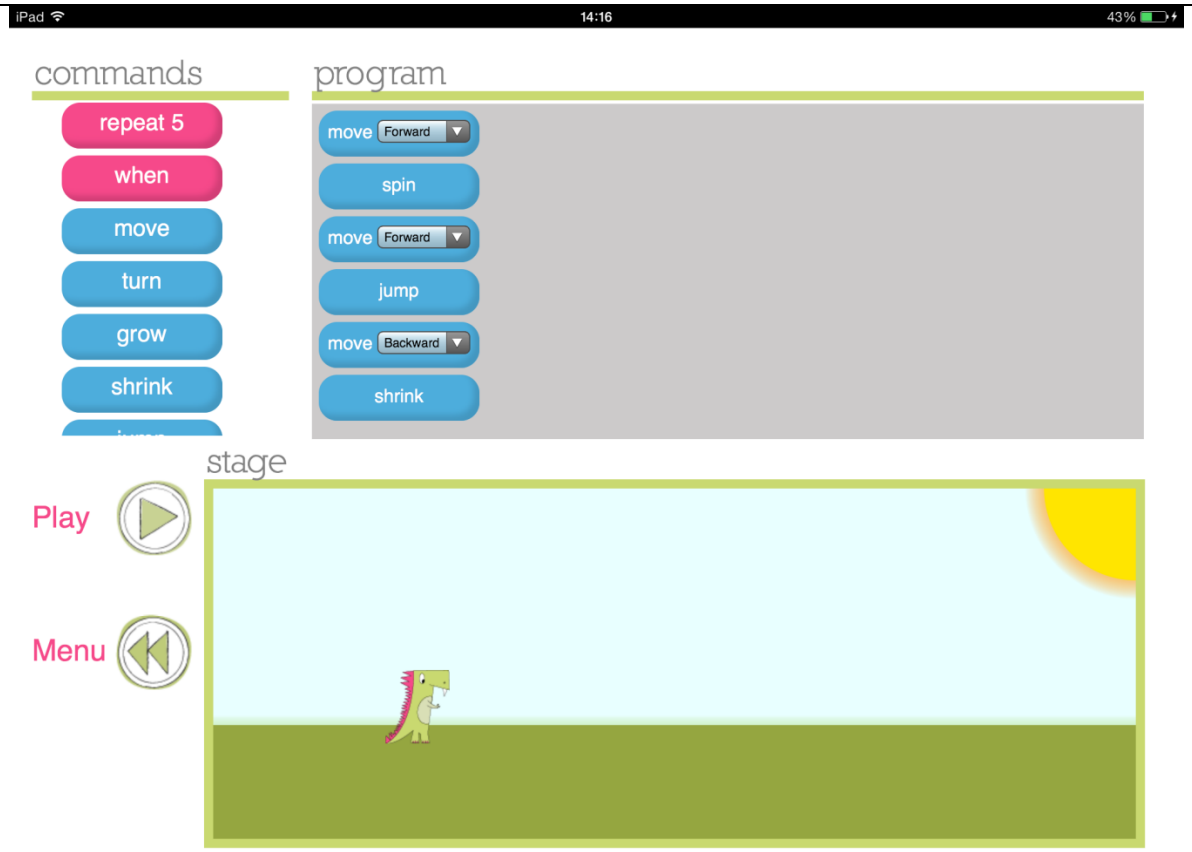
Evaluatiefiche

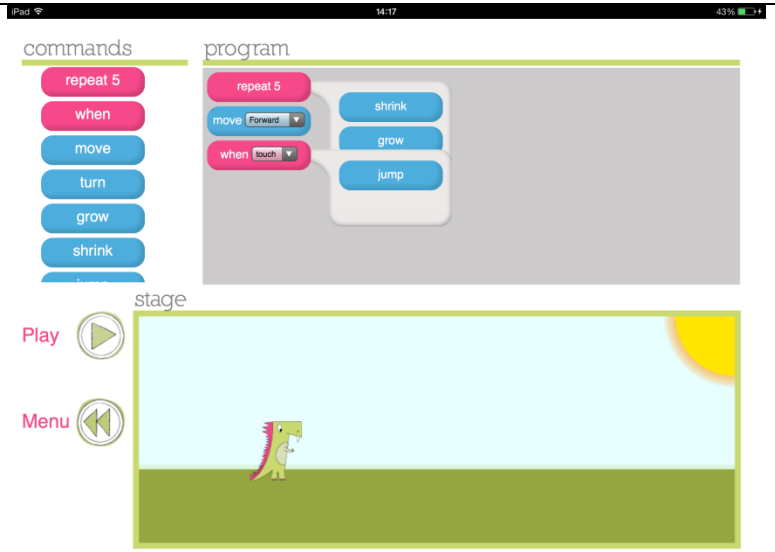
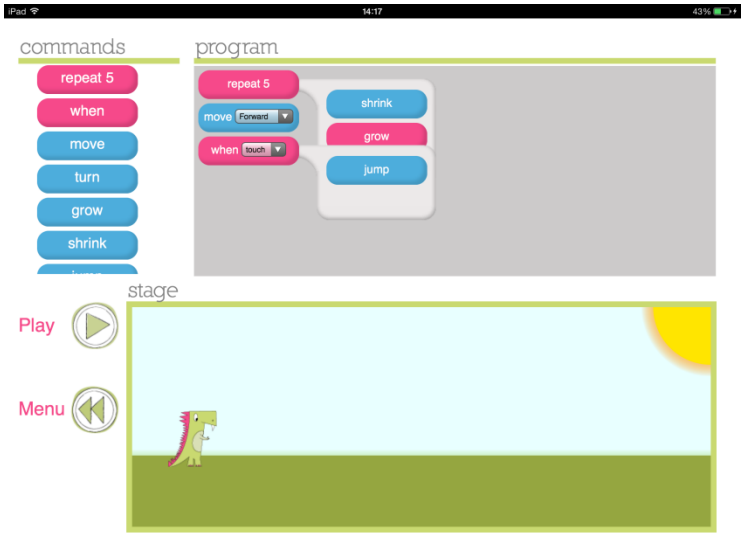
Score: --

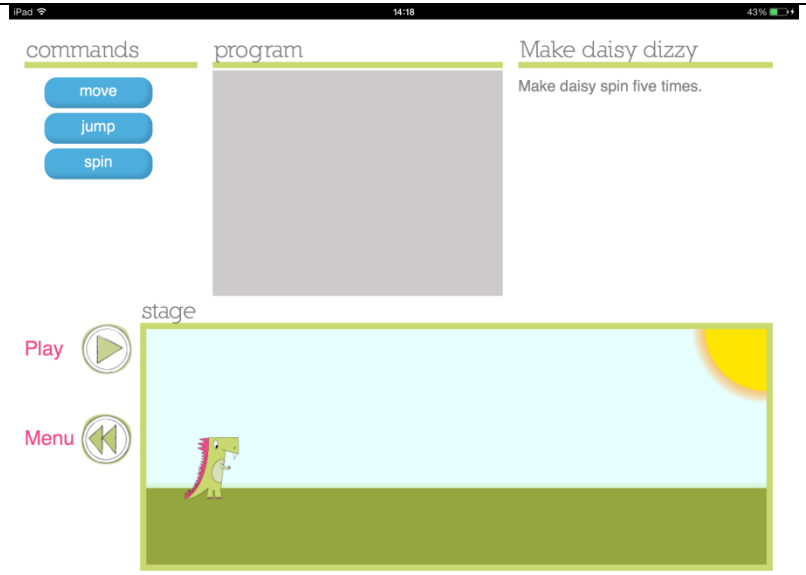
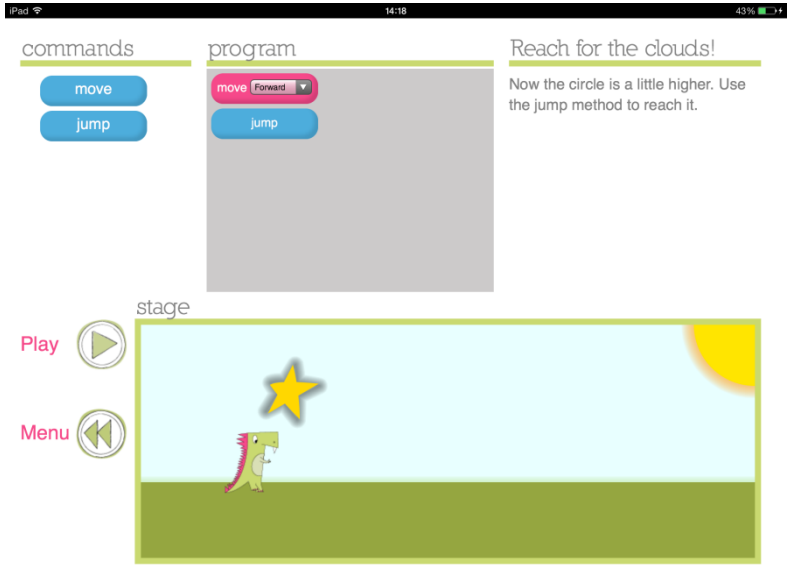
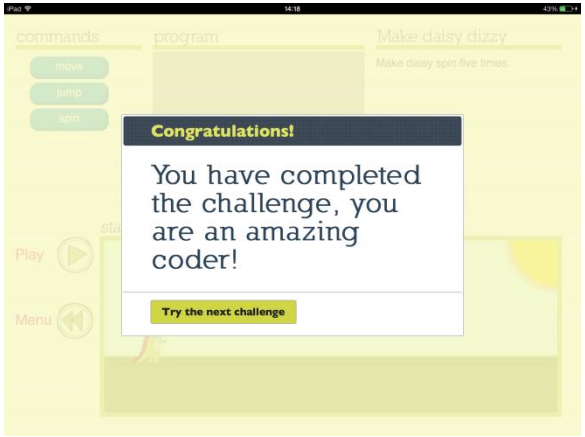
App: Daisy the dinosaur

	★	★★	★★★	★★★★
Constructief	De app is beperkt tot het gebruik van hogere orde vaardigheden waaronder evalueren, analyseren en toepassen.	De app maakt meestal gebruik van hogere orde vaardigheden waaronder evalueren, analyseren en toepassen.	De app maakt gebruik van hogere orde vaardigheden waaronder evalueren, analyseren en toepassen.	De app stimuleert het gebruik van hogere orde vaardigheden waaronder evalueren, analyseren en toepassen.
Cumulatief	De app biedt geen flexibiliteit om te voldoen aan de behoeften van de leerlingen.	De app biedt beperkte flexibiliteit om de inhoud en instellingen aan te passen om zo te voldoen aan de behoeften van de leerlingen.	De app biedt enige flexibiliteit om de inhoud en instellingen aan te passen om zo te voldoen aan de behoeften van de leerlingen.	De app biedt volledige flexibiliteit om de inhoud en instellingen aan te passen om zo te voldoen aan de behoeften van de leerlingen.
Zelfgestuurd	Er wordt geen feedback aan de leerlingen gegeven.	Er wordt beperkte feedback aan de leerlingen gegeven.	Er wordt feedback aan de leerlingen gegeven.	Er wordt specifieke feedback aan de leerlingen gegeven.
Doelgericht	De app heeft geen verbinding met het doel en is niet geschikt voor leerlingen.	De app heeft een beperkte verbinding met het doel en is niet geschikt voor leerlingen.	De app is gerelateerd aan het doel en is vooral geschikt voor leerlingen.	De app heeft een sterke verbinding met het doel en is geschikt voor leerlingen.
Gebruiksvriendelijkheid	De app is moeilijk te gebruiken door leerlingen.	De leerkracht moet de leerlingen steeds helpen om de app te gebruiken.	De leerlingen hebben instructie van de leerkracht nodig om de app te gebruiken.	De leerlingen kunnen de app zelfstandig gebruiken.
Delen	Er is geen samenvatting van de prestaties per leerling. Het 'product' van de leerling is niet opgeslagen.	Er is een beperkte samenvatting van de prestaties per leerling. Het 'product' van de leerling is niet opgeslagen.	Er is een samenvatting van de prestaties per leerling. Het 'product' van de leerling is opgeslagen, maar het verdelen is beperkt.	Er is een samenvatting van de prestaties per leerling. Het 'product' van de leerling kan worden opgeslagen en verdeeld naar de leerkracht of naar de andere leerlingen.
Taal	Engelstalig			
Compatibiliteit	Geen. Projecten kunnen niet worden opgeslagen en gedeeld.			
Helpfunctie	Niet aanwezig.			
Prijs	Gratis			

Oefening 1


iPad
14:16
43%
commands
repeat 5
when
move
turn
grow
shrink
program
move Forward
spin
move Forward
jump
move Backward
shrink
stage
Play
Menu

Oefening 2	 

Challenge mode	  

Opmerkingen

Oefening 1

Er kan niet gewerkt worden met variabelen.
Tijdens het uitvoeren van de programmacode toont het programma, met behulp van kleur, welke programmalijn wordt uitgevoerd. Zo kan mooi de sequentie worden aangetoond.

Oefening 2

Er kan gewerkt worden met de iteratie maar deze is vrij beperkt.
Het blokje 'repeat 5' vertegenwoordigd de iteratie. Hoeveel keer de lus herhaald moet worden kan niet worden ingesteld.

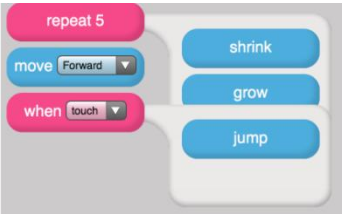


Fig. 34 voorbeeldprogramma Hopscotch (Ann-Sophie Fevery, 2014)

Hier kan ook enkel de enkelvoudige selectie worden gebruikt.

Andere opmerkingen

- Er zijn 2 modes in de applicatie: free-play mode (om zelf programmacode te schrijven) en de challenge mode (om uitdagingen aan te gaan).
- De uitdagingen worden steeds moeilijker.
- Er wordt niet uitgelegd wat er fout is, indien de leerlingen een fout maken.
- De gemaakte uitdagingen worden niet bijgehouden. De leerlingen moeten steeds opnieuw beginnen bij het opnieuw opstarten van de applicatie.
- De leerlingen leren de functionaliteiten, op zelfstandige basis, van het pakket.
- De applicatie is vrij kinderlijk.
- Het figuurtje die gebruikt wordt oogt nogal kinderlijk. Je kan ook geen andere figuren importeren.
- De projecten kunnen niet worden opgeslagen (free-play mode)
- Er kan niet echt gewerkt worden op verschillende oefenniveaus. De applicatie heeft hiervoor te weinig functionaliteiten.

Samenvatting

Deze applicatie is in principe bruikbaar in het lager onderwijs, maar aangezien het een Engelstalige applicatie is niet.
Het is vrij gebruiksvriendelijk en heeft een zeer goede deelfunctie. Op het gebied van 'leren in een krachtige leeromgeving' scoort deze applicatie minder: er is geen feedbackfunctie, beperkte controlestructuren, je kan geen programmacode zelf intypen, …
Deze applicatie is niet bruikbaar in het secundair onderwijs. De applicatie heeft een aantal cruciale knelpunten: niet alle controlestructuren kunnen gebruikt worden, geen opslag, geen deelfunctionaliteit, geen gerichte feedback, …
Er is nog te veel werk aan deze applicatie voordat deze kan ingezet worden in een les.

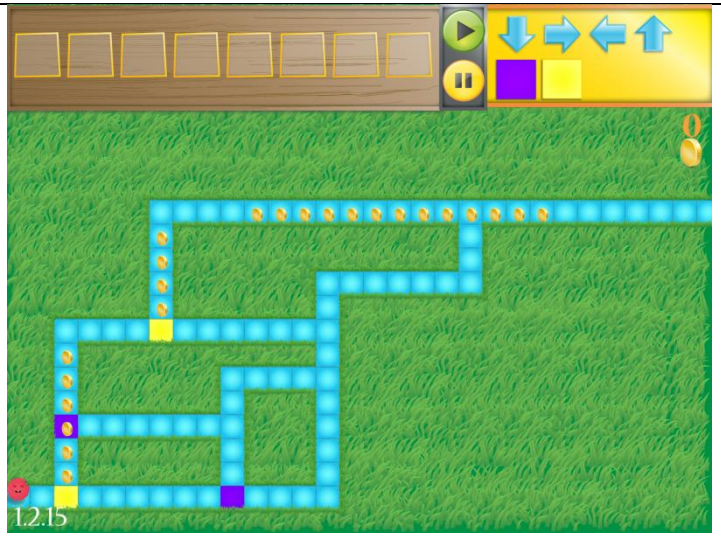
Evaluatiefiche

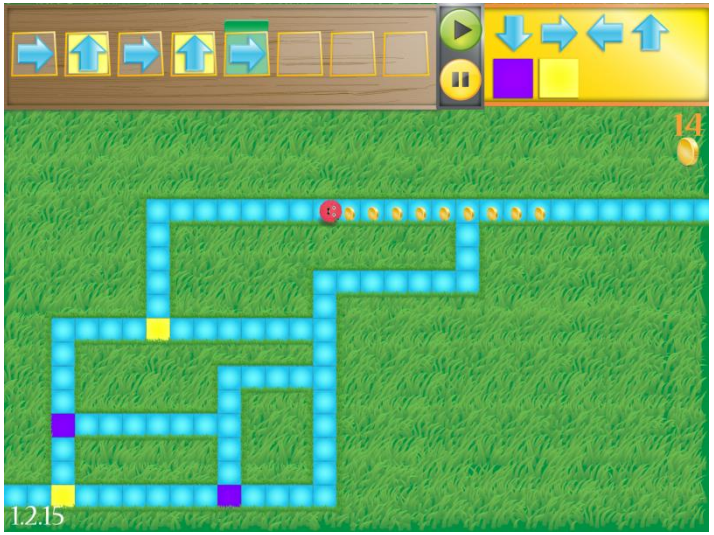
Score: +/-

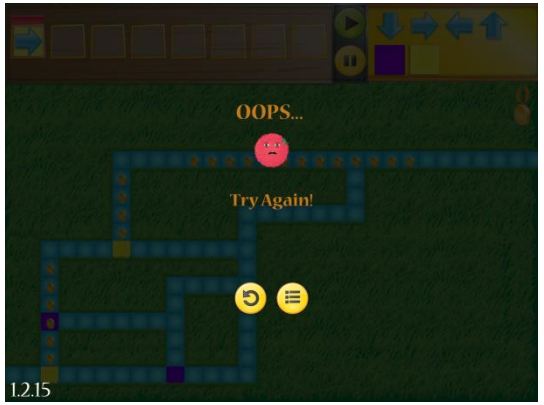
App: Kodable

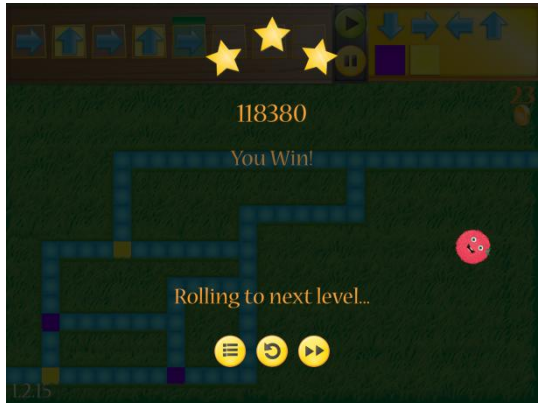
	★	★★	★★★	★★★★
Constructief	De app is beperkt tot het gebruik van hogere orde vaardigheden waaronder evalueren, analyseren en toepassen.	De app maakt meestal gebruik van hogere orde vaardigheden waaronder evalueren, analyseren en toepassen.	De app maakt gebruik van hogere orde vaardigheden waaronder evalueren, analyseren en toepassen.	De app stimuleert het gebruik van hogere orde vaardigheden waaronder evalueren, analyseren en toepassen.
Cumulatief	De app biedt geen flexibiliteit om te voldoen aan de behoeften van de leerlingen.	De app biedt beperkte flexibiliteit om de inhoud en instellingen aan te passen om zo te voldoen aan de behoeften van de leerlingen.	De app biedt enige flexibiliteit om de inhoud en instellingen aan te passen om zo te voldoen aan de behoeften van de leerlingen.	De app biedt volledige flexibiliteit om de inhoud en instellingen aan te passen om zo te voldoen aan de behoeften van de leerlingen.
Zelfgestuurd	Er wordt geen feedback aan de leerlingen gegeven.	Er wordt beperkte feedback aan de leerlingen gegeven.	Er wordt feedback aan de leerlingen gegeven.	Er wordt specifieke feedback aan de leerlingen gegeven.
Doelgericht	De app heeft geen verbinding met het doel en is niet geschikt voor leerlingen.	De app heeft een beperkte verbinding met het doel en is niet geschikt voor leerlingen.	De app is gerelateerd aan het doel en is vooral geschikt voor leerlingen.	De app heeft een sterke verbinding met het doel en is geschikt voor leerlingen.
Gebruiksvriendelijkheid	De app is moeilijk te gebruiken door leerlingen.	De leerkracht moet de leerlingen steeds helpen om de app te gebruiken.	De leerlingen hebben instructie van de leerkracht nodig om de app te gebruiken.	De leerlingen kunnen de app zelfstandig gebruiken.
Delen	Er is geen samenvatting van de prestaties per leerling. Het 'product' van de leerling is niet opgeslagen.	Er is een beperkte samenvatting van de prestaties per leerling. Het 'product' van de leerling is niet opgeslagen.	Er is een samenvatting van de prestaties per leerling. Het 'product' van de leerling is opgeslagen, maar het verdelen is beperkt.	Er is een samenvatting van de prestaties per leerling. Het 'product' van de leerling kan worden opgeslagen en verdeeld naar de leerkracht of naar de andere leerlingen.
Taal	Engelstalig			
Compatibiliteit	Geen. Projecten kunnen niet worden gedeeld. Het niveau van de leerling wordt bijgehouden.			
Helpfunctie	Zeer uitgebreid aanwezig (engelstalig).			
Prijs	Gratis. Voor extra functionaliteiten moet je betalen.			

Oefening


0
1.2.15


14
1.2.15


OOPS...
Try Again!
1.2.15


118380
You Win!
Rolling to next level...


1

Opmerkingen

Oefening

- Ik kon geen twee oefeningen uitwerken zoals bij de andere applicaties aangezien je hier niet zelf kan programmeren.
- De leerstof word aangeboden door middel van een spel. Je moet telkens een level voltooien om naar het volgende te kunnen gaan.
- De controlestructuren komen allen aan bod maar worden niet benoemd. De verschillende cognitieve niveaus worden dus niet allemaal bereikt.
- Er wordt niet gewerkt met variabelen.

Andere opmerkingen

- 3 spelmodussen;

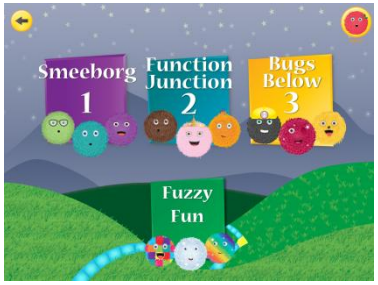


Fig. 35 keuze spelmodus (Ann-Sophie Fevery, 2014)

- ouder-, leerkracht- en kindermodus;

Fig. 36 keuze spelersmodus (Ann-Sophie Fevery, 2014)

- oplossingsmodel;
- evolutie;
- evolutie van de leerlingen kan de leerkracht volgen;
- spelvorm;
- moeilijkheidsgraden;
- handleiding.

Fig. 37 voorbeeldpagina handleiding (Ann-Sophie Fevery, 2014)

Samenvatting

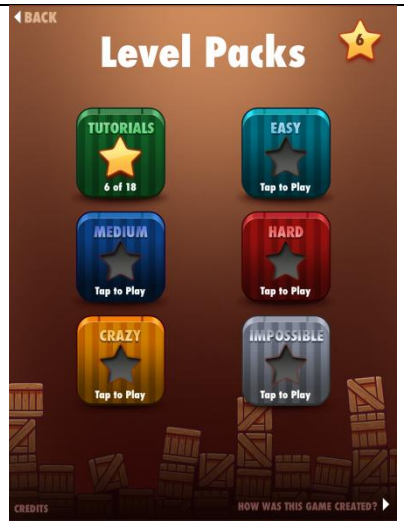
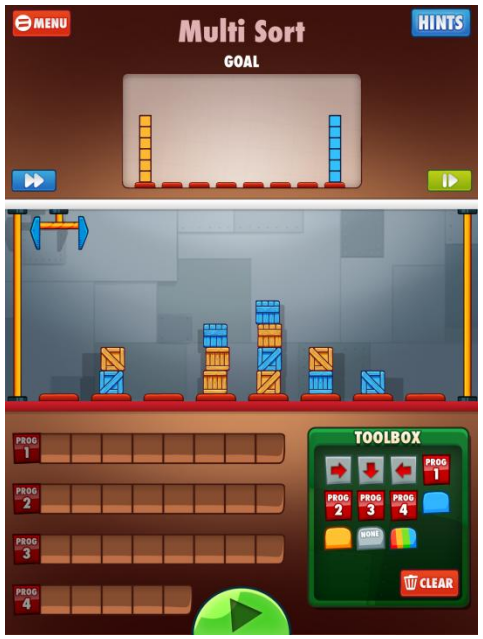
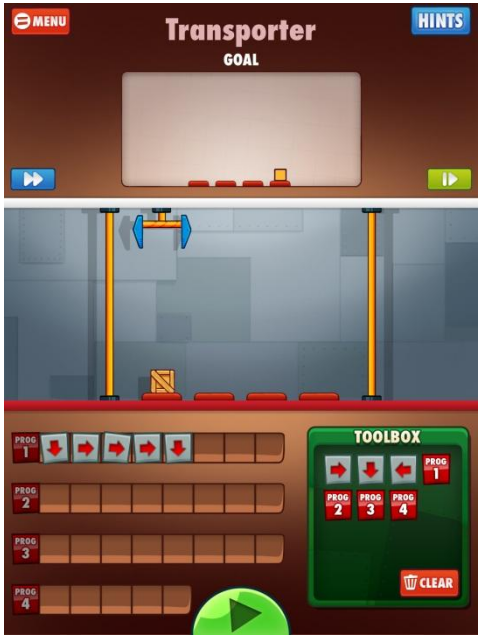
Deze applicatie kan gebruikt worden om een introductieles mee te beginnen of om de leerstof te herhalen bij een syntheseles.
Alle leerstof wordt aangehaald maar telkens op hetzelfde niveau dus bepaalde doelen zullen niet bereikt worden.
Kortom deze applicatie is bruikbaar in het secundair onderwijs, maar is niet bruikbaar om een hele lessenreeks rond te bouwen.

Evaluatiefiche

Score: +

App: Cargo-Bot

	★	★★	★★★	★★★★
Constructief	De app is beperkt tot het gebruik van hogere orde vaardigheden waaronder evalueren, analyseren en toepassen.	De app maakt meestal gebruik van hogere orde vaardigheden waaronder evalueren, analyseren en toepassen.	De app maakt gebruik van hogere orde vaardigheden waaronder evalueren, analyseren en toepassen.	De app stimuleert het gebruik van hogere orde vaardigheden waaronder evalueren, analyseren en toepassen.
Cumulatief	De app biedt geen flexibiliteit om te voldoen aan de behoeften van de leerlingen.	De app biedt beperkte flexibiliteit om de inhoud en instellingen aan te passen om zo te voldoen aan de behoeften van de leerlingen.	De app biedt enige flexibiliteit om de inhoud en instellingen aan te passen om zo te voldoen aan de behoeften van de leerlingen.	De app biedt volledige flexibiliteit om de inhoud en instellingen aan te passen om zo te voldoen aan de behoeften van de leerlingen.
Zelfgestuurd	Er wordt geen feedback aan de leerlingen gegeven.	Er wordt beperkte feedback aan de leerlingen gegeven.	Er wordt feedback aan de leerlingen gegeven.	Er wordt specifieke feedback aan de leerlingen gegeven.
Doelgericht	De app heeft geen verbinding met het doel en is niet geschikt voor leerlingen.	De app heeft een beperkte verbinding met het doel en is niet geschikt voor leerlingen.	De app is gerelateerd aan het doel en is vooral geschikt voor leerlingen.	De app heeft een sterke verbinding met het doel en is geschikt voor leerlingen.
Gebruiksvriendelijkheid	De app is moeilijk te gebruiken door leerlingen.	De leerkracht moet de leerlingen steeds helpen om de app te gebruiken.	De leerlingen hebben instructie van de leerkracht nodig om de app te gebruiken.	De leerlingen kunnen de app zelfstandig gebruiken.
Delen	Er is geen samenvatting van de prestaties per leerling. Het 'product' van de leerling is niet opgeslagen.	Er is een beperkte samenvatting van de prestaties per leerling. Het 'product' van de leerling is niet opgeslagen.	Er is een samenvatting van de prestaties per leerling. Het 'product' van de leerling is opgeslagen, maar het verdelen is beperkt.	Er is een samenvatting van de prestaties per leerling. Het 'product' van de leerling kan worden opgeslagen en verdeeld naar de leerkracht of naar de andere leerlingen.
Taal	Engelstalig			
Compatibiliteit	Geen. Projecten kunnen niet worden gedeeld. Het niveau van de leerling wordt bijgehouden.			
Helpfunctie	Niet aanwezig.			
Prijs	Gratis			

Oefening	  

Opmerkingen

Oefening

- Ik kon geen twee oefeningen uitwerken zoals bij de andere applicaties aangezien je hier niet zelf kan programmeren.
- De leerstof word aangeboden door middel van een spel. Het goede aan dit spel is dat je niet telkens een level moet voltooien om naar het volgende te kunnen gaan. Je kan kiezen op welk niveau je start.
- De controlestructuren komen allen aan bod maar worden niet benoemd. De verschillende cognitieve niveaus worden dus niet allemaal bereikt.
- Aan het programmeren komt geen code te pas. Je kiest commando's en koppelt ze aan elkaar.

Andere opmerkingen

- Je kan deze applicatie niet gebruiken in landscape-modus.
- Deze applicatie heeft verschillende moeilijkheidsgraden.
- Cargo-bot biedt geen deelfunctionaliteiten.
- Er wordt geen concrete feedback gegeven aan de leerlingen waar de fouten precies aanwezig zijn. Dit is vrij moeilijk voor hen.

Fig. 38 foutanalyse (Cargo-Bot, 2014)

- Alle controlestrucuren zijn aanwezig.
- De spelvorm is vrij ééntonig. Je krijgt steeds dezelfde opgave voorgeschoteld (= dozen verzetten).
- Men begint tamelijk snel met recursie (binnen een procedure het zichzelf aanroepen).
- Veel YouTube filmpjes te vinden met de oplossing.

Samenvatting

Deze applicatie is in principe bruikbaar in het secundair onderwijs. Aangezien je niet alle competenties kan bereiken, zou ik het eerder gebruiken om een introductieles programmeren mee te starten.

Deze applicatie is niet bruikbaar in het lager onderwijs vanwege de moeilijkheidsgraad.

Evaluatiefiche

Score: ++

App: Codea

	★	★★	★★★	★★★★
Constructief	De app is beperkt tot het gebruik van hogere orde vaardigheden waaronder evalueren, analyseren en toepassen.	De app maakt meestal gebruik van hogere orde vaardigheden waaronder evalueren, analyseren en toepassen.	De app maakt gebruik van hogere orde vaardigheden waaronder evalueren, analyseren en toepassen.	De app stimuleert het gebruik van hogere orde vaardigheden waaronder evalueren, analyseren en toepassen.
Cumulatief	De app biedt geen flexibiliteit om te voldoen aan de behoeften van de leerlingen.	De app biedt beperkte flexibiliteit om de inhoud en instellingen aan te passen om zo te voldoen aan de behoeften van de leerlingen.	De app biedt enige flexibiliteit om de inhoud en instellingen aan te passen om zo te voldoen aan de behoeften van de leerlingen.	De app biedt volledige flexibiliteit om de inhoud en instellingen aan te passen om zo te voldoen aan de behoeften van de leerlingen.
Zelfgestuurd	Er wordt geen feedback aan de leerlingen gegeven.	Er wordt beperkte feedback aan de leerlingen gegeven.	Er wordt feedback aan de leerlingen gegeven.	Er wordt specifieke feedback aan de leerlingen gegeven.
Doelgericht	De app heeft geen verbinding met het doel en is niet geschikt voor leerlingen.	De app heeft een beperkte verbinding met het doel en is niet geschikt voor leerlingen.	De app is gerelateerd aan het doel en is vooral geschikt voor leerlingen.	De app heeft een sterke verbinding met het doel en is geschikt voor leerlingen.
Gebruiksvriendelijkheid	De app is moeilijk te gebruiken door leerlingen.	De leerkracht moet de leerlingen steeds helpen om de app te gebruiken.	De leerlingen hebben instructie van de leerkracht nodig om de app te gebruiken.	De leerlingen kunnen de app zelfstandig gebruiken.
Delen	Er is geen samenvatting van de prestaties per leerling. Het 'product' van de leerling is niet opgeslagen.	Er is een beperkte samenvatting van de prestaties per leerling. Het 'product' van de leerling is niet opgeslagen.	Er is een samenvatting van de prestaties per leerling. Het 'product' van de leerling is opgeslagen, maar het verdelen is beperkt.	Er is een samenvatting van de prestaties per leerling. Het 'product' van de leerling kan worden opgeslagen en verdeeld naar de leerkracht of naar de andere leerlingen.
Taal	Engelstalig			
Compatibiliteit	De projecten kunnen worden opgeslagen en gedeeld via Xcode. Er is ook een mogelijkheid om het tabletscherm te delen met een computerscherm.			
Helpfunctie	Sterk aanwezig. (Forum, website, in applicatie zelf)			
Prijs	Betalend			

Oefening 1


Main
-- constanten
function setup()
waarde=10
print("startwaarde=",waarde)
end
function draw()
end

Oefening 2

Main

```
-- Piramide

function setup()
end

function draw()
    for n = 1, 10, 1 do
          for m = 1, 11-n, 1 do
             ellipse(100+m*60+(n-1)*30, 60+n*60, 40, 40)
   end
       end
end
```

Opmerkingen

Oefening 1

- Er kan gewerkt worden met variabelen en cosntanten.
- Tijdens het uitvoeren van de programmacode toont het programma niet welke programmalijn wordt uitgevoerd.
- De waarde van variabelen en constanten worden automatisch in een blauwe kleur aangeduid. Dit is een handig hulpmiddel voor leerlingen.

Oefening 2

- Er kan gewerkt worden met de controlestructuur iteratie.
- De code van de controlestructuren is vrij gelijkaardig aan die van andere programmeertalen.
- De controlestructuren, in deze oefening de iteratie, worden automatisch in een andere kleur aangeduid. Op deze manier zien de leerlingen de structuur van de code beter.

Andere opmerkingen

- De applicatie is betalend.
- Het tabletscherm kan gedeeld worden met een computerscherm.
- De projecten kunnen worden opgeslagen en gedeeld met de leerkracht of medeleerlingen. Het neemt wel veel tijd in beslag.
- De leerlingen krijgen gerichte feedback. De code wordt onderlijnd waar er een fout is ingeslopen.
- Je hebt het gevoel van echt programmeren.
- Je kan games ontwikkelen. Het is een programmeeromgeving die aanspreekt bij leerlingen.
- Je kan verschillende oefeningen maken.
- Lua is een moeilijke programmeertaal.
- Het is geen alledaagse programmeertaal.
- Alle controlestructuren kunnen gebruikt worden.

Samenvatting

Deze applicatie is zeker bruikbaar in het secundair onderwijs. Je kan deze applicatie inzetten voor een hele lessenreeks rond algoritmisch denken want alle competenties kunnen bereikt worden.
Je kan werken met verschillende oefenniveaus, werken aan foutanalyse, games laten ontwikkelen, …
Kortom de leerlingen zullen zeker geboeid zijn door deze programmeeromgeving.

Deze applicatie is niet bruikbaar in het lager onderwijs vanwege de moeilijkheidsgraad.

1.3.8 Besluit

Na de verschillende applicaties te evalueren ben ik tot de volgende besluiten gekomen.

De applicaties Hopscotch en Daisy the dinosaur zijn niet echt te gebruiken in het secundair onderwijs vanwege het makkelijke niveau.
Beide applicaties zouden goede initiatieapplicaties zijn voor het lager onderwijs. Het enige grote nadeel om dit te gaan gebruiken in het lager onderwijs is de taal. Beide applicaties zijn Engelstalig. Hopelijk wordt deze drempel overwonnen door de eventuele ingang van de Engelste taal in de lagere school.

De applicaties Kodable, Cargo-Bot en Codea zijn allemaal bruikbaar voor het secundair onderwijs.
Kodable is een goede applicatie maar werkt steeds op hetzelfde niveau waardoor alle leerlingen steeds dezelfde oefeningen moeten maken. De snelle leerlingen zullen in deze applicatie geen uitdaging vinden.
Cargo-Bot is een applicatie waar je niet alle competenties uit het leerplan mee kan bereiken. De beslissing om Cargo-Bot niet te gaan gebruiken is dan ook snel gemaakt.
Daarom gaat mijn voorkeur uit naar Codea. In het besluit bij de evaluatiefiche van Codea, kan je de grootste voor- en nadelen lezen.
Ik heb dan ook mijn lessenpakket uitgewerkt met deze applicatie.
Codea heeft één groot nadeel:
Eén van mijn vereisten was eigenlijk ook om een goede applicatie te vinden die platformonafhankelijk is. Codea kan enkel geïnstalleerd worden via de App Store (Apple-afhankelijk). Ik heb dit principe moeten aan de kant schuiven, want er was niet echt een platformonafhankelijke applicatie te vinden die even goed is als Codea.

1.4 Eerst denken dan doen

1.4.1 Onderzoek

"Eerst denken dan doen" is de belangrijkste didactische wenk/attitude bij algoritmisch denken. In het leerplan staat dit ook vermeld (zie figuur).
Ik heb bij deze onderzoeksvraag een methode/stappenplan uitgewerkt om aan deze attitude te kunnen werken.

Competentie 9 - Algoritmisch denken

Algemene didactische wenken

- "Eerst denken en dan doen" is het belangrijkste bij algoritmisch denken.
- Bij complexere problemen een algoritme schematisch (laten) voorstellen door een Nassi-Shneiderman diagram (NSD). In de andere gevallen volstaat een uitgebreide beschrijving van de verwerkingsfase in de analyse.

Fig. 39 fragment uit het leerplan 2011/039 (VVKSO, 2011)

1.4.2 Stappenplan

In de cursus heb ik gewerkt met een stappenplan, hieronder wat meer uitleg.

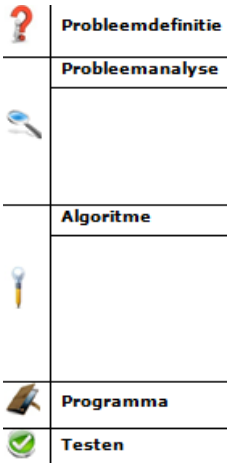
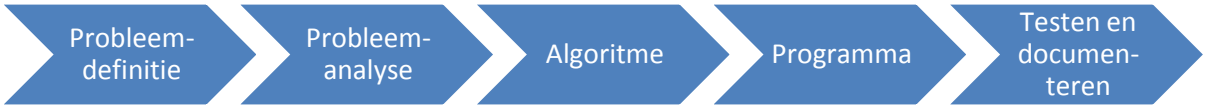


Fig. 40 voorbeeld uit de cursus (Ann-Sophie Fevery, 2014)



- Probleemdefinitie

  In de probleemdefinitie worden de eisen van het probleem en de gewenste resultaten beschreven.
  Dat is eigenlijk de opdracht die de leerlingen voorgeschoteld krijgen.

- Probleemanalyse

  In de probleemanalyse worden de wegen die naar een oplossing leiden bestudeerd en wordt er de beste uitgekozen. Dit betekent dat er onderzocht moet worden welke mogelijkheden er zich voordoen, waar de moeilijkheden schuilen en welke bijkomende gegevens er nodig zijn.

De leerlingen moeten volgend schema steeds invullen:

- Algoritme

  In deze fase wordt de oplossingsmethode stap voor stap uitgetekend. De leerlingen gebruiken hiervoor een Nassi-Schneidermanndiagram (zie verder).

- Programma

  De leerlingen gaan in deze fase aan de slag in Codea. Hierin wordt de werkelijke code geschreven.

- Testen en documenteren

  In deze fase wordt getest of het programma voldoet aan de gestelde eisen (probleemdefinitie). Controleer het programma op syntactische fouten, structuurfouten met behulp van testwaarden.

2 Lessenpakket

2.1 Codea

Codea is de combinatie van Code en Ideas.
Het is een programmeertaal die 60 beelden per seconde hertekent en weergeeft op je scherm, dit voor het gebruik van games en andere grafische doeleinden. Het wordt vooral gebruikt voor de weergave van 2D, in mindere mate voor 3D (beperkt).

Codea start met de schrijftaal Lua waarin verscheidene extra bibliotheken zijn toegevoegd om toegang te krijgen tot de Ipad. Een interne grafische en fysische bibliotheek is reeds ingebouwd waardoor het bijvoorbeeld gemakkelijk is om bepaalde dingen te laten reageren op zwaartekracht en aanraakbewegingen.

2.2 Lua

Lua is een programmeertaal die ontworpen is door Roberto Lerusalimschy, Waldemar Celes en Luiz Henrique de Figueiredo en is verschenen in 1993.

Lua betekent "maan" in het Portugees. Het is dus geen afkorting maar een naamwoord. Vandaar dat Lua begint met een hoofdletter en de volgende letters kleine letters zijn.

Lua is gratis en heeft reeds zijn kunnen bewezen in tal van industriële applicaties en games. Ook is het pakket mobiel bruikbaar (Android, iOS, BREW, Symbian, Windows). Het is makkelijk te importeren in een specifieke applicatie en kan uitgebreid worden met 'libraries' geschreven in andere talen.

Lua heeft ook zijn eigen 'community' waar allerlei vragen kunnen worden gesteld onder de gebruikers.

2.2.1 Sterke punten

Er zijn een aantal redenen die de makers van de taal zelf opgeven als reden om te kiezen voor Lua. Hieronder som ik de belangrijkste redenen op.

- Lua is een stevige taal
  Lua is gebruikt in vele industriële toepassingen (bijvoorbeeld Adobe's Photoshop) en games (bijvoorbeeld Angry Birds). Lua is momenteel de leidende scripttaal in games en er zijn verschillende boeken over geschreven.
- Lua is snel
  Lua heeft een grote reputatie ten opzichte van prestaties.Op verschillende punten blijkt Lua de snelste taal van alle scripttalen te zijn.
- Lua is draagbaar
  Lua draait op iOS, Windows, mobiele apparaten (Android, iOS, Symbian, Windows Phone,…).
- Lua is integreerbaar
  Lua is een snelle taal met een kleine bestandsgrootte zodat je het eenvoudig kunt integreren in een toepassing. Het is gemakkelijk om Lua uit te breiden met bibliotheken die geschreven zijn in andere talen. Het is ook gemakkelijk om programma's uit te breiden met Lua die geschreven zijn in een andere programmeertaal.
- Lua is gratis
  Lua is een gratis open-source programma. Het kan worden gebruikt voor elk doel. Je kan het gewoon downloaden en gebruiken.

2.2.2 Toepassingen

Er zijn verschillende games en programma's die Lua gebruiken:

- SimCity 4
- Angry Birds
- World of Warcraft
- Adobe Photoshop Lightroom

2.3 Cognitieve multimediatheorie van Mayer

2.3.1 Leren

Wat is leren?
Leren is een constructief, cumulatief, zelfgestuurd, doelgericht, contextgebonden, coöperatief en individueel verschillend proces van kennisverwerving, betekenisgeving en vaardigheidsontwikkeling. (De Corte, 1996)

2.3.2 Digitaal leren

Digitaal leren is leren waarbij de leeractiviteiten van de leerlingen worden ondersteund of uitgevoerd met gebruik van digitale leermiddelen. (Baars, Wieland, van de Ven, Jager, 2006)

2.3.3 Wat is multimedia? Wat zegt de theorie van Mayer?

Deze multimediatheorie wordt vooral aangeboden en toegepast in het onderwijs met de bedoeling het menselijk leren te bevorderen. Mayers theorie biedt een inzicht om de invloed te beschrijven van de verschillende leermaterialen die multimedia ons te bieden heeft.

Multimedia is grotendeels afhankelijk van de visie(s) die men hierop heeft. Mayer baseert zich op twee visies voor de verdere uitwerking van de cognitieve multimediatheorie.
De visie van de presentatie en de visie over het zintuiglijke. De kern van de cognitivistische visie is dat de leerlingen en studenten bij hun leerproces leren schema's opbouwen die in hun langetermijngeheugen worden opgeslagen. Die schema's vormen de basis voor het verdere leren om zo makkelijker nieuwe informatie te kunnen situeren, organiseren en integreren in het geheugen.

De eerste visie slaat op het aanbieden of presenteren van leermateriaal via meerdere kanalen. Mayer baseert zich op deze visie maar in aangepaste versie. Volgens Mayer heeft deze eerste visie enkel betrekking op het aanbieden van informatie via 2 kanalen: auditief (voorbeeld een illustratie) en visueel (voorbeeld een geschreven tekst).
Mayer beschrijft deze methode als 'dual-code' of 'dual-channel'.

De tweede visie, het zintuiglijke, slaat niet op de soorten verwerkingskanalen, wel op de zintuigen die hiervoor worden gebruikt om deze informatie te ontvangen. Gebaseerd op deze visie kan men afleiden dat bij multimedia verschillende zintuigen betrokken zijn bij het ontvangen van een bericht of boodschap. Mayer stelt hierbij dat presentaties enkel uit auditieve en visuele materialen bestaan.

Zowel de eerste als tweede visie lijken enorm op elkaar, toch zijn het twee verschillende visies op multimedia. De eerste legt de nadruk op de verwerkingskanalen, de tweede visie op de zintuigen die worden gebruikt om deze informatie te ontvangen.

Het proces dat leidt tot de opbouw van betere leerschema's gebeurt in een drietal fasen: selectieprocessen, organisatieprocessen en integratieprocessen. Tijdens de selectieprocessen selecteert de leerling de informatie uit de verschillende leermaterialen die hij/zij relevant vindt om te onthouden. De organisatieprocessen zorgen dat de nieuwe informatie en/of leerstof wordt omgezet en georganiseerd in mentale modellen. Die wordt dan op zich terug gekoppeld aan de voorkennis. Dit zijn de integratieprocessen.

2.3.4 Onderliggende assumpties

De multimediakenmerken van informatie zijn van groot belang bij de hierboven reeds beschreven processen. Mayer zijn cognitieve multimediatheorie benadrukt dit sterk en is ook deels gebaseerd op zijn eigen veronderstellingen, wat de theorie dus niet helemaal objectief maakt. Mayer zegt bijvoorbeeld dat het menselijk werkgeheugen enkel uit auditieve en visuele kanalen bestaat, met andere woorden we gebruiken slechts twee zintuigen: onze ogen en oren.

Samen met nog andere auteurs schuift Mayer drie belangrijke veronderstellingen/assumpties voorop: de 'dual-channel' assumptie, de 'limited capacity' assumptie en de 'active processing' assumptie.

<u>Dual-channel assumptie</u>
Hiervoor heeft hij zich vooral gebaseerd op de theorie van het werkgeheugen van Baddeley. Hierop verdergaand stelt Mayer dat er tussen het auditieve en visuele kanaal slechts connecties kunnen worden gemaakt wanneer deze gelijktijdig in het werkgeheugen aanwezig zijn.
Hoewel we beschikken over twee kanalen om informatie op te nemen en te verwerken staan die niet los van elkaar. Er zijn wisselwerkingen tussen beide.

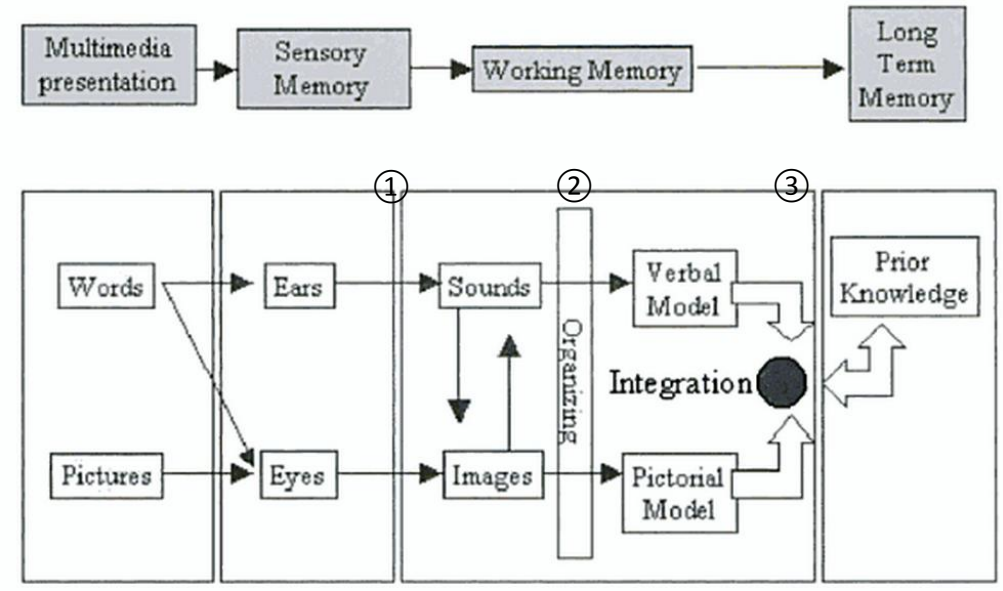


Fig. 41 schema dual channel assumptie (Ivan d'Haese en Martin Valcke, onbekend)

Verduidelijking bij de figuur:

1. het selecteren van relevante beelden voor het verwerken in het visuele werkgeheugen en/of het selecteren van relevante woorden voor het verwerken in het verbale werkgeheugen;
2. het organiseren van de geselecteerde beelden tot een visueel model en/of het organiseren van de geselecteerde woorden tot een verbaal model;
3. het integreren van de verbale- en visuele modellen met de reeds aanwezige voorkennis.

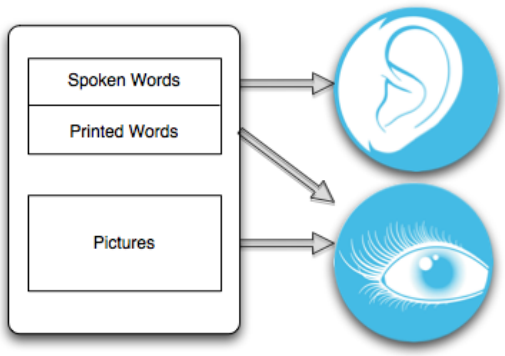


Fig. 42 overzicht dual channel assumptie (Michael Ramey, 2013)

Limited capacity assumptie

Ten tweede zegt de cognitive load theorie van Sweller en de theorie van het werkgeheugen van Baddeley dat het werkgeheugen beperkt is in zijn capaciteit. De mens is dus niet in staat onbeperkt informatie op te slaan of te verwerken. Wanneer leermaterialen worden ontworpen moet daarmee rekening worden gehouden. Enkel visuele informatie tonen betekent dat we de beschikbare capaciteit via het auditieve kanaal negeren en omgekeerd.

Active processing assumptie

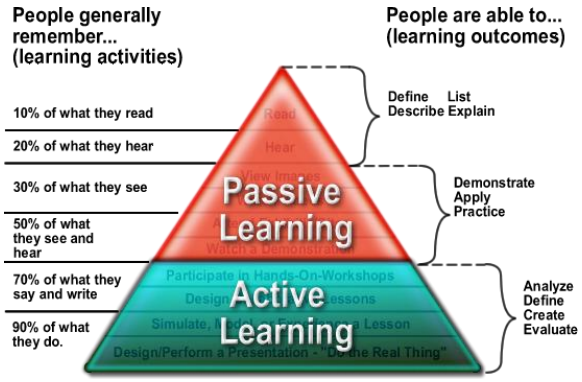


Fig. 43 leeractiviteiten (Study guides and strategies, 2001)

Ten derde beschouwt Mayer de mens als een soort van processor die alle kennis en informatie opslaat en verwerkt, die betekenisvol leert wanneer hij het SOI-model toepast (zie verder). Om actief te leren zegt Mayer dat verscheidene cognitieve processen gecoördineerd uitgevoerd moeten worden tijdens het leren.

Voorbeeld:

Structuurtype	Beschrijving
Proces	Oorzaak - Gevolg
Generalisatie	Idee en de samenhangende concepten beschrijven
Classificatie	Domein in sets en subsets analyseren

Fig. 44 structuurtypes (Ivan d'Haese en Martin Valcke, onbekend)

SOI model:

De theorie van Mayer is deels gebaseerd op het SOI-model en zijn 5 stappen.

- **S**electeren van relevante woorden
- **S**electeren van relevante figuren
- **O**rganiseren van relevante woorden
- **O**rganiseren van relevante figuren
- **I**ntegreren van woord- en beeld gerelateerde representaties

Deze drie assumpties (de 'dual-channel' assumptie, de 'limited capacity' assumptie en de 'active processing' assumptie) zijn de hoofdpijlers van de cognitieve multimediatheorie. Ze bepalen hoe multimediale informatie/leerstof wordt verwerkt en hoe we die verschillende leermaterialen moeten ontwerpen (zie ontwerpprincipes).

2.3.5 De ontwerpprincipes

1) Het multimediaprincipe

   De graad waarin en hoe het geheugen informatie kan opslaan is afhankelijk van de inputs of kanalen die worden gebruikt om die informatie te bezorgen. Dit principe stelt dat het geheugen vlugger en efficiënter informatie kan opslaan wanneer woorden en beelden gecombineerd worden. Een combinatie van verbale en visuele modellen helpen om vlugger verbanden/relaties te kunnen leggen. Zo niet is het gevormde mentale model beperkter of onvolledig.

2) Het spatial contiguity principe

   Leren gaat beter en vlugger wanneer overeenstemmende woorden en figuren/beelden dicht bij elkaar worden gepresenteerd, zoals in cursussen of op computerschermen. Dit heeft ook vooral economische voordelen. Zowel in boeken als op computerschermen zorgt een te groot ruimtegebruik voor grotere kosten. Vanuit de multimedia theorie ligt dit anders. Daarin stelt men de vraag of het beter is die ruimte tussen tekst en beeld te behouden, weliswaar beperkt, of zoveel mogelijk te integreren. Men is geneigd toch steeds ruimte te laten tussen visuele en verbale informatie, toch zegt onderzoek dat het integreren van beide representaties beter is. Dit vermijdt dat studenten zelf moeten zoeken welke tekst bij welk beeld hoort, die op zijn beurt het werkgeheugen minder belast zodat het leren efficiënter gebeurt. Kortom wanneer men leermaterialen ontwikkelt, zorgt men best dat de overeenstemmende beelden en woorden dicht bij elkaar worden gepresenteerd.

3) Het temporal contiguity principe

   Voorafgaand wat men zei over het 'spatial contiguity' principe stelt men hier dat corresponderende woorden en beelden beter simultaan en niet opeenvolgend worden aangeboden. Het werkgeheugen wordt zo minder belast en kunnen er vlugger relaties worden gelegd wat het leren opnieuw bevorderd. Bij geschreven/gedrukte leermaterialen geldt dit principe ook. Wanneer een afbeelding op een andere pagina staat dan de corresponderende tekst gaat het begrijpen en het leren een stuk moeilijker.

4) Het coherentieprincipe

   Wanneer men een teveel aan materiaal aanbiedt heeft dit een negatieve invloed op het leren. Extra overbodige informatie moet dus worden vermeden. Het coherentieprincipe duidt ook aan dat, wanneer niet relevante geluiden worden aangeboden bij de informatie, dit ook het leren benadeelt. Een voorbeeld hiervan is een multimedia presentatie met niet relevante muziek of achtergrondgeluid. Het zorgt opnieuw voor een extra belasting van het werkgeheugen (cognitive load).

Zorg dus bij de ontwikkeling van leermaterialen dat je 'to the point' blijft en geen onnodige informatie aanbiedt.

5) Het modaliteitsprincipe

   Verder bouwend op de 'dual channel' assumptie komt men bij het modaliteitsprincipe terecht. De cognitieve verwerkingscapaciteit is groter wanneer zowel visuele- als audio-informatie worden aangeboden en verwerkt. De multimediatheorie van Mayer stelt dat leren efficiënter verloopt wanneer de informatie gesproken wordt gepresenteerd dan wanneer naast een beeld/animatie nog gedrukte tekst wordt aangeboden. Zo niet steunt de verwerking ervan enkel op het visuele kanaal. Dit gaat nochtans in tegen de intuïtie van vele ontwerpers.

   Worden zowel beelden met audio gepresenteerd dan worden beide cognitieve verwerkingskanalen aangesproken wat op zich de 'cognitive load' vermijdt en er is meer vrije ruimte voor het werkgeheugen.

6) Het redundantieprincipe

   De theorie van Sweller & Chandler stelt reeds dat het elimineren van extra overbodige informatie het resultaat, het leren bevordert (zie het coherentieprincipe). Mayer gaat daar terug verder op in en maakt een vergelijking tussen wanneer een combinatie van visuele- en audio-informatie wordt aangeboden of wanneer een combinatie van animatie, geluid met geschreven tekst wordt aangeboden.

   Conclusie: een combinatie leermaterialen van animatie en geluid doet studenten beter leren dan wanneer animatie wordt aangeboden gecombineerd met geluid en tekst. De 'dual channel' assumptie verklaart dit terug voor een stuk. Een combinatie van animatie en geschreven tekst zorgt voor een cognitieve overload van het visuele kanaal in het werkgeheugen.

   In principe wordt dezelfde informatie dubbel aangeboden, visueel en auditief. Vanuit de cognitieve multimediatheorie wordt die dubbel aangeboden informatie in vraag gesteld. Die stelt namelijk (zie 'limited capacity' assumptie) dat die dubbele informatie zorgt voor een overbelasting van het visuele kanaal van het werkgeheugen. Onderzoek bevestigt dit.

7) Het principe van de individuele verschillen

   Uit alle reeds besproken principes van de multimediatheorie blijkt dat studenten met weinig voorkennis en een groter ruimtelijk inzicht daar meer effect van ondervinden dan wanneer reeds een zekere voorkennis aanwezig is. Die laatste kunnen namelijk hun reeds aanwezige mentale modellen aanspreken om mogelijke tekorten in de multimediarepresentatie te compenseren. Een meer ontwikkeld ruimtelijk inzicht zorgt er op zijn beurt voor dat er beter en sneller een mentaal model kan worden gevormd, zo ook van minder goede multimedialeermaterialen.

   Het principe van de individuele verschillen zegt ons dus dat we bij de uitwerking van multimedialeermaterialen zeker rekening moeten houden met de doelgroep, rekening houdend met hun voorkennis. Die wordt onderverdeeld in 3 aanpakken:

   - leermaterialen specifiek aanpassen aan de kenmerken van de studenten;
   - laat de student/leerling zelf kiezen: ontwikkel leermaterialen voor de verschillende doelgroepen;
   - zorg voor vooropleiding, om de eventuele voorkennis en tekorten bij te schaven.

2.3.6 Besluit

Mayer stelt dus met zijn cognitieve multimediatheorie dat multimedia-instructie het verbale en visuele stimuleert, dat geïntegreerd kan worden met de beschikbare voorkennis, waardoor op die manier nieuwe informatie of kennis kan worden geconstrueerd.

De verschillende ontwerpprincipes maken ons duidelijk om na te denken over hoe men best leermaterialen ontwikkelt en uitwerkt.

Zo heb ik ook rekening gehouden met deze principes om mijn lessenpakket te ontwikkelen (zie Mayer toegepast in het lessenpakket).

2.4 Mayer toegepast in het lessenpakket

2.4.1 Het multimediaprincipe

Principe: de combinatie van woorden en beelden om zo vlugger verbanden en relaties te kunnen leggen.

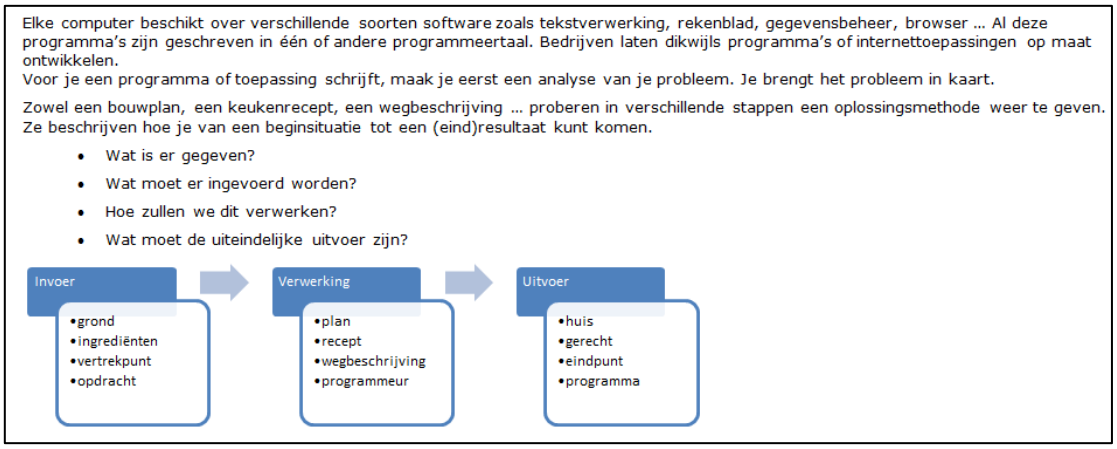


Fig. 45 analyse van een probleem (A. Fevery, 2014)

De figuur hierboven is een voorbeeld waarin het verbale en visuele worden gecombineerd. De inhoud van de tekst wordt versterkt en ondersteund door de figuur. Het voorbeeld voldoet niet helemaal aan het multimediaprincipe. Zowel woorden als beelden worden aangeboden, maar beide via het visuele kanaal (dual channel assumptie).

Een oplossing zou kunnen zijn, om de gedrukte tekst auditief aan te bieden. Aangezien het de bedoeling is dat de leerlingen de cursus raadplegen op de tablet, zou dat perfect haalbaar zijn.

Fig. 46 eindresultaat oefening Codea (YouTube,24 oktober 2013)

Het voorbeeld (figuur) hierboven is een beter voorbeeld waarin het verbale en visuele worden gecombineerd. Er komt animatie (visuele/beelden) en audio (verbale/woorden) in voor.

2.4.2 Het spatial contiguity principe

Principe: de afstand tussen de overeenstemmende woorden en beelden is minimaal.

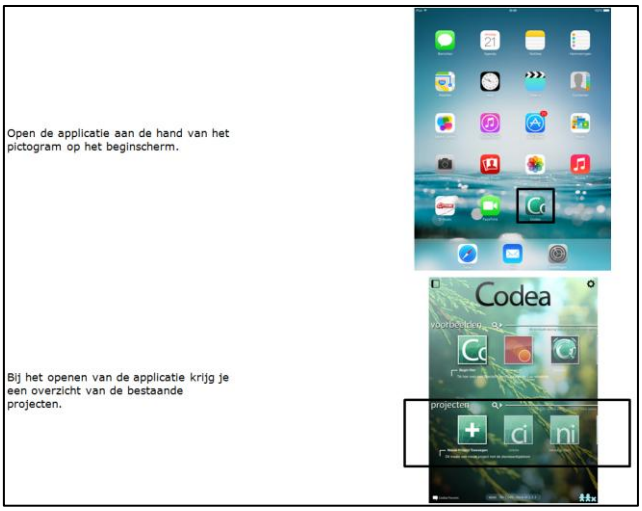


Fig. 47 installatie app Codea (A. Fevery, 2014)

De figuur hierboven toont een voorbeeld waarbij de beelden dichtbij de tekst staan.

2.4.3 Het temporal contiguity principe

Principe: corresponderende woorden en beelden moeten simultaan worden aangeboden.

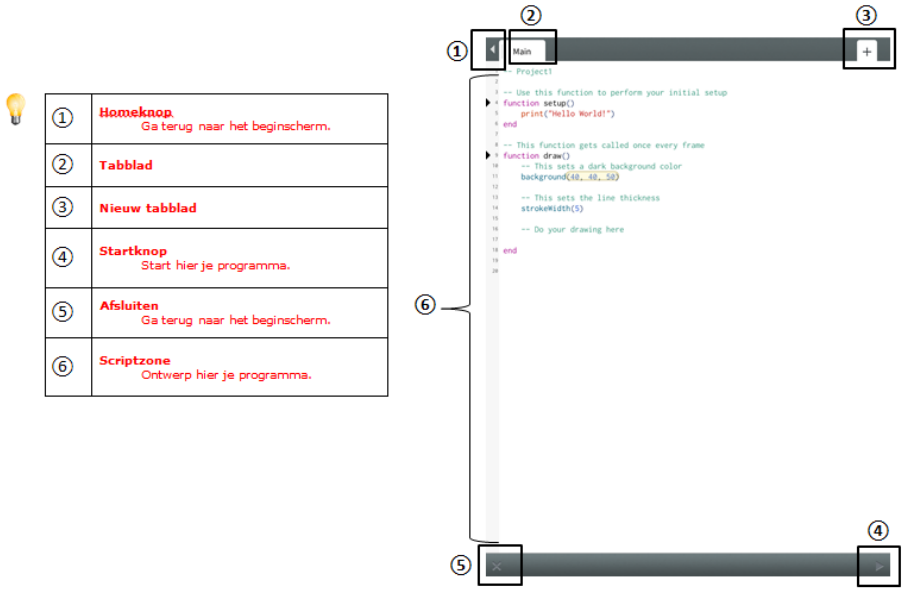


Fig. 48 verkennen van Codea (A. Fevery, 2014)

De figuur hierboven toont een voorbeeld waarbij de tekst en afbeeldingen simultaan worden aangeboden, in plaats van opeenvolgend of op een andere bladzijde zijn gedrukt.

2.4.4 Het coherentieprincipe

Principe: geen onnodige informatie aanbieden.

In ons dagelijks leven maken we (dikwijls onbewust) gebruik van algoritmes. Denk maar aan:

- de handleiding bij een zelfbouw-meubelpakket;
- de instructies van een gps;
- de recepten in een kookboek;
- het knopen van een das;
- ...

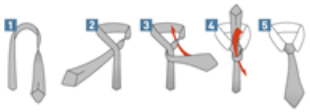


Fig. 49 algoritme (A. Fevery, 2014)

De tekst is duidelijk en de beelden zijn beperkt en effectief. De leerlingen kunnen zich concentreren op het maken van de verbindingen tussen tekst en beeld zonder afgeleid te worden door te veel extra materiaal.

2.4.5 Het modaliteitsprincipe

Principe: naast beeld/animatie een gesproken tekst horen.

Fig. 50 filmpje "waarom programmeren?" (YouTube, 2014)

Dit filmpje krijgen de leerlingen te zien bij het begin van de lessen rond programmeren. In dit filmpje wordt duidelijk waarom ze leren programmeren. In het filmpje wordt gesproken, er verschijnt zelden tekst op het scherm.

2.4.6 Het redundantieprincipe

Principe: een combinatie van animatie en geluid doet beter leren.

In het filmpje (zie modaliteitsprincipe) worden er vooral beelden/animatie getoond in combinatie met gesproken tekst. Er wordt dus geen geschreven tekst toegevoegd, wanneer dezelfde informatie al gesproken wordt meegegeven.

2.4.7 Het principe van de individuele verschillen

Principe: rekening houden met de doelgroep (voorkennis).

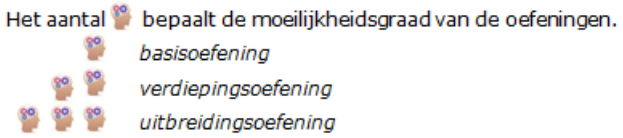


Fig. 51 moeilijkheidsgraden oefeningen (A. Fevery, 2014)

In de figuur hierboven wordt het systeem van de oefeningen getoond. Er zijn dus verschillende oefeningen ontworpen voor leerlingen met een hoge of lage voorkennis. De leerlingen krijgen zo de kans te kiezen tussen de verschillende oefeningen.

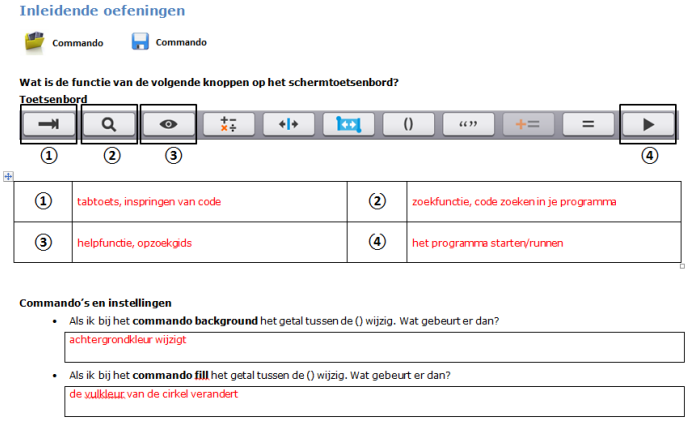


Fig. 52 voorbeeld inleidende oefening (A. Fevery, 2014)

Sommige inleidende oefeningen zijn zo ontworpen dat ook de leerlingen zelfstandig nieuwe theorie kunnen verwerken. De inleidende oefeningen kunnen ook perfect in groepjes of klassikaal gemaakt worden.

2.5 Aanpak

2.5.1 Algemeen

In dit deel wil ik even mijn visie over de didactische aanpak van de cursus uiteenzetten.

De leerlingen moeten in het bezit zijn van een iPad, de applicatie Codea en de applicatie PDF Expert.

PDF Expert is een applicatie waarmee je PDF-documenten kan (in)lezen. In deze documenten kan je dan notities aanbrengen, stukken tekst markeren en andere aantekeningen maken.

De competentie algoritmisch denken wordt aangebracht met de applicatie Codea op de iPad.

De bijhorende cursus is gemaakt om (hoofdzakelijk) als digitale cursus te gebruiken. Het is hoofdzakelijk de bedoeling dat de leerlingen de digitale cursus kunnen invullen aan de hand van PDF Expert.
Ik heb ondervonden dat het ook mogelijk is om de cursus op papier te gebruiken en om enkel de iPad te gebruiken voor de applicatie Codea.

De cursus is opgedeeld in verschillende hoofdstukken. Ik heb de hoofdstukken zo gekozen dat er per hoofdstuk aan één bepaalde competentie wordt gewerkt. In elk hoofdstuk kun je dezelfde structuur terugvinden.

In het begin van een hoofdstuk staan telkens alle doelstellingen vermeld. Dit kunnen de leerlingen gebruiken bij het studeren, ook wel kapstokken genoemd.

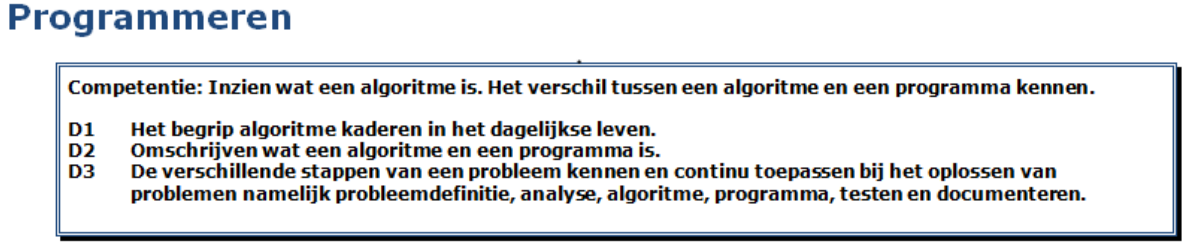


Fig. 53 doelstellingen hoofdstuk programmeren (A. Fevery, 2014)

Na de doelstellingen wordt de theorie uitgelegd. In sommige hoofdstukken is de theorie uitgeschreven, in andere hoofdstukken moeten de leerlingen aan de hand van inleidende oefeningen zelf op zoek gaan naar de theoretische achtergrond.
In de hoofdstukken waar de theorie is uitgeschreven is het de bedoeling dat dit klassikaal uitgelegd wordt. Als in deze uitgeschreven theorie belangrijke tot zeer belangrijke uitleg staat wordt dit aangeduid met een lampje.

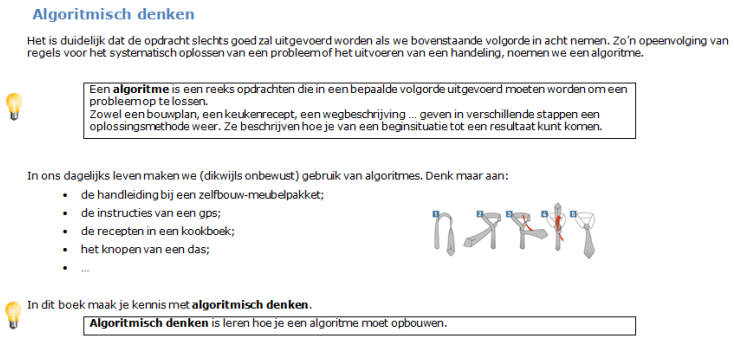


Fig. 54 voorbeeld uitgeschreven theorie (A. Fevery, 2014)

In de hoofdstukken waar met inleidende oefeningen wordt gewerkt kan de leerkracht kiezen of dit zelfstandig door de leerlingen of klassikaal wordt verwerkt.

De theorie met eventuele inleidende oefeningen wordt gevolgd door een commando-overzicht. Deze lijst geeft de geziene commando's in dat hoofdstuk in een overzicht weer. De leerlingen kunnen dit gebruiken bij het maken van de oefeningen.

Op het einde van elk hoofdstuk zijn er een reeks oefeningen.
Deze oefeningen zijn opgedeeld in (oefen)niveaus. Het is de bedoeling dat de leerling zelf weet of aanvoelt welk niveau hij aankan, om dan tijdens de oefeningen tot een hoger niveau te raken/komen.
Elke leerling zou tot het niveau van de verdiepingsoefeningen moeten komen om de doelstellingen te behalen.

Het aantal  bepaalt de moeilijkheidsgraad van de oefeningen.
Hieronder een overzicht van de gebruikte symbolen:

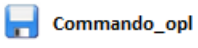
 basisoefening

verdiepingsoefening

uitbreidingsoefening

Voor sommige oefeningen of inleidende oefeningen hebben de leerlingen een startbestand nodig.
De leerlingen kunnen alle startbestanden vinden op de volgende link:
https://docs.google.com/document/d/1H8w843kaljUlksCsn1bzUyBZLOxQD9YRwq03xPjt3FI/edit?usp=sharing

De startbestanden die de leerlingen nodig hebben worden weergegeven met volgend symbool: 

Naast het symbool van het startbestand, wordt ook telkens aangegeven hoe de leerlingen de oefening moet opslaan. Dit wordt weergegeven met volgend symbool: 

Het vraagt nogal wat tijd om de startbestanden vanuit Google Drive te kopiëren naar de applicatie Codea. Ik raad aan om dit op een moment te doen wanneer de leerlingen beschikken over een computer en over een iPad op eenzelfde tijdstip.
Hieronder een stappenplan:

- Computer: Surf naar onderstaande link
  https://docs.google.com/document/d/1H8w843kaljUlksCsn1bzUyBZLOxQD9YRwq03xPjt3FI/edit?usp=sharing
- Leg kort de bedoeling uit aan de leerlingen.

- iPad: Maak een nieuw project aan in Codea. Je geeft het project de naam van het startbestand.
- iPad: Zet AirCode aan op de iPad. Er verschijnt nu een code op het scherm.
- Laptop: Maak een nieuw tabblad aan en surf naar de code die op de iPad verschijnt.
- Laptop: Open het project waar je de code wil plakken.
- Laptop: Kopieer de code uit Google Drive (eerste tabblad) naar het project (tweede tabblad)

Om aan de attitude 'eerst denken en dan doen' te werken, wordt in de cursus gewerkt met een bepaald stappenplan.
De uitleg van het stappenplan vind je in deze scriptie bij het hoofdstuk 'eerst denken dan doen'.

Het is de bedoeling dat de leerkracht (bijna) elke les een nieuw stukje theorie uitlegt of dat de leerlingen een nieuw stukje theorie zelfstandig verwerken.
Daarna kunnen de leerlingen verder werken aan de oefeningen van de vorige lessen of beginnen aan de oefeningen van de nieuwe theorie.
Het is niet de bedoeling dat alle leerlingen telkens met dezelfde oefeningen bezig zijn in iedere les. De leerlingen verwerken alle oefeningen op zelfstandig basis op hun eigen tempo. Hiervoor is er een planning gemaakt, zodat de leerlingen weten wanneer welke oefeningen moeten afgewerkt zijn (zie bijlage).

Op het einde van de cursus vind je een overzicht van de geziene controlestructuren en commando's.

2.5.2 Leerplan informatica

inleiding

De cursus is, uiteraard, gebaseerd op de doelstellingen die het leerplan oplegt. Ik heb gewerkt met het leerplan informatica 2011/045 (tweede graad tso Handel).
Ik heb dit leerplan moeten gebruiken omdat ik de cursus heb uitgetest in het vierde jaar handel (zie toepassing binnen de stage).

Deelcompetenties

De cursus is gericht op competentie 11: algoritmisch denken.

In het leerplan wordt aangegeven om 20 lestijden te spenderen aan deze competentie. Ik heb een planning opgesteld die weergeeft hoeveel lestijden je nodig hebt om de cursus te geven (zie verder: aanpak - leerkracht). In de planning heb ik geen 20 lestijden nodig om alle competenties te bereiken.

Deelcompetentie	**Hoofdstuk**
11.1 Inzien wat een algoritme is. Het verschil tussen een algoritme en een programma kennen.	
11.1.1 Het begrip algoritme kaderen in het dagelijkse leven.	Hoofdstuk 'programmeren'
11.1.2 Omschrijven wat een algoritme en een programma is.	Hoofdstuk 'programmeren'
11.1.3 De verschillende stappen in het oplossen van een probleem kennen en continu toepassen bij het oplossen van problemen nl. probleemdefinitie, analyse, algoritme, programma, testen en documenteren	Hoofdstuk 'programmeren'
11.2 Een probleemstelling omzetten in een werkend programma. De verschillende controlestructuren	
11.2.1 Ongeacht de eenvoud of de complexiteit van een probleem, een analyse maken en vooraleer tot het gebruik van de computer over te gaan, minimaal voor zichzelf het principe van een oplossing formuleren.	Hoofdstuk 'programmeren'

11.2.2 De verschillende elementen en mogelijkheden van de gebruikte ontwikkelomgeving doelgericht aanwenden	Hoofdstuk 'het verkennen van Codea'
11.2.3 Met variabelen en constanten werken.	Hoofdstuk 'constanten en variabelen'
11.2.4 De toekenningsoperator gebruiken.	Hoofdstuk 'constanten en variabelen'
11.2.5 Rekenkundige-, vergelijkings- en logische operatoren integreren.	Hoofdstuk 'constanten en variabelen'
11.2.6 De controlestructuren met hun kenmerken kennen en toepassen waaronder de sequentie, de selectie en de iteratie.	Hoofdstuk 'de sequentie'
11.2.7 De eenzijdige, tweezijdige en geneste selectie of keuze toepassen.	Hoofdstuk 'de tweezijdige selectie' Hoofdstuk 'de geneste selectie' Hoofdstuk 'de meervoudige selectie'
11.2.8 De voorwaardelijk en begrensde herhaling gebruiken	Hoofdstuk 'iteraties of lussen'

In de overzichtstabel hierboven kan je zien dat alle deelcompetenties aan bod komen in het cursus. Deze tabel geeft je ook een overzicht in welke hoofdstukken aan welke deelcompetenties wordt gewerkt.
Sommige deelcompetenties komen in alle hoofdstukken aan bod, ik heb dan het (de)hoofdstuk(ken) vermeld waarin de klemtoon wordt gelegd.

In de figuur hieronder kan je zien dat er per hoofdstuk in de cursus telkens aangegeven wordt aan welke competenties wordt gewerkt. Bij elke competentie zijn er bijhorende doelstellingen opgesteld.

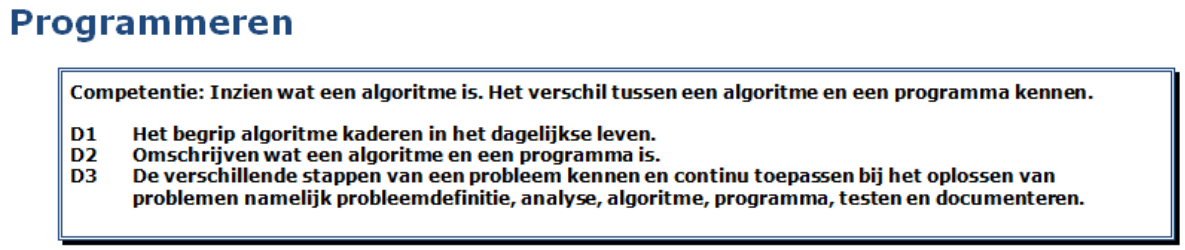


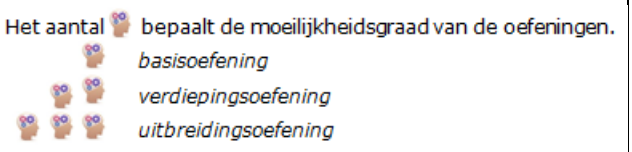
Fig. 55 voorbeeld hoofdstuk 'programmeren' (A. Fevery, 2014)

Algemene didactische wenken

In het leerplan bij competentie 11 worden ook een aantal algemene didactische wenken meegegeven. Bij het maken van de cursus heb ik hier ook rekening mee gehouden.

Hieronder een overzicht:

Algemene didactische wenk	Uitwerking in de cursus
"Eerst denken en dan doen" is het belangrijkste bij algoritmisch denken.	Aan deze attitude is veel aandacht besteed in de cursus. Hiervoor verwijs ik naar de onderzoeksvraag 'eerst denken dan doen' die in deze scriptie uitgebreid is beschreven.

Bij complexere problemen een algoritme schematisch (laten) voorstellen door een Nassi-Shneiderman diagram. In de andere gevallen volstaat een uitgebreide beschrijving van de verwerkingsfase in de analyse.	Ook hiervoor verwijs ik naar de onderzoeksvraag 'eerst denken dan doen'.
Gebruik een actuele, motiverende ontwikkelomgeving.	Codea is een ontwikkelomgeving die vaak gebruikt wordt om games te ontwerpen (zie hoofdstuk lessenpakket: Codea + Lua). Het is ook de bedoeling dat de leerkracht in de eerste lessen de leerlingen warm maakt, door een aantal games te tonen. Deze games zouden de leerlingen op het einde van de lessenreeks zelfs moeten kunnen (na)maken.
Toon eens de werking van een gemaakt programma in een professionele ontwikkelomgeving.	De leerkracht kan in het begin van de lessenreeks het spel Angry Birds tonen. Dit spel is gemaakt met Codea.
De complexiteit van en de soort oefeningen zijn afhankelijk van de studierichting. In een meer wiskundige richting komen meer wiskundige probleemstellingen aan bod.	De complexiteit van de oefeningen is aangegeven als volgt:  De leerlingen kunnen kiezen met welke oefening ze beginnen. De oefeningen aanpassen aan de studierichting (bv. handel) is met deze ontwikkelomgeving niet mogelijk. Aangezien je met Codea vooral games kan ontwikkelen/programmeren.
Geef de leerlingen de tijd en de ruimte om zelfstandig oefeningen uit te werken. Succeservaringen zijn heel belangrijk.	Ik heb een planning opgesteld die weergeeft welke oefeningen de leerlingen moeten afwerken tegen een bepaalde datum. Zo kunnen ze zelfstandig aan de slag. (zie verder: aanpak - leerkracht)
Vertrek vanuit de fouten van de leerlingen bij het aanleren van foutanalyse. Leer hen zelf hun fouten te analyseren. Gebruik hiervoor onder meer de mogelijkheden van de ontwikkelomgeving.	Codea is een goede omgeving om te werken aan foutanalyse. Indien er een fout is gesloopt in de code, duidt Codea dit aan in een bepaalde kleur.
Leer de leerlingen ook een werkend programma aanpassen.	De leerlingen moeten in verschillende oefeningen starten met een startbestand waarin al code is geschreven. Hieronder een voorbeeld: 

2.5.3 Leerkracht

Bij de uitwerking van de cursus is er ook gedacht aan de leerkracht.

Ik heb een aantal dingen gemaakt die voor de leerkracht nuttig kunnen zijn:

- Er is een leerkrachtenversie van de cursus ter beschikking. Dit houdt in dat er een oplossingsmodel is van alle oefeningen. Je kan dit terugvinden op de CD die bij deze scriptie zit.
- Er is uitleg over de aanpak van de cursus voorzien. Zodat de leerkracht weet wat de bedoeling is van de cursus. Je kan dit terugvinden op de CD.
- Er is een planning opgesteld zodat de leerlingen tijdens een oefenmoment zelfstandig aan de slag kunnen. Deze planning vind je terug in de bijlagen van deze scriptie en op de CD.
- Alle startbestanden die de leerlingen nodig hebben staan in een Google Drive document. De leerkracht hoeft daar niets meer aan te wijzigen, kant-en-klaar materiaal dus!
  Je kan alle startbestanden terugvinden op de volgende link:

  https://docs.google.com/document/d/1H8w843kaljUlksCsn1bzUyBZLOxQD9YRwq03xPjt3FI/edit?usp=sharing

  De link staat ook in het begin van de cursus vermeld.

3 Toepassing binnen de stage

3.1 Stage

Ik liep 6 weken stage in het Sint-Pieterscollege/Sint-Jozefshandelsschool te Blankenberge.
Ik had daar de eer om te kunnen werken met leerlingen die elk over een eigen iPad beschikken.

Ik gaf informatica in 3 handel talen (2 uren per week) en 4 handel (4 uren per week).

School uit Blankenberge mag iPad dan toch invoeren

14/09/2012 om 16:57 - Bijgewerkt op 19/08/2013 om 15:20

Het Blankenbergse Sint-Pieterscollege/Sint-Jozefshandelsschool mag dan toch het gebruik van de iPad op school verplichten. De Vlaamse regering heeft daarvoor het licht op groen gezet.

Fig. 56 artikel iPadschool (Knack, 14 september 2012)

3.2 Toepassing

3.2.1 Inleiding

Ik gaf 2 weken algoritmisch denken, met mijn eigen cursus, aan de klas 4 handel. In deze klas zaten 17 leerlingen die 4 uren informatica per week hebben. De leerlingen hadden nog geen programmeren gekregen en hadden dus nog geen voorkennis van deze materie.

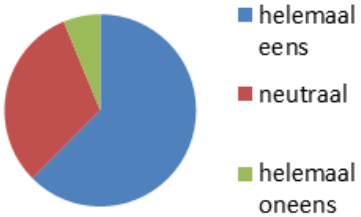
In de bijlagen van deze scriptie vind je de lesvoorbereiding van deze lessen.

3.2.2 Evaluatie

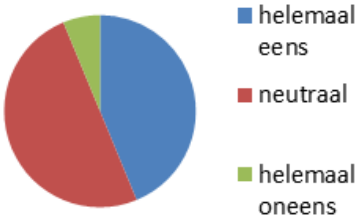
Ik vond het belangrijk om ook eens de mening van de leerlingen te weten (Codea, cursusmateriaal, …). Na het uittesten van deze lessen stelde ik een enquête op voor de leerlingen.
De enquête vind je terug in de bijlagen.

De resultaten van de enquête werkte ik hieronder uit.

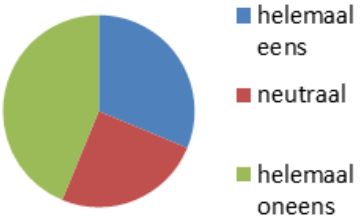
Codea wekt mijn interesse, omdat ik met spelletjes bezig ben.



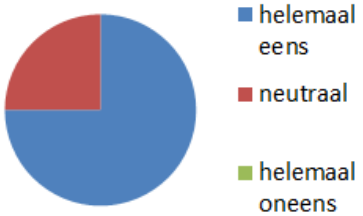
Programmeren vind ik het moeilijkste onderwerp binnen het vak informatica.



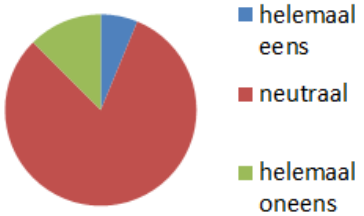
Ik was snel weg met de interface (knoppen, projecten aanmaken, ...) van Codea.



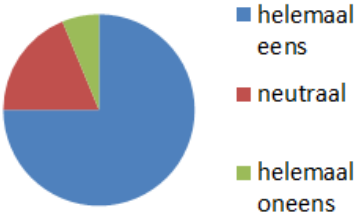
De oefeningen zijn overzichtelijk in de cursus.



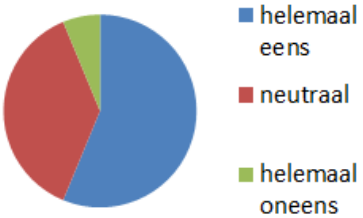
Ik vind het leuker/beter dat we programmeren met de iPad dan met de computer.



Ik zou nog wel wat meer willen leren over Codea/programmeren.



De oefeningen zijn uitdagend (verschillende moeilijkheidsgraden).



Wat ik nog wil zeggen over Codea, programmeren en/of de cursus:

Enkele (letterlijke) antwoorden van leerlingen:

- Het interesseert me.
- Programmeren is leuk en zeker op de app Codea.
- Codea is tof!
- Gestructureerde cursus.
- Programmeren is niets voor mij...
- Wanneer je het programmeren kan, wordt het leuker.
- Het was leuk en interessant.
- Programmeren is vrij moeilijk.

3.2.3 Besluit evaluatie

Na de uitwerking van de resultaten van de enquête, ben ik tot volgende conclusies gekomen:

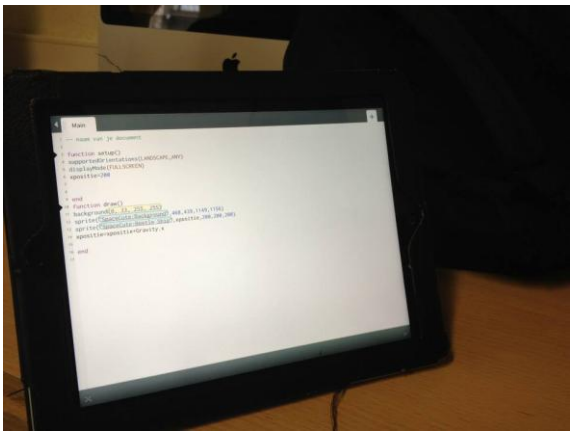
- De leerlingen vinden Codea niet bepaald leuker of boeiender vanwege het iPadgebruik, maar wel omdat ze vooral met spelletjes bezig zijn. Dit is toch wel een opmerkelijk verschil in visie tegenover andere programmeeromgevingen, waar de leerlingen meestal minder enthousiast over zijn.
- Programmeren blijft een moeilijk onderwerp voor vele leerlingen ook met de applicatie Codea.
- De applicatie Codea vinden de leerlingen moeilijk op het gebied van interface.
- De cursus vinden de leerlingen overzichtelijk en de oefeningen zijn uitdagend genoeg.

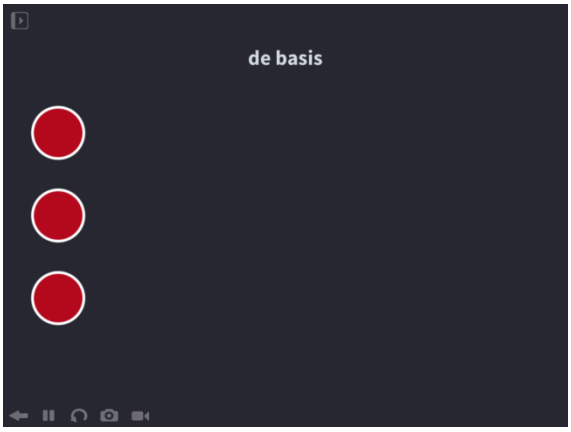
Na mijn stagelessen heb ik ondervonden dat ik nog een aantal aanpassingen moest doen aan mijn cursus. Hieronder zie je de aanpassingen die ik heb aangebracht:

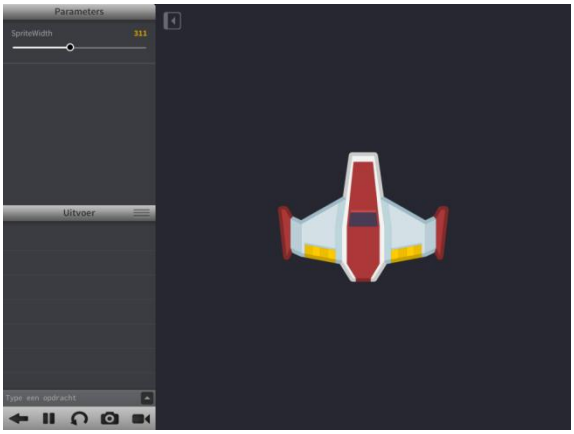
- Ik had de oefeningen vergeten te nummeren. Ik heb dus een nummering aangebracht bij de oefeningen. De nummering begint per hoofdstuk opnieuw.
- Ik heb ook een aantal oefeningen moeten wijzigen van (oefen)niveau. Een aantal oefeningen had ik te laag ingeschat waardoor ik de oefeningen bij basisoefeningen had geplaatst. Sommige oefeningen pasten beter bij de categorie verdiepingsoefeningen, ik heb dat dan ook aangepast.
- Ik heb ondervonden dat de leerlingen tijdens oefeningen erg lang moesten zoeken naar bepaalde commando's die in theorie vermeld staan. Ik heb dus op het einde van elk hoofdstuk een overzicht gemaakt van alle commando's die gezien zijn in dat hoofdstuk. Ik denk dat dit een goed hulpmiddel zal zijn voor de leerlingen tijdens oefenmomenten.
- Het overzetten van de startbestanden op de iPads is vrij omslachtig. Het werken met een Google Drive account is de beste oplossing maar nog steeds omslachtig. Hiervoor heb ik helaas geen oplossing kunnen vinden. Ik hoop dat Codea, omtrent dit probleem, snel met een oplossing komt.
- Tijdens mijn stagelessen had ik de gewoonte om elke les een nieuw stuk theorie te geven. Daarna was er telkens een oefenmoment waarin de leerlingen de oefeningen maakten over het nieuwe stukje theorie.
  Dat was geen goed idee, de ene leerling was veel sneller klaar dan de andere en vooral de ene leerling had het vorige hoofdstuk al heel goed onder de knie en de andere kon nog wel wat oefening gebruiken. Daarom heb ik de aanpak van de cursus wat gewijzigd. Ik heb een planning opgesteld zodat de leerlingen de oefeningen zelfstandig en op eigen tempo kunnen verwerken.
  Meer uitleg over deze aanpak vind je in deze scriptie bij aanpak - algemeen.

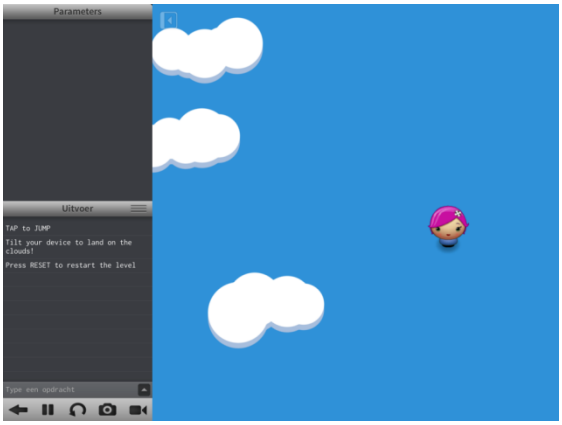
3.2.4 Sfeerbeelden

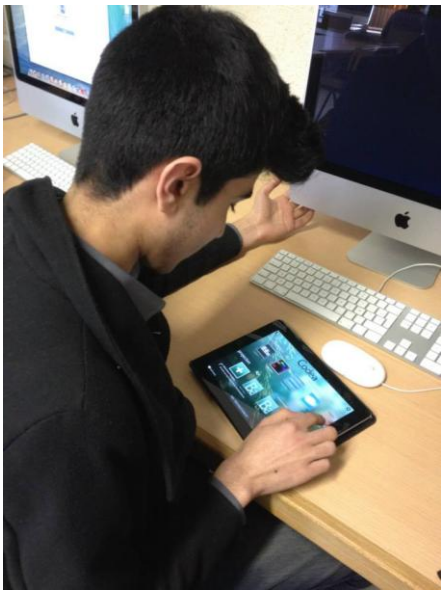
Hieronder wat sfeerbeelden van tijdens deze (test)lessen.











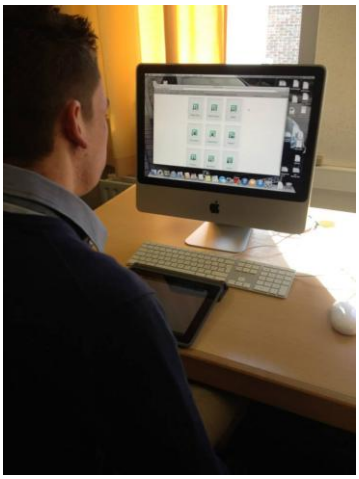


Fig. 57 status Sint Jozef Sint Pieter (Facebook, 2014)

Fig. 58 bericht Sint Jozef Sint Pieter (Facebook, 2014)

4 Besluit

De eerste onderzoeksvraag ging uit naar de aanpak van het onderwerp algoritmisch denken in andere landen. Voor dit onderzoek heb ik mij toegespitst op de leerplannen van Wallonië, Nederland en het Verenigd Koninkrijk.
De leerplannen van Wallonië en Nederland staan nog niet zo ver als die bij ons. Bij beide wordt programmeren wel gegeven maar de inhoud van het vak is van een lager niveau. Het Verenigd Koninkrijk staat al een stapje verder dan wij. De inhoud van dit onderwerp is veel diepgaander dan bij ons.

De tweede onderzoeksvraag ging uit naar de kennis rond algoritmisch denken in het lager onderwijs. Om op deze onderzoekvraag een antwoord te kunnen formuleren stelde ik een enquête op.
De leerkrachten uit het basisonderwijs hebben zeer weinig kennis hebben over het onderwerp, zo concludeerde ik ook dat het onderwerp niet tot weinig gegeven werd in het basisonderwijs. Een andere vraag in de enquête rond het gebruik van programmeren in de klas bevestigde deze conclusie.
Een kleine meerderheid van de leerkrachten en een groep leerlingen vinden dat programmeren in het basisonderwijs zou moeten gegeven worden. Terwijl de andere groep leerlingen vinden dat programmeren zou moeten gegeven worden vanaf de eerste graad secundair onderwijs. Iedereen is het eens dat de programmeren vroeger moet gegeven worden dan de tweede graad secundair onderwijs, maar men is het niet eens vanaf welke leeftijd dit dan zou moeten gebeuren.

De derde onderzoeksvraag ging uit naar welke applicatie het meest geschikt zou zijn om een lessenpakket rond op te bouwen. Ik stelde een evaluatiefiche op, om dit te kunnen onderzoeken.
De applicaties Hopscotch en Daisy the dinosaur zijn goede initiatieapplicaties maar niet te gebruiken in het secundair onderwijs. De applicaties Kodable, Cargo-Bot en Codea zijn bruikbaar voor het secundair onderwijs. Codea kwam hier als de beste applicatie uit waardoor ik mijn lessenpakket rond deze applicatie opbouwde.

De vierde en laatste onderzoeksvraag ging uit naar de didactische wenk "eerst denken dan doen" bij algoritmisch denken. Bij deze onderzoeksvraag heb ik een methode/stappenplan uitgewerkt om aan deze attitude te kunnen werken. Dit stappenplan wordt ook gebruikt doorheen het hele lessenpakket.

Het lessenpakket dat ik heb uitgewerkt is met de applicatie Codea, die in de programmeertaal Lua is geschreven. Het lessenpakket is samengesteld aan de hand van de cognitieve multimediatheorie van Mayer. Alle principes van deze theorie zijn toegepast in het lessenpakket.

Daarnaast heb ik ook het lessenpakket toegepast in mijn stage en nog verder aangepast waar nodig.

5 Lijst van tabellen en figuren

6 Bronnenlijst

Klasse (1 april 2009). *Acht verschillen tussen Vlaams en Waals onderwijs*. Geraadpleegd op 5 december 2013 via http://www.klasse.be/archief/acht-verschillen-tussen-vlaams-en-waals-onderwijs/

Wetenwind Ryckevelde (2014). *Onderwijssystemen*. Geraadpleegd op 20 oktober 2013 via http://westenwind.ryckevelde.be/nl/onderwijssystemen-6433.html

Department for Education (11 september 2013). *National curriculum in England: computing programmes of study*. Geraadpleegd op 12 november 2013 via https://www.gov.uk/government/publications/national-curriculum-in-england-computing-programmes-of-study/national-curriculum-in-england-computing-programmes-of-study

Lua.org (2011). *Lua 5.2 Reference Manual*. Geraadpleegd op 14 september 2013 via http://www.lua.org/manual/5.2/

VVKSO (1 september 2011). *Leerplan informatica 2011/039*. Geraadpleegd op 20 september 2013 via http://ond.vvkso-ict.com/vvksomainnieuw/leerplanpubliek.asp?NR=2011/039

Universiteit Nederland, Eindhoven en Twente (2009). *Positie van het vak informatica in havo/vwo*. Geraadpleegd op 3 februari 2014 via http://www.ou.nl/documents/14300/a97edc83-fa16-45c0-8cd8-40de4456cab5

Kennisnet (december 2011). *Zelf programmeren*. Geraadpleegd op 19 februari 2013 via http://www.kennisnet.nl/fileadmin/contentelementen/kennisnet/Onderzoek/Documenten/KNS_Programmeeronderwijs_FINAL.PDF

Kennisnet (1 juli 2011). *Rendement van objectgeoriënteerd programmeeronderwijs*. Geraadpleegd op 4 april 2014 via http://www.kennisnet.nl/uploads/tx_kncontentelements/eindverslag.pdf

Kennisnet (onbekend). *Objectgeoriënteerd programmeren in het voortgezet onderwijs*. Geraadpleegd op 4 april 2014 via http://www.kennisnet.nl/uploads/tx_kncontentelements/OOprogrammerenOpVOV2.pdf

Kennisnet (20 januari 2012). *Objectgeoriënteerd programmeren in het voortgezet onderwijs*. Geraadpleegd op 4 april 2014 via http://www.kennisnet.nl/uploads/tx_kncontentelements/Video_-_ORD.pdf

Klasse (8 november 2013). *Tot een op de vijf lessen in secundair in andere taal*. Geraadpleegd op 31 maart 2014 via http://www.klasse.be/leraren/39465/tot-een-op-vijf-lessen-in-secundair-in-andere-taal/

Codekinderen (onbekend). *Mijn kind online*. Geraadpleegd op 25 november 2013 via http://www.codekinderen.nl/

Microsoft Research (onbekend). *Kodu*. Geraadpleegd op 25 maart 2014 via http://research.microsoft.com/en-us/projects/kodu/

Leesvoer (5 december 2013). *Zelfs games maken zonder programmeerkennis met Kodu*. Geraadpleegd op 25 maart 2014 via http://www.leesvoer.be/geen-categorie/zelf-games-maken-zonder-programmeerkennis-met-kodu

Knack (14 september 2012). *School uit Blankenberge mag iPad dan toch invoeren*. Geraadpleegd op 25 maart 2014 via http://www.knack.be/nieuws/technologie/school-uit-blankenberge-mag-ipad-dan-toch-invoeren/article-normal-65967.html

Vandaag.be (14 september 2012). *Kinderen leren programmeren in Estse basisonderwijs*. Geraadpleegd op 5 februari 2014 via http://www.vandaag.be/entertainment/119784_kinderen-leren-programmeren-in-estse-basisonderwijs.html

Leesvoer (3 maart 2014). *Moet elke leerling kunnen programmeren?* Geraadpleegd op 5 maart 2014 via http://www.vandaag.be/entertainment/119784_kinderen-leren-programmeren-in-estse-basisonderwijs.html

Leesvoer (3 maart 2014). *Moet elke leerling kunnen programmeren?* Geraadpleegd op 5 maart 2014 via http://www.vandaag.be/entertainment/119784_kinderen-leren-programmeren-in-estse-basisonderwijs.html

Bitescience (5 februari 2013). *iPad makes children enthusiastic to learn but doesn't ensure higher grades.* Geraadpleegd op 5 maart 2014 via http://www.vandaag.be/entertainment/119784_kinderen-leren-programmeren-in-estse-basisonderwijs.html

Frankwatching.com (21 oktober 2013). *Programmeren? Dat leer je op de basisschool!* Geraadpleegd op 5 maart 2014 via http://www.frankwatching.com/archive/2013/10/21/programmeren-dat-leer-je-op-de-basisschool/

CoderDojoBelgium (onbekend). *CoderDojoWAT?.* Geraadpleegd op 4 april 2014 via http://www.frankwatching.com/archive/2013/10/21/programmeren-dat-leer-je-op-de-basisschool/

Onderwijs Vlaanderen (2007). *Lager onderwijs - ICT - Leergebiedoverschrijdende eindtermen.* Geraadpleegd op 4 april 2014 via http://www.ond.vlaanderen.be/curriculum/basisonderwijs/lager-onderwijs/leergebiedoverschrijdend/ict/algemeen.htm

De Smet, N. (2010). *iPad integratie in het basisonderwijs.* Geraadpleegd op 28 september 2013 via http://www.scriptiebank.be/en/node/2322

Vincent, T. (4 maart 2012). *Ways to evaluate educational apps.* Geraadpleegd op 28 september 2013 via http://www.scriptiebank.be/en/node/2322

Haese, I. & Valcke, M. (2005). Digitaal leren. Tielt: Lannoo.

Smeulders, L. (2013). Desktopper: multimedia en web 2.0. Averbode: Averbode.

Mesdom, F. & Steppe, G. (2013). iSee: multimedia en presentatie. Brugge: die Keure.

Coppens, J., Debruyn, L., Garrevoet, W., Goris, M., Van der Eedt, P., Van Deuren, D. & Verschraege, M. *Informatica*. Kalmthout: Pelckmans

Smeulders, L. (2013). Desktopper: algoritmisch denken. Averbode: Averbode.

Mesdom, F. & Steppe, G. (2013). iSee: algoritmisch denken. Brugge: die Keure.

7 Bijlagen