2017-2018

Onderzoek Yvette Vianen

Hoe beïnvloedt leeftijd van basisschoolleerlingen het bevorderen van computational thinking skills?

Naam: Yvette Vianen 140099

Begeleider: Petra Jansen

Datum: 12-04-2018

Naam profiel: Wereld, wetenschap en technologie

Inhoudsopgave

Samenvatting blz. 2

Hoofdstuk 1: Inleiding blz. 3

- Verlegenheidssituatie praktijkschool
- Aanleiding en doel
- Hoofdvraag
- Deelvragen praktijk
- Deelvragen theorie
- Verantwoording groepen

Hoofdstuk 2: Theoretisch kader blz. 5

- Computational thinking
- Programmeren
- Waarom programmeren?
- 21-Eeuwse vaardigheden
- Computational thinking bij jonge basisschoolleerlingen
- Computational thinking bij oudere basisschoolleerlingen
- Conclusie

Hoofdstuk 3: Methode blz. 11

- Deelvraag 1 t/m 7

Hoofdstuk 4: Resultaten blz. 16

- Deelvraag 1 t/m 7

Hoofdstuk 5: Conclusie en discussie blz. 26

- Discussie en aanbevelingen vervolgonderzoek
- Aanbevelingen voor de praktijk

Literatuurlijst blz. 29

Bijlagen blz. 31

- Bijlage 1 t/m 5: onderzoeksinstrumenten
- Bijlage 6: drie observaties
- Bijlage 7: presentatie team
- Bijlage 8 tot en met 16: uitwerking resultaten

Procesformulier blz. 56

Samenvatting

De Howiblo wil hedendaags en vernieuwend onderwijs, maar geven hier in het schooljaar 2017/2018 nog geen vorm aan in de praktijk. De school vindt programmeren interessant, maar ze weten niet of het in de onderbouw al veel zin heeft om aan te bieden. Daaruit ontstond de volgende hoofdvraag: Hoe beïnvloedt leeftijd van basisschoolleerlingen het bevorderen van computational thinking skills op de Howiblo? Het doel van dit onderzoek is het verbeteren van de computational thinking skills in de groepen 3 en 8 op de Howiblo, zodat de leerlingen kennismaken met innovatief onderwijs. Computational thinking is een 21-eeuwse vaardigheid en de 21-eeuwse vaardigheden hebben ze nodig in de toekomst, vandaar dat deze vaardigheden in het hedendaagse onderwijs moeten worden aangeleerd (Kennisnet, 2016.). Computational thinking is het denkproces waardoor problemen en hun oplossingen effectief uitgevoerd kunnen worden door een informatie verwerkende tussenpersoon (Kennisnet, 2016.). Programmeren is een middel voor computational thinking en houdt in dat leerlingen leren om te creëren in plaats van consumeren (Mediawijsheid, z.j.). Een andere bron voegt hieraan toe dat programmeren het maken van een programma voor verplaatsbare objecten door machines, robots of tools is (Jeuring, Corbalan, Montfort, Es & Leeuwestein, 2016). Er is een doorlopende leerlijn programmeren van groep 1 tot en met 8. Hieraan zijn verschillende denkvaardigheden gekoppeld. Voor het onderzoek hebben de desbetreffende leerlingen een vragenlijst ingevuld en zijn de twee groepsleerkrachten geïnterviewd om de beginsituatie in kaart te brengen. De leerkrachten hadden weinig kennis en ervaring met programmeren in de klas. De leerlingen leek het erg leuk, maar dachten dat ze er niet goed in waren. Zes leerlingen van groep 3 en zes leerlingen van groep 8 hebben een nulmeting gehad met Scratch. De leerlingen gebruikten voldoende scripts, maar hun reeksen waren eenvoudig.

Daarna hebben de leerlingen drie keer geprogrammeerd. Groep 3 heeft gewerkt met Codewise, de Blue-Bot en geprogrammeerd met plastic bekers. Groep 8 heeft gewerkt met de Microbit, Codemonster en hebben ook geprogrammeerd met plastic bekers. Vervolgens is er een effectmeting uitgevoerd, waarin de leerlingen vonden dat ze beter konden programmeren en het erg leuk vonden. Door de drie effectmetingen kan er geconcludeerd worden dat de leerlingen van groep 3 in totaal meer scripts gebruikten in tegenstelling tot groep 8. De ontwikkeling van de leerlingen verschilt per denkvaardigheid. De leerlingen uit groep 3 hebben meer scripts gebruikt en zijn meer vooruitgegaan in de denkvaardigheden dan groep 8. Echter hebben de leerlingen uit groep 8 een grotere ontwikkeling doorgemaakt in het gebruikmaken van de verschillende denkvaardigheden. Deze resultaten zijn aan het team gepresenteerd en het team heeft kennisgemaakt met enkele vormen van programmeren. Het advies voor de school is om het gestructureerd en schoolbreed in te gaan zetten. Echter is het onderzoek gebaseerd op vier denkvaardigheden van computational thinking en om een betrouwbare conclusie te kunnen trekken, moeten alle vaardigheden worden onderzocht met een grotere onderzoeksgroep.

Inleiding

Verlegenheidssituatie praktijkschool
De Howiblo is een school met ongeveer vierhonderd leerlingen. De school is katholiek, de ouders zijn hoogopgeleid en de school wil dat de kwaliteit van het onderwijs hoog is. Steeds beter onderwijs is hun doel. In de praktijk werken ze daarom met moderne leermethoden, die worden ondersteund door computers. De school wil graag handelingsgericht en hedendaags onderwijs geven. De school staat voor inventief, innovatief, inspirerend en boeiend onderwijs. De school vindt samenwerken belangrijk. Volgens de school moet de autonomie van de leerlingen bewaard blijven, omdat het leerproces belangrijk is en de leerling moet hier zelf inzicht in krijgen. Bovendien vindt de Howiblo creativiteit belangrijk, zoals ateliers. Daarnaast doet de Howiblo mee aan het concept van BAS: bouwen aan een adaptieve school. De vierdejaarsstage van de onderzoeker wordt gelopen in groep 3 met dertig leerlingen. De leerlingen zijn leergierig en er heerst een goed pedagogisch klimaat. Door het streven naar hedendaags onderwijs wil de school aandacht besteden aan de nieuwe technologische ontwikkelingen in de klas. Betrokkenen gaven aan dat er in de bovenbouw al mee was geëxperimenteerd door middel van ateliers, waarbij leerlingen zelf mochten kiezen of ze deze vaardigheid wilden leren. De leerlingen vonden dit interessant, maar hadden nog weinig kennis. De leerkracht van groep 3 gaf aan dat er in de onderbouw geen aandacht wordt gegeven aan de nieuwe technologie, zoals programmeren. Ze vroeg zich af of het invoeren op jonge leeftijd wel zin heeft.

Aanleiding en doel
Al met al wil de school hedendaags en vernieuwend onderwijs geven, maar geven hier in de praktijk geen vorm aan. Vooral het onderwerp programmeren vonden de leerkrachten interessant. Het doel in dit onderzoek is het doen van aanbevelingen hoe de Howiblo computational thinking skills in kan voeren in de bovenbouw en eventueel in de onderbouw. Daarvoor is kennis nodig over de ontwikkeling van computational thinking skills tussen de verschillende leeftijden. Door het onderzoek is er genoeg kennis om te concluderen wat voor ontwikkeling de leerlingen hebben doorgemaakt en wat de voor- en nadelen zijn bij jongere- en oudere basisschoolleerlingen. Door de uitvoering van het onderzoek zijn er mogelijkheden voor leerkrachten om het uit te voeren in de praktijk. Het doel van dit onderzoek is het verbeteren van de computational thinking skills in de groepen 3 en 8 op de Howiblo, zodat de leerlingen kennismaken met innovatief onderwijs. Hierdoor wordt er rekening gehouden met de behoefte van de school, omdat de Howiblo graag wil dat de leerlingen beter worden in de technologische vaardigheden. Het doel hiervan is dus dat de leerlingen niet alleen kennismaken met vernieuwend onderwijs, maar daar ook een ontwikkeling in doormaken. Computational thinking is een 21-eeuwse vaardigheid en die vaardigheden hebben ze nodig in de toekomst, vandaar dat deze vaardigheden in het hedendaagse onderwijs moeten worden aangeleerd (Kennisnet, 2016.). Computational thinking hoort bij innovatief onderwijs.

Hoofdvraag
Hoe beïnvloedt leeftijd van basisschoolleerlingen het bevorderen van computational thinking skills op de Howiblo?

Deelvragen praktijk
Deelvraag 1: Hoe wordt computational thinking door de leerkracht ingezet in het huidige onderwijs in groep 8?
Deelvraag 2: Hoe wordt computational thinking in het huidige onderwijs ervaren door de leerlingen van groep 8?
Deelvraag 3: Hoe wordt computational thinking door de leerkracht ingezet in het huidige onderwijs in groep 3?

Deelvraag 4: Hoe wordt computational thinking in het huidige onderwijs ervaren door de leerlingen van groep 3?
Deelvraag 5: In welke maten worden de computational thinking skills van de leerlingen in groep 8 bevorderd gedurende drie activiteiten?
Deelvraag 6: In welke maten worden de computational thinking skills van de leerlingen in groep 3 bevorderd gedurende drie activiteiten?
Deelvraag 7: Wat zijn de verschillen en overeenkomsten tussen de ontwikkeling van computational thinking skills in groep 3 en 8?

Deelvragen theorie
Wat is computational thinking?
Welke verbinding heeft computational thinking met programmeren?
Aan welke 21-eeuwse vaardigheden draagt computational thinking nog meer bij?
Welke voor- en nadelen heeft computational thinking bij jonge basisschoolleerlingen?
Welke voor- en nadelen van computational thinking bij oudere basisschoolleerlingen?

Verantwoording groepen
In het onderzoek wordt groep 3 met groep 8 vergeleken, omdat de leerkracht in groep 3 met deze vraag kwam. De leerkracht wil weten of het zin heeft om in groep 3 te starten met deze nieuwe vaardigheden, omdat ze normaal gesproken alleen bezig zijn met leren lezen en schrijven. Overigens is dit de stagegroep van de onderzoeker. Vervolgens is het interessant om te kijken naar de oudere basisschoolleerlingen. De groepen 3 en 8 worden vergeleken, omdat de leerlingen in groep 8 aan het eind van hun basisschool zitten. Hierdoor hebben zij de meeste hoeveelheid kennis en vaardigheden van alle leerlingen op de school. Dit is een interessante reden om te bekijken of de leerlingen daar een grotere groei doormaken dan in groep 3 waar ze net beginnen met het leren van de basiskennis. In het onderzoek worden dus jonge leerlingen met oudere leerlingen vergeleken. Ter vertegenwoordiging zijn groep 3 en 8 gekozen.

Theoretisch kader

In het theoretisch kader worden de definities van computational thinking en programmeren toegelicht. Daarnaast wordt de leerlijn programmeren en het belang van programmeren beschreven. Bovendien wordt er beschreven welke 21-eeuwse vaardigheden er aan programmeren gekoppeld kunnen worden. Ten slotte worden de gebruikswijze en de voor- en nadelen van programmeren vergeleken bij jonge- en oude basisschoolleerlingen. Op het einde wordt er kritisch gekeken welke bronnen er geschikt zijn voor het onderzoek.

Computational thinking

Computational thinking is het denkproces waardoor problemen en hun oplossingen effectief uitgevoerd kunnen worden door een informatie verwerkende tussenpersoon (Kennisnet, 2016.). Dit kan een mens of een computer zijn, maar dit kan ook een combinatie van beide zijn. Het zijn de creatieve vaardigheden die belangrijk zijn om problemen op te lossen door middel van digitale tools (Kennisnet, 2016.). Deze problemen zijn complex, doordat ze veel informatie en variabelen bevatten. Programmeertaal is een formele taal die wordt geschreven, zodat de computer opdrachten kan uitvoeren. Deze taal is anders dan de andere spreektalen en daarom moet iedereen dit eerst leren. Vorig jaar op de stageschool van de onderzoeker moesten de leerlingen een robot besturen, maar ze wisten niet hoe de functies van een robot heetten. Hierdoor konden de leerlingen niet aan de slag. Het gaat om het nadenken over hoe de informatie tot stand is gekomen, waardoor je computers kan benutten voor het oplossen hiervan. Computational thinking wordt ook wel gezien als de nieuwe manier om de digitalisering van de samenleving een plek te geven in de klas (Kennisnet, 2016.). Het is wetenschappelijk niet bewezen dat computational bijdraagt aan deze vaardigheden. Als er wordt gekeken naar de samenleving, dan weten we wel dat het noodzakelijk is. Tegenwoordig moet er creatief worden omgegaan met de computer als bouwsteen voor de informatiemaatschappij. In de beroepen van tegenwoordig moet je bijna altijd wel resultaten interpreteren met computerprogramma's. Leerkrachten hebben ook ervaren dat ze die programma's moeten kunnen besturen. Als we nu spelenderwijs aandacht zouden besteden aan computational thinking en programmeren, pakken de leerlingen dit later sneller op (Kennisnet, 2016.).

Overigens zijn er meer definities van dit ruime begrip. Hieronder staan er drie opgenoemd, die de meeste definities dekken. Volgens Vernieuwenderwijs zijn het vaardigheden om een beter overzicht te krijgen van complexe problemen (Onderwijs in de 21^{e} eeuw, 2016). Don Zuiderman zegt dat het een proces van een oplossing voor een open probleem (algoritmisch probleem) kan vertellen door meerdere variabelen te gebruiken, zodat een mens of computer het begrijpt (Onderwijs in de 21^{e} eeuw, 2016). Het SLO zegt dat het een samenspel is van de mens en de computer (Onderwijs in de 21^{e} eeuw, 2016).

Er zijn veel definities van computational thinking beschreven, maar ze hangen allemaal met elkaar samen. Dit komt, omdat al deze bronnen zijn geschreven in de laatste twee jaar. Het is namelijk een actueel onderwerp. De samenhang van deze bronnen is van toepassing op de onderzoeksgroep, want er wordt een probleem opgelost in samenspel met de mens en/of computer. Als er wordt afgegaan op één definitie is dit niet betrouwbaar, want het is een ruim begrip, want de leerlingen merken dat ze met veel verschillende dingen bezig zijn. Vandaar dat de verschillende definities meer begrip geven. Ook voor de leerlingen is het dan beter te begrijpen.

Programmeren

Programmeren houdt in dat leerlingen leren om te creëren in plaats van om te consumeren (Mediawijsheid, z.j.). Bij het programmeren leren de leerlingen meerdere vaardigheden, namelijk creatief en logisch denken, ruimtelijk inzicht, probleemoplossend vermogen, structureren en samenwerken (Mediawijsheid, z.j.).

Echter een andere bron hanteert een andere definitie van programmeren. Zij beweren dat programmeren het maken van een programma voor verplaatsbare objecten door machines, robots of tools is (Jeuring, Corbalan, Montfort, Es & Leeuwestein, 2016). Onderdelen hiervan zijn volgorde, keuze, herhaling, toestandsvariabele en functie. Deze bron beweert dat programmeren kan met programmeertaal, maar dat hoeft niet. Het kan ook in een tool of een game (plugged). Daarnaast bestaan er ook unplugged methodes voor programmeren, waarbij er geen gebruik gemaakt wordt van computers (Jeuring, Corbalan, Montfort, Es & Leeuwestein, 2016). Computational thinking is meer dan programmeren en digitale geletterdheid. De leerlingen oefenen om creatief en oplossingsgericht te zijn en het verdiepen van hun kennis van digitale systemen (Kennisnet, 2016.). Programmeren is hierbij wel belangrijk, want dat is de trein achter al het andere in de digitale wereld. Door te programmeren, leer je ook de vaardigheden van computational thinking (Kennisnet, 2016.). Programmeren is geen doel op zich, maar een middel. Vaak worden de termen coding en programmeren door elkaar gebruikt. Programmeurs moeten acties, die ze willen dat een computer uitvoert, in termen van een computer vertalen. Het schrijven van deze broncode heet codering (NLDIT computerkennis, z.j.). Dit kan ook programmeren worden genoemd, omdat dit een fase is van de computer. Alleen is het proces meer dan het schrijven van een code, namelijk ook het uitvoeren en herstellen van fouten (NLDIT computerkennis, z.j.). Codering is dus het maken van de code en bij programmeren kan worden verwezen naar elke fase van het proces (NLDIT computerkennis, z.j.).

Er zijn tien denkvaardigheden bij programmeren van computational thinking met het doel om problemen op zo'n manier te formuleren dat de computer het oplost. Dat zijn de volgende denkvaardigheden: probleem (her)formuleren, gegevens verzamelen, gegevens analyseren, gegevens visualiseren, probleem decompositie, abstractie, algoritmes, automatisering, simulatie en modellering en parellelization (SLO curriculum van de toekomst, 2015). Deze aanboddoelen zijn beschreven in drie fasen van complexiteit in de leerlijn van groep 1 tot en met 8. De keuzes die hierin gemaakt worden, sluiten aan bij de verschillen van de leerlingen. Voor computational thinking zijn een paar leerhoudingen nodig van leerlingen, namelijk dat ze om kunnen gaan met complexiteit, doorzettingsvermogen bij lastige, open problemen en dat ze kunnen communiceren en samenwerken met anderen om samen hun doel te bereiken (SLO curriculum van de toekomst, 2015). De andere bron wil deze begrippen niet kopiëren, omdat ze hun eigen tien begrippen willen gebruiken (Kennisnet, z.j.). Deze keuze is gemaakt, omdat ze de leerlijn praktisch uitvoerbaar willen houden voor alle leerkrachten. Daarom willen ze begrippen gebruiken die herkenbaar en eenvoudig zijn. Ze hebben een eigen leerlijn programmeren opgesteld, maar deze lijkt op SLO. Deze begrippen zijn de kennis en ervaring voor het programmeren. Bovendien staan de begrippen in de volgende volgorde, waardoor ze in complexiteit toenemen: algoritmes, decompositie, patronen, herhaling, fouten, voorwaarden, abstractie, functie, variabele en representatie (Kennisnet, z.j.). Deze begrippen zijn onverdeeld in onder-, midden- en bovenbouw, maar een school maakt hier zijn eigen keuzes in door het niveau van de leerlingen. Op de vorige stageschool van de onderzoeker waren ze in de bovenbouw nog bezig met de doelen van de middenbouw, omdat de leerlingen nog te weinig kennis hadden. Dit is gedaan voor een leerlijn die in de praktijk eenvoudig uit te voeren is door middel van eenvoudigere begrippen. Binnen deze leerlijn kan de school zelf de invulling bepalen. Elk begrip vervuld doelen van de SLO en daardoor is er steeds een koppeling gemaakt tussen de begrippen van Kennisnet en de doelen van het SLO (Kennisnet, z.j.). Deze leerlijn geeft een praktische uitwerking voor een onderdeel van toekomstgericht onderwijs.

Kennisnet leerlijn programmeren	**SLO tien denkvaardigheden computational thinking**
Algoritmes	Probleem (her)formuleren
Decompositie	Gegevens verzamelen

Patronen	Gegevens analyseren
Herhaling	Gegevens visualiseren
Fouten	Probleem decompositie
Voorwaarden	Abstractie
Abstractie	Algoritmes
Functie	Automatisering
Variabele	Simulatie
Representatie	Modellering en parellelization

Kennisnet, z.j. *SLO curriculum van de toekomst, 2015*

De hier bovengenoemde leerlijn programmeren, bestaat uit begrippen die nodig zijn om te kunnen programmeren. Erik Woning beweert dat deze begrippen unplugged kunnen worden uitgevoerd in activiteiten op school om eenvoudig te kunnen integreren in de praktijk (2016). Daarentegen kan het programmeren ook tastbaar worden aangeboden. Dit wordt niet als apart vak aangeboden, maar verwerkt in het huidige curriculum. De leerlijn biedt een brede basis om stap voor stap te beginnen. Erik Woning beweert dat na de eenvoudige opdrachten, de leerlingen meer ruimte voor creativiteit krijgen en dan verandert hun rol van consument naar producent door de technologie (2016). Deze bron beweert hetzelfde als de eerste bron.

Ook de theorieën over programmeren lopen uiteen. De ene bron focust zich vooral op het creëren (Mediawijsheid, z.j.) en de andere bron richt zich meer op een programma maken met behulp van tools (Jeuring, Corbalan, Montfort, Es & Leeuwestein, 2016). De laatste bron is het meest van toepassing op de onderzoeksgroep, omdat de leerlingen door middel van een tool leren om te programmeren. De leerlingen ervaren dat ze door middel van voorwerpen stapjes in een programma kunnen uitvoeren. Natuurlijk zijn de leerlingen ook bezig met het creëren (beschreven in de eerdere bron), omdat ze nieuwe dingen maken. Echter kan creëren ook op andere vlakken, zoals bij kunst. Dit komt niet in het onderzoek voor en dit zien de leerlingen niet als programmeren. Daarnaast wordt de leerlijn van programmeren (Kennisnet z.j.) gekoppeld aan de doelen voor computational thinking bij het SLO (SLO curriculum van de toekomst, 2015). SLO beschrijft tien denkvaardigheden die nodig zijn. Deze vaardigheden zijn opgebouwd in drie fasen van complexiteit van groep 1 tot en met 8. Kennisnet geeft daar zijn eigen draai aan, waardoor het concreet is uit te voeren in de praktijk. Het is makkelijker in de praktijk uit te voeren, omdat de begrippen in volgorde van complexiteit zijn gerangschikt en zijn uitgewerkt per bouw. Hierdoor sluiten de activiteiten beter aan bij het niveau van de leerlingen. Dus deze twee bronnen overlappen elkaar, maar Kennisnet zorgt ervoor dat de leerlijn praktisch haalbaar is in de onderzoekspraktijk. In het onderzoek wordt er gebruik gemaakt van beide bronnen, omdat Kennisnet een praktische uitwerking geeft van de verschillende begrippen en SLO geeft de moeilijkheid aan van de verschillende denkvaardigheden. Deze kennis is beide nodig om te kunnen meten of de leerlingen zich ontwikkelen.

Waarom programmeren?

Programmeren wordt in de toekomst steeds belangrijker en daarom is het essentieel dat leerlingen de vaardigheden opdoen die ze helpen om aan de 'achterkant' van een apparaat te kunnen kijken. Michiel Lucassen beweert dat een voordeel van programmeren is dat er veel baankansen zijn in deze sector (2017). Als de leerlingen jong de mogelijkheden ontdekken, komen er meer programmeurs. Ten tweede beweert Michiel Lucassen dat programmeren niet eenvoudig is, maar het wel direct feedback geeft (2017). Hierdoor leren de leerlingen om fouten te maken. Bovendien leren de leerlingen beter denken volgens Michiel Lucassen, omdat ze complexe problemen moeten oplossen (2017). Hierbij moeten ze kritisch denken. Daarnaast beweert Michiel Lucassen dat de leerlingen leren om zelf iets te maken, waardoor ze hun creativiteit inzetten (2017). Ten slotte is Michiel

Lucassen van mening dat de leerlingen leren omgaan met algoritmes, waardoor zij begrijpen hoe bepaald nieuws tot je komt (2017). Door programmeren snap je beter hoe de wereld werkt.

ICT is nodig in onze maatschappij en het wordt steeds belangrijker om met computers om te gaan (PO-Raad, 2017). We bereiden de leerlingen goed voor op de toekomst als we leren hoe ze hier mee om moeten gaan (PO-Raad, 2017). De PO-Raad vindt het belangrijk dat de leerlingen digitaal geletterd worden (PO-Raad, 2017). Programmeren is een middel om tegemoet te komen aan computational thinking. De school kan zelf keuzes maken op welke wijze zij aandacht besteden aan digitale geletterdheid (PO-Raad, 2017).

21-Eewse vaardigheden

Computational thinking is een onderdeel van digitale geletterdheid en hieronder vallen de volgende 21-eeuwse vaardigheden: ICT-vaardigheden, informatievaardigheden, mediawijsheid en computational thinking (Kennisnet, 2016.). Computational thinking is dus zelf al een 21-eeuwse vaardigheid. Bij 21 -eeuwse vaardigheden gaat het erom dat ze vaardigheden opdoen, die ze in hun toekomst nodig hebben. Deze vaardigheden moeten dus nu aangeleerd worden (Kennisnet, 2016.). Creatief denken, probleemoplossend denken en digitale geletterdheid worden nog weinig in de klas benut, maar door programmeren leren de leerlingen ook factoren van het moderne computergebruik (Kennisnet, 2016.).

Kennisnet, 2016

Mediawijsheid staat nu voor alles wat te maken heeft met media, ICT en hoe je hiermee moet werken, dus programmeren valt hieronder (Pardoen, z.j.). Mediawijsheid is een onderdeel van de digitale geletterdheid. Ook deze bron beweert dat digitale geletterdheid de paraplu is voor de vier 21-eeuwse vaardigheden: ICT-vaardigheden, informatievaardigheden, mediawijsheid en computational thinking skills (Pardoen, z.j.). Dit zijn dezelfde als Kennisnet. Onderling hangen deze vaardigheden samen, want de leerlingen ontwikkelen ze tegelijk (Pardoen, z.j.).

Digitale geletterdheid
1. ICT-basisvaardigheden
2. *Computational thinking*
3. Informatievaardigheden
4. Mediawijsheid

Pardoen, z.j.

Ypenberg Moonen beweert dat computational thinking gaat om het begrijpen van de digitale wereld, maar dat kinderen ook andere vaardigheden opdoen tijdens het programmeren (2017). Volgens Ypenberg Moonen zijn dat de volgende vaardigheden: het leren logisch nadenken, creatief denken, ruimtelijk inzicht, probleemoplossend vermogen creëren en samenwerken (2017). Creatief denken, probleemoplossend vermogen en samenwerken zijn ook 21-eeuwse vaardigheden, waar leerlingen in de toekomst profijt van hebben.

De ene bron beweert dat computational thinking één van de vier 21-eeuwse vaardigheden waren voor digitale geletterdheid (Kennisnet, 2016), maar Ypenberg Moonen (2017) beweert dat creatief denken, oplossingsgericht denken en samenwerken er ook bij horen. Volgens de mening van de onderzoeker behoort computational thinking tot de digitale geletterdheid, maar er worden

tegelijkertijd ook andere 21-eeuwse vaardigheden aangesproken. Deze vaardigheden zijn belangrijk voor de toekomst, omdat ze andere beroepen gaan doen dan wij nu gewend zijn.

Computational thinking bij jonge basisschoolleerlingen
In Engeland leren de leerlingen jong programmeren, zodat ze de grondbeginselen van alle informatievaardigheden en mediawijsheid kunnen opslaan (Maes, 2015). Bij de jonge leerlingen is dit vaak niet achter een computer, maar unplugged (zoals rollenspellen) (Maes, 2015). Hierdoor blijven ze spelenderwijs leren. Er zijn vier vormen van programmeren: unplugged, blokken, visueel en tekstueel. Er zijn vier uitvoeringen door programma's: robots, grafisch, tekstueel of unplugged (Jeuring, Corbalan, Es & Leeuwestein, 2016). Blokken en robots worden gebruikt in de onderbouw. Leerlingen uit de onderbouw leren de programmeerconcepten snel door de blokken (Jeuring, Corbalan, Es & Leeuwestein, 2016). Daarnaast hebben de leerlingen veel plezier. Op de eerste stageschool van de onderzoeker waren de kleuters aan het werk met de Bee-Bots en de leerlingen waren razend enthousiast. Britse leerlingen doen dit vanaf hun 5^{e} jaar, vandaar dat ze al twee computertalen kennen in groep 8 (Maes, 2015). De leerlingen hoeven niet allemaal programmeur te worden, maar door computational thinking leren kinderen al jong om hun creatieve, oplossingsgerichte vermogen en kennis van de digitale wereld tot zich te nemen (Maes, 2015). Hoe later leerlingen in aanraking komen met computers, hoe meer de leerlingen gelijk zeggen dat het niks voor hen is of dat ze het niet kunnen (Visser, 2017). Meisjes hebben op jonge leeftijd (al rond acht jaar) vooroordelen over hun eigen prestaties (Visser, 2017). Er zit een meisje in groep 3, die bij alle nieuwe dingen zegt dat ze het niet kan en dan doet ze het niet of met minder motivatie, zoals het werken met de computer. Volgens deze bron voorkom je dit door vroeg te beginnen. Echter is niet iedereen het hier mee eens, want we gebruiken ook allemaal een stoel en die hoeven we ook niet allemaal te leren maken. De leerlingen moeten dus wel een computer kunnen gebruiken, maar we hoeven ze in het begin van het primair onderwijs niet aan te leren hoe ze dat moeten maken (Maes, 2015).

Computational thinking bij oudere basisschoolleerlingen
In Engeland zijn de leerlingen vroeg begonnen met het leren van deze vaardigheden, waardoor de oudere leerlingen deze kennis kunnen uitbreiden (Maes, 2015). Oudere leerlingen hoeven ook niet alles met een beeldscherm te doen, ook de complexere vaardigheden kunnen op papier. In de bovenbouw wordt er ook bijvoorbeeld met tekstuele programma's gewerkt (kan unplugged). Dit kan alleen in de bovenbouw, omdat ze goed moeten lezen en schrijven (verdieping). In de bovenbouw maken ze kennis met symbolen en de meest gebruikte vorm is visueel programmeren. Als de leerlingen in de bovenbouw eerst tekstueel en vervolgens visueel leren programmeren, leidt dit tot betere resultaten is gebleken uit een onderzoek (Jeuring, Corbalan, Es & Leeuwestein, 2016). Dit komt, omdat deze vaardigheden ook worden toegepast in andere situaties. Op de vorige stageschool van de onderzoeker gebruikten de leerlingen dit bijvoorbeeld ook bij begrijpend lezen. Het is belangrijk dat ook oudere leerlingen leren programmeren, want het is belangrijk voor de arbeidsmarkt, maar ze moeten ook mee kunnen praten over het nieuws (burgerschap) (Visser, 2017). Het nieuws gaat veel over software. Laatst was er veel op het nieuws over hacken en daar moet je in de bovenbouw over kunnen praten. Daarnaast is creativiteit een belangrijk aspect (Visser, 2017). Bovendien moet het voor alle leerlingen aangeboden worden, dus in de bovenbouw en het voortgezet onderwijs. Als het verplicht wordt in het voortgezet onderwijs, kan het minder in het primair onderwijs (Visser, 2017).

Sommige theorieën vinden dat je vroeg moet beginnen met het opleiden in computational thinking voor betere vaardigheden en het kennismaken met het vak, zodat de leerlingen niet zeggen dat ze het niet kunnen. De leerlingen die vroeg beginnen vinden het meestal leuk. Deze bron richt zich

echter vooral op meisjes. Hierdoor kan er geen uitspraak worden gedaan over deze ontwikkeling bij jongens en dat maakt dit onderzoek minder bruikbaar. Bovendien wordt er beweert dat je het in het primair onderwijs minder hoeft aan te bieden, als het in het voortgezet onderwijs wordt aangeboden (Visser, 2017). Dit is een andere mening ten opzichte van de overige bronnen, waar er wordt beschreven dat het altijd zin heeft om jong te beginnen. Anderen vinden dat leren programmeren helemaal niet hoeft, omdat niet iedereen dit hoeft te leren (Maes, 2015). Leerlingen vinden het echter wel fijn om een keuze te hebben of ze er later wat mee willen doen. Daarom moet het volgens de meeste leerlingen wel aangeboden worden en is deze bron bij dit gedeelte minder bruikbaar in de onderzoekspraktijk. Overigens is dit een groot verschil met de overige bronnen, maar deze bron is niet bewezen met een echt onderzoek. Dit is een mening van de auteur en dat maakt het minder bruikbaar. Bij jonge kinderen worden programmeerconcepten snel geoefend door blokken en robots en bij oudere leerlingen eerst tekstueel en vervolgens visueel (Jeuring, Corbalan, Es & Leeuwestein, 2016*).* Dit is wel onderzocht, waardoor deze informatie betrouwbaarder is. Daarnaast is deze theorie te benutten in de praktijk, omdat het onderzoek met jonge leerlingen en oudere leerlingen wordt uitgevoerd.

Conclusie

Al met al is programmeren een middel om bij te dragen aan computational thinking. Er is een leerlijn bestaande uit tien denkvaardigheden van computational thinking, waardoor de leerlingen bepaalde reeksen formuleren, zodat de computer het oplost. Programmeren wordt in de toekomst steeds belangrijker, waardoor de leerlingen daar goed op moeten worden voorbereid. Computational thinking is zelf al een 21-eeuwse vaardigheid, maar het spreekt ook nog meer 21-eeuwse vaardigheden aan. Jonge basisschoolleerlingen leren vooral unplugged. Het voordeel wanneer ze het vroeg leren, is dat de leerlingen (vooral meisjes) minder onzeker zullen zijn op dit gebied. De oudere basisschoolleerlingen programmeren eerst tekstueel en dan visueel voor een beter resultaat. De oudere leerlingen verdiepen zich in de leerstof, waardoor ze zich beter kunnen redden in de huidige maatschappij.

Methode

Het onderzoek is een ontwerpgericht onderzoek, omdat de leerlingen in meerdere lessituaties leren programmeren. Zelfgecreëerde lessen worden gegeven aan de hand van de theorie en de ervaringen die zijn opgedaan. Dit gebeurt door middel van drie interventies, waarna de ontwikkeling van de leerlingen wordt gemeten.

Deelvraag 1: Hoe wordt computational thinking door de leerkracht ingezet in het huidige onderwijs in groep 8?

Participanten

De groepsleerkracht van groep 8 wordt geïnterviewd, omdat zij het huidige onderwijs in groep 8 het best kent. Daarnaast heeft zij de meeste kennis over de leerlingen, waardoor ze de leerlingen het best kan inschatten.

Onderzoeksinstrumenten

Semigestructureerd interview groepsleerkracht

In dit interview liggen een aantal open hoofdvragen vast, zodat deze onderwerpen aan bod komen (Harinck, 2013, p. 138-139). Tijdens het gesprek is er ruimte om door te vragen, waardoor er een vollediger beeld ontstaat. Tijdens het interview worden er notities gemaakt, waardoor de verwerking is begonnen, omdat de essentiële informatie is genoteerd. Het is belangrijk dat deze leerkracht wordt geïnterviewd, omdat zij de meeste kennis heeft over het huidige onderwijs dat wordt gegeven in de klas, waar het onderzoek wordt uitgevoerd. Zie bijlage 1 voor dit onderzoeksinstrument.

Procedure

Aan het begin van het onderzoek wordt het interview gehouden. Dit vindt plaats in december. Dit is een onderdeel van de nulmeting.

Analyse

Semigestructureerd interview groepsleerkracht

Het interview wordt eerst verwerkt in een matrix. Deze vorm past goed bij interviewmateriaal, vooral bij semigestructureerd interviews, omdat er veel kwalitatieve gegevens vrijkomen (Harinck, 2013, p.181). Vervolgens wordt de essentiële informatie beschreven in een verslag, waardoor beide beginsituaties helder zijn. Hierbij wordt gebruik gemaakt van de vignettebenadering, waardoor alleen dezelfde antwoorden worden beschreven, die meerdere keren worden verteld.

Deelvraag 2: Hoe wordt computational thinking in het huidige onderwijs ervaren door de leerlingen van groep 8?

Participanten

De vragenlijst wordt bij zes leerlingen van groep 8 afgenomen. Deze leerlingen zijn willekeurig gekozen, zodat het een zo betrouwbaar mogelijk onderzoek is. Bij grote aantallen is een steekproef verstandig, omdat dat representatief is voor de populatie (Harinck, 2013, p. 119). Tussen deze leerlingen zitten verschillen, zowel op het gebied van leerstrategieën als gedrag. Hierdoor wordt er tegemoetgekomen aan de verschillen van de leerlingen. Differentiatie is belangrijk op de Howiblo.

Onderzoeksinstrumenten

Gestructureerde vragenlijst leerlingen

Bij zes leerlingen uit groep 8 wordt de vragenlijst afgenomen, omdat het onderzoek bij hen wordt uitgevoerd. Hierdoor ontstaat er een beter beeld van de huidige situatie en de ontwikkeling van de leerlingen. De leerlingen weten zelf het best wat ze hebben geleerd. Het is een gestructureerd instrument, omdat alle leerlingen zich mogen uitspreken over de vraag (Harinck, 2013, p.127). Dit

instrument leent zich voor meningen en ervaringen. Een vragenlijst heeft vaak open en gesloten vragen. Open vragen zijn belangrijk, zodat er voldoende informatie wordt verkregen (Harinck, 2013, p.127). Het is een korte vragenlijst, zodat de leerlingen zich kunnen blijven concentreren. Zie bijlagen 2 en 3 voor deze onderzoeksinstrumenten.

Procedure
Aan het begin van het onderzoek worden de vragenlijsten afgenomen. Dit vindt plaats in december. Dit is een onderdeel van de nulmeting

Analyse
Gestructureerde vragenlijst leerlingen
De kwalitatieve gegevens worden verwerkt in een verslag met de essentiële aspecten om deze deelvraag te beantwoorden. Hierbij wordt gebruik gemaakt van de vignettebenadering, waardoor de terugkerende antwoorden worden benadrukt. Hierdoor wordt de beginsituatie helder. De kwantitatieve gegevens (vraag 7 en 8) worden gebruikt voor een andere deelvraag.

Deelvraag 3: Hoe wordt computational thinking door de leerkracht ingezet in het huidige onderwijs in groep 3?
De participanten, onderzoeksinstrumenten, procedure en de analyse zijn hetzelfde als bij deelvraag 1, alleen wordt het nu uitgevoerd met de groepsleerkracht van groep 3. Zie bijlage 1 voor dit onderzoeksinstrument.

Deelvraag 4: Hoe wordt computational thinking in het huidige onderwijs ervaren door de leerlingen van groep 3?
De participanten, onderzoeksinstrumenten, procedure en de analyse zijn hetzelfde als bij deelvraag 2, alleen wordt het nu uitgevoerd met zes leerlingen van groep 3. Er wordt voor deze leerlingen geschreven als ze zelf nog niet kunnen schrijven. Zie bijlagen 2 en 3 voor deze onderzoeksinstrumenten.

Deelvraag 5: In welke maten worden de computational thinking skills van de leerlingen in groep 8 bevorderd gedurende drie activiteiten?
Participanten
De drie interventies worden met de hier bovengenoemde zes leerlingen van groep 8 uitgevoerd, omdat zij deel uitmaken van het onderzoek en hiervoor willekeurig gekozen zijn.

Onderzoeksinstrumenten
De leerlingen oefenen tijdens de interventies in tweetallen, zodat ze kunnen samenwerken. In het onderzoek worden er instrumenten voor de interventies gekozen met open opdrachten, zodat de leerlingen hun creativiteit benutten (21 -eeuwse vaardigheid). Bovendien worden ze eigenaar van hun project. Bij het begin van het onderzoek gaan de leerlingen individueel aan de slag met Scratch. Er wordt toegelicht hoe de verschillende scripts werken (minimale uitleg) en vervolgens krijgen de leerlingen vrijheid om de opdracht zelf uit te voeren. De interventies staan beschreven in bijlage 6.
Gestructureerde vragenlijst leerlingen
Dezelfde zes leerlingen van groep 8 vullen twee vragenlijsten in. Een vragenlijst voor de drie interventies en een vragenlijst na de drie interventies. Voor deze deelvraag gaat het om de laatste twee vragen. De leerlingen scoren hun ontwikkeling met getallen van 1 tot en met 5. De leerlingen weten zelf het best wat voor ontwikkeling ze hebben doorgemaakt. Bij de tweede vragenlijst lichten ze ook toe wat ze hebben geleerd. Zie bijlagen 2 en 3 voor deze onderzoeksinstrumenten.
Beoordeling aantal scripts
Tijdens de nulmeting en de effectmeting worden dezelfde zes leerlingen van groep 8 geobserveerd.

Zij worden individueel geobserveerd. Hierbij zijn er duidelijke aandachtspunten (Harinck, 2013, p.151), namelijk de vaardigheden die worden geoefend voor het programmeren. Bij deze observatie wordt Scratch gebruikt, waarin er complexe vaardigheden worden gebruikt. Dit instrument is afgestemd op de begrippen van Kennisnet en het SLO en het niveau van de leerlingen. Deze aantallen worden geturfd en beoordeeld met een matig, voldoende of goed. Zie bijlage 4 voor dit onderzoeksinstrument.

Beoordeling niveau denkvaardigheid (fases van de leerlijn)

Bij deze observatie worden ook dezelfde zes leerlingen van groep 8 geobserveerd voor de nul- en effectmeting. Bij de nul- en effectmeting worden dezelfde onderzoeksinstrumenten gebruikt voor een betrouwbaar resultaat. Deze observatie vindt plaats na een bepaalde tijdsduur, omdat de observator gestructureerd kijkt naar de afgelopen periode via beoordelingsschalen (Harinck, 2013, p.153). De beoordelingsschalen zijn de fases van complexiteit in de leerlijn van computational thinking (SLO curriculum van de toekomst, 2015). Deze begrippen zijn gecombineerd met de begrippen van de leerlijn programmeren (Kennisnet, z.j.). Zie bijlage 5 voor dit onderzoeksinstrument.

Observatie ontwikkeling complexiteit in de vier denkvaardigheden en samenwerking

Tijdens het uitvoeren van de drie interventies worden de activiteiten van de zes leerlingen objectief geobserveerd, zodat de ontwikkeling in beeld wordt gebracht. Deze interventies staan uitgewerkt in bijlage 6. Hierbij wordt gebruik gemaakt van de vignettebenadering. Dit wordt bij analyse toegelicht.

<u>Procedure</u>

De gegevens worden op meerdere momenten verzameld. Tijdens de nulmeting in december worden de vaardigheden van de leerlingen gemeten. Het doel is dat de leerlingen door drie interventies in de maanden januari en februari deze vaardigheden beter gaan ontwikkelen. Op een onderzoeksdag worden twee interventies uitgevoerd, namelijk drie kwartier in groep 3 en drie kwartier in groep 8. Dit gebeurt drie weken achter elkaar. Het observeren gebeurt in natuurlijke situaties, omdat het deel uitmaakt van de activiteiten (Harinck, 2013, p. 121). De ontwikkeling wordt gemeten bij de effectmeting begin maart. Daarnaast worden er gegevens verzamelt door lesmaterialen te bekijken, want de leerlingen gaan computational thinking oefenen op verschillende manieren. Het doel blijft hetzelfde, maar er worden verschillende instrumenten gebruikt. Als ze steeds zouden werken met hetzelfde lesmateriaal, voeren ze een trucje uit met dat instrument. De leerlingen oefenen dan niet specifiek het programmeren. Dit onderdeel is helemaal nieuw op de Howiblo, vandaar dat de kennis van de onderzoeker wordt verhoogd door middel van een cursus op de Marnix Academie, theorie en interventies met de leerlingen.

<u>Analyse</u>

Gestructureerde vragenlijst leerlingen

De kwantitatieve gegevens van de vragenlijst worden verwerkt in een frequentieverdeling, omdat dit zorgt voor een overzichtelijke weergave (Harinck, 2013, p.173). De onderzoeksgroep is kleiner dan tien personen. Hierbij is het handig om de percentages weg te laten, want dit kan misleiden (Harinck, 2013, p.173). Hierbij wordt de verwerking van een cirkeldiagram gekozen, omdat een grafische vorm de ontwikkeling zichtbaar maakt en dat eenvoudig kan worden vergeleken met de effectmeting. Uiteindelijk wordt overzichtelijk weergegeven hoe leuk en hoe goed de leerlingen programmeren vinden.

Beoordeling aantal scripts

Alle scores worden een voor een gescoord (Harinck, 2013, p.181). Hiervan wordt een frequentietabel gemaakt, zodat de scores overzichtelijk worden weergegeven. Hierbij wordt kwalitatief materiaal op een systematische wijze verwerkt en gecodeerd (Harinck, 2013, p.181). De leerlingen worden

beoordeeld op het aantal scripts wat ze gebruiken. Dit zijn de functies die de leerlingen gebruiken tijdens het programmeren. Voor groep 8 wordt er gekeken naar de volgende scripts: beweging, uiterlijk, geluid, gebeurtenissen en besturen. Deze worden het meest gebruikt. Pen, data, waarnemen en functies zijn niet opgenomen in het onderzoeksinstrument, omdat die scripts beide metingen niet worden gebruikt. Als de leerlingen minder dan vier scripts gebruiken, is dit matig. Bij vier scripts, krijgen ze een voldoende en bij vijf (of meer) scripts krijgen ze een goede beoordeling. Uiteindelijk wordt dit weergegeven in een cirkeldiagram, waardoor het een helder beeld geeft en de nul- en effectmeting eenvoudig met elkaar kunnen worden vergeleken.

Beoordeling niveau denkvaardigheid (fases van de leerlijn)

Hierbij wordt kwalitatief materiaal op een systematische wijze verwerkt en gecodeerd (Harinck, 2013, p.181). Er wordt gekeken in welke fase de leerlingen zitten op de leerlijn bij vier denkvaardigheden. De eerste denkvaardigheid is het herformuleren van een probleem. Fase 0 is als ze geen probleem kunnen oplossen met de computer. Fase 1 is het verkennen van de mogelijkheden om problemen op te lossen met een computer. Fase 2 is het verkennen of een probleem opgelost kan worden met een computer en bij fase 3 moeten ze de vorige fasen beheersen en op hun genomen beslissingen kunnen reflecteren. De tweede denkvaardigheid is herhaling/automatisering. Bij fase 0 maken ze geen gebruik van herhaling. Bij fase 1 beseffen ze dat een computer een taak eindeloos kan laten herhalen. Bij fase 2 vergelijken ze de resultaten met en zonder apparaten en bij fase 3 onderzoeken ze op welke manier een computerprogramma kan ondersteunen bij een terugkerende taak of handeling. De derde denkvaardigheid is algoritmes en procedures. Bij fase 0 kunnen de leerlingen geen reeks in stappen zetten. Bij fase 1 kunnen ze een taak uitvoeren door dit stap voor stap in een logische volgorde te plaatsen en mogelijke fouten eruit te halen. Bij fase 2 maken ze kennis met de betekenis van een algoritme en kunnen ze een algoritme door middel van codes en symbolen representeren. Bij fase 3 kunnen de leerlingen hun eigen set instructies ontdekken door logisch te redeneren en ze kunnen een eenvoudige computercode schrijven. De laatste denkvaardigheid is probleem decompositie. Bij fase 0 kunnen ze geen taak opdelen in stappen en corrigeren ze hun eigen fouten niet. Bij fase 1 kunnen ze een eenvoudige taak opdelen in deeltaken. Bij fase 2 kunnen ze een proces opdelen in verschillende stappen en waar mogelijk elke stap weer in deelstappen. Bij fase 3 kunnen ze een grotere en meer complexere taak in deeltaken verdelen. De scores worden namelijk geturfd in een frequentietabel, waardoor er een staafdiagram van kan worden gemaakt om de voortgang van de leerlingen in beeld te brengen. Deze nul- en effectmeting kunnen overzichtelijk met elkaar worden vergeleken.

Observatie ontwikkeling complexiteit in de vier denkvaardigheden en samenwerking

De kwalitatieve gegevens worden verwerkt in een verslag. Hierbij wordt gebruik gemaakt van de vignettebenadering, waarbij verschillende kenmerken van programmeren steeds worden benoemd (Harinck, 2013, p.180). Hierbij wordt gelet op de volgende vaardigheden die de leerlingen steeds opdoen tijdens de verschillende vormen van programmeren: in een logische volgorde programmeren, complexiteit, herhalen en fouten eruit halen. Daarbij wordt ook gekeken naar een gebeurtenis die steeds terugkeert en die wordt beschreven, namelijk samenwerken. In een verslag wordt per interventie beschreven hoe de leerlingen deze begrippen hebben gebruikt.

Deelvraag 6: In welke maten worden de computational thinking skills van de leerlingen in groep 3 bevorderd gedurende drie activiteiten?

De participanten, onderzoeksinstrumenten, procedure en de analyse zijn hetzelfde als deelvraag 5, alleen wordt het nu uitgevoerd met zes leerlingen van groep 3. Het onderzoeksinstrument verschilt wel met deelvraag 5. De onderzoeksinstrumenten voor de beoordeling in aantal scripts en het niveau in denkvaardigheid zijn verschillend in groep 3 en 8, omdat de leerlingen in groep 3 met Scratch junior werken. Dit zijn andere en minder complexere vaardigheden dan Scratch voor groep 8. Daar

maken de leerlingen meer en complexere keuzes. In groep 3 wordt er gekeken naar de volgende scripts: gebeurtenis, beweging, teksten en uiterlijk, audio, herhalingen en stoppen. De andere scripts binnen deze app zijn geen enkele meting benut. Daarom zijn deze scripts niet opgenomen in het onderzoeksinstrument. De onderzoeksinstrumenten zijn afgestemd op de begrippen van kennisnet en het SLO en het niveau van de leerlingen. Deze aantallen worden geturfd en beoordeeld met een matig, voldoende of goed. Zie bijlagen 2 tot en met 5 voor de onderzoeksinstrumenten.

Deelvraag 7: Wat zijn de verschillen en overeenkomsten tussen de ontwikkeling van computational thinking skills in groep 3 en 8?
De participanten, onderzoeksinstrumenten en de procedure is hetzelfde als bij deelvraag 5 en 6. Echter worden bij de analyse, de resultaten van de onderzoeksinstrumenten van deelvraag 5 en 6, met elkaar vergeleken en dit wordt beschreven in een verslag. Hierbij wordt gebruikt gemaakt van de vignettebenadering. Deze benadering is bij deelvraag 5 uitgelegd. Echter is er één verschil. Om het onderzoeksinstrument van het aantal scripts goed te kunnen vergelijken, worden de concrete aantallen in een frequentietabel weergegeven. Om het onderzoekinstrument voor de niveaus in denkvaardigheden goed te vergelijken wordt een puntensysteem gehanteerd. Bij fase nul krijgen de leerlingen nul punten, bij fase één krijgen de leerlingen één punt, bij fase twee krijgen de leerlingen twee punten en bij fase drie krijgen de leerlingen drie punten. Deze punten worden per denkvaardigheid opgeteld, zodat de groei tussen de nul-en effectmeting duidelijk wordt en tussen de twee groepen kan worden vergeleken in een verslag.

Tijdens de teamvergadering wordt de uitkomst van het onderzoek gepresenteerd. Dit wordt gedaan aan de hand van een PowerPoint met de kwalitatieve en kwantitatieve gegevens uitgewerkt in grafische diagrammen. De presentatie staat in bijlage 7.

Keuze verantwoord op basis van de verlegenheidssituatie
Hierboven is te lezen dat de keuzes van de onderzoeker bewust zijn genomen aan de hand van de verkenning van de verlegenheidsituatie. Echter is het belangrijk om sommige bevindingen te benadrukken. In de verkenning is er kennis vernomen dat de leerlingen weinig ervaring hadden met plugged en unplugged programmeren en geen kennis hadden van de benodigde vaardigheden. Vandaar dat beide vormen in de interventies aan bod komen en er willekeurig vier denkvaardigheden zijn gekozen, die eenvoudig te beoordelen en uit te voeren zijn. Bovendien ligt het probleem ook bij de leerkrachten, omdat ze zelf weinig kennis hebben. Daarom krijgen ze uitleg van de onderzoeker over wat programmeren en computational thinking inhoudt. Bovendien zijn de vragenlijsten aangepast op de leeftijd van de leerlingen. De vragenlijsten zijn kort gehouden, zodat de jonge leerlingen van groep 3 zich ook kunnen concentreren. Bovendien is bij de verkenning waargenomen dat er veel verschillende leerlingen zijn, waardoor er ook een steekproef is genomen van de leerlingen. Dit geeft een representatief beeld voor de school.

Resultaten

De resultaten van de nulmeting, de interventies en de effectmeting worden hieronder objectief beschreven. Deze metingen zijn uitgevoerd op basis van de theorie en de gemaakte onderzoeksinstrumenten.

Deelvraag 1: Hoe wordt computational thinking door de leerkracht ingezet in het huidige onderwijs in groep 8?

Semigestructureerd interview groepsleerkracht

Uit het interview met de leerkracht komen de volgende resultaten (bijlage 8): Deze leerkracht had programmeren niet eerder toegepast in haar huidige klas, omdat ze er nog niet aan toe is gekomen en ze het zelf niet kan. Ze gaf aan dat ze eerst zelf een cursus moet volgen. Wel hadden sommige leerlingen het bij de ateliers gekozen en hebben ze het twee keer gedaan bij een andere leerkracht. Deze leerkracht wilt met programmeren beginnen in de weektaak, zodat je de leerlingen kan begeleiden. Ze zou het nooit klassikaal aanbieden. Ze vindt programmeren belangrijk, zodat ze met de tijd mee gaat en de leerlingen het later nodig hebben. De beginsituatie in deze klas ligt ver uit elkaar, omdat een aantal leerlingen goed zijn met de computer en met tips snel leren. Anderen hebben meer uitleg nodig en hebben er geen inzicht in. Na de drie interventies zullen de leerlingen weten wat programmeren inhoudt. Ze leren het beter, als ze er meer lessen in krijgen. Uiteindelijk zullen ze wel beter worden, maar sommige hebben hier veel hulp bij nodig en kunnen dit niet alleen. Ze weet zelf het verschil niet tussen programmeren en computational thinking.

Deelvraag 2: Hoe wordt computational thinking in het huidige onderwijs ervaren door de leerlingen van groep 8?

Gestructureerde vragenlijst leerlingen

Alle leerlingen hebben nog nooit geprogrammeerd op school. Dit zouden ze wel graag willen doen, omdat ze het leuk vinden. Vijf van de zes leerlingen heeft thuis ook nog nooit geprogrammeerd. De andere leerling had een robot gekregen. De helft van de leerlingen beweerden dat programmeren iets met de computer is en de andere helft beweerden dat programmeren betekent dat je dingen laat bewegen of praten. De meeste leerlingen gaven aan dat ze het belangrijk vonden, omdat je er veel van leert en het later kan gebruiken. Een paar andere leerlingen zeiden dat je leert hoe je met computers om moet gaan en dat vinden ze tegenwoordig belangrijk. Een voorbeeld van een vragenlijst staat in bijlage 9.

Deelvraag 3: Hoe wordt computational thinking door de leerkracht ingezet in het huidige onderwijs in groep 3?

Semigestructureerd interview groepsleerkracht

Uit het interview met de leerkracht komen de volgende resultaten (bijlage 8): Deze leerkracht heeft programmeren nog niet toegepast in haar huidige klas, omdat ze zich er thuis nog niet in thuis voelt. Ze zou opdrachten willen geven, waar de leerlingen zelf mee aan de slag kunnen. Dit kan in de weektaak of gekoppeld met andere vakken. Ze vindt het wel belangrijk dat het wordt aangeboden, want de leerlingen leren stap voor stap te denken, waardoor ze leren plannen. Bovendien kom je er in de toekomstige banen niet onderuit. Het niveau in haar klas is wisselend, omdat sommige er het een en ander vanaf weten (IPAD) en anderen weten er helemaal niets over. Na de drie interventies denken de leerlingen meer in stappen en hebben ze er kennis mee gemaakt. Daarna zullen ze zich verder moeten ontwikkelen. Dit moet zeker gebeuren, als ze er nog helemaal niet mee bekend zijn. Deze leerkracht weet het verschil niet tussen programmeren en computational thinking.

Deelvraag 4: Hoe wordt computational thinking in het huidige onderwijs ervaren door de leerlingen van groep 3?

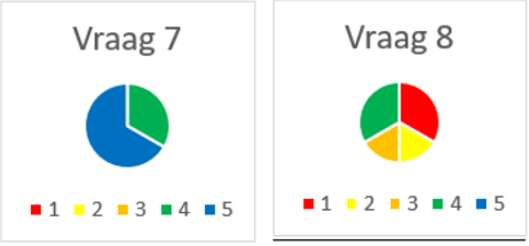
Gestructureerde vragenlijst leerlingen

Vijf van de zes leerlingen zou graag programmeren in de klas (hebben dit allemaal nog nooit gedaan), omdat ze het leuk en interessant vinden. De andere leerling vindt zijn eigen robot thuis niet leuk. Vier van de zes leerlingen heeft thuis nooit geprogrammeerd. De andere twee hebben een robot laten bewegen. Vijf van de zes leerlingen wisten niet wat programmeren was. De andere leerling vertelde een verkeerde betekenis, namelijk dat de juffrouw iets zegt en dat jij het moet opschrijven. Alle leerlingen vinden het belangrijk om te doen op school, omdat het goed voor je is (je leert ervan). Daarnaast zeiden ze ook dat je het later kan gebruiken met je werk of thuis om dingen te maken. Eén leerling gaf aan dat je uiteindelijk meer weet over wetenschap. Een voorbeeld van een vragenlijst staat in bijlage 10.

Deelvraag 5: In welke maten worden de computational thinking skills van de leerlingen in groep 8 bevorderd gedurende drie activiteiten?

Gestructureerde vragenlijst leerlingen nulmeting

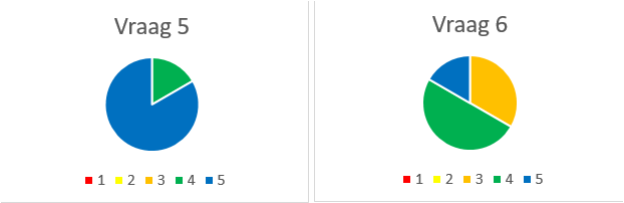
In grafiek 1 is af te lezen dat de leerlingen programmeren leuk lijken te vinden. Eén derde van de leerlingen denkt dat ze helemaal niet kunnen programmeren. Eén derde denkt dat ze wel kunnen programmeren en de overige leerlingen denken dat ze het niet zo goed kunnen. De frequentieverdeling en de vragenlijst zijn te vinden in bijlage 9.



Grafiek 1: hoe leuk is programmeren (vraag 7) en hoe goed kunnen ze programmeren (vraag 8)?

Gestructureerde vragenlijst leerlingen effectmeting

De leerlingen hebben geleerd dat ze op veel verschillende manieren kunnen programmeren. Ze denken dat programmeren betekent dat je verschillende stappen in een vlak sleept en dat het dan wordt uitgevoerd. De leerlingen verschillen van mening waarom het belangrijk is op school. Drie leerlingen zeiden dat je het later nodig hebt, twee leerlingen beweerden dat je in stappen leert denken en een andere leerling gaf aan dat het gebruikmaken van de computer bij de tijd van tegenwoordig hoort. Vijf leerlingen willen dit vaker doen in de klas, omdat ze het leuk vonden en er beter in zijn geworden. Eén leerling vindt het geen schoolvak. Dit is te zien in grafiek 2. De frequentieverdeling over vraag vijf en zes en de vragenlijsten zijn te vinden in bijlage 9.

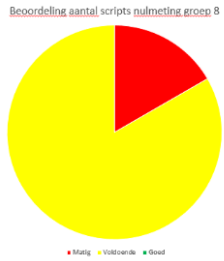


Grafiek 2: hoe leuk is programmeren (vraag 5) en hoe goed kunnen ze programmeren (vraag 6)?

Beoordeling aantal scripts nulmeting

In grafiek 3 is te zien dat vijf leerlingen voldoende scripts hebben gebruikt. In het aantal scripts zat weinig verschil tussen de leerlingen. Eén leerling scoorde matig. De leerlingen gebruikten de bewegingen twee tot vijf keer. Het uiterlijk en geluid gebruikten ze één tot drie keer. Bij gebeurtenissen is er een duidelijk verschil. De ene helft gebruikt dit één of twee keer en de andere

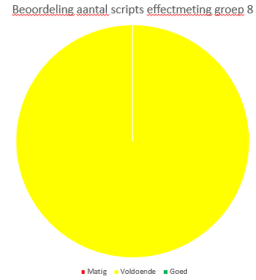
helft gebruikt dit vier of vijf keer. Dit is ook het geval bij besturen. De frequentietabel en alle cirkeldiagrammen staan in bijlage 11.



Grafiek 3: beoordeling aantal scripts

Beoordeling aantal scripts effectmeting

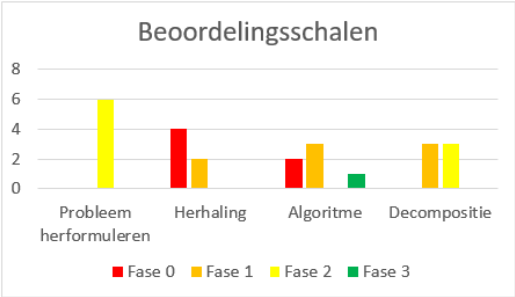
In grafiek 4 is te zien dat alle leerlingen voldoende scripts gebruikten. De leerlingen gebruikten het script bewegen twee tot zeventien keer. Dit geldt ook voor de gebeurtenissen. Een enkele leerling gebruikte dit script een of twee keer. Het uiterlijk gebruikte de helft van de leerlingen helemaal niet en de andere drie leerlingen vier, vijf en acht keer. Dit loopt uiteen. Het geluid wordt nul, twee, drie en vier keer gebruikt. Bij het script besturen liepen de aantallen ook uiteen. De helft gebruikten deze vaardigheid helemaal niet. De drie anderen gebruikten dit script twee, vier of zeven keer. De frequentietabel en alle cirkeldiagram worden weergegeven in bijlage 11.



Grafiek 4: beoordeling aantal scripts

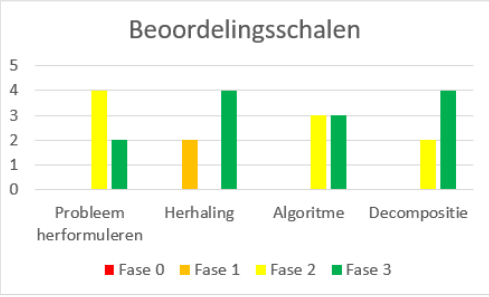
Beoordeling niveau denkvaardigheid (fases van de leerlijn) nulmeting

In grafiek 5 is te zien dat alle leerlingen een probleem konden oplossen met behulp van de computer (fase twee), maar niet konden reflecteren op de genomen stappen in dat proces (fase drie). Vier leerlingen hadden nog geen kennis van herhalingen, maar twee leerlingen konden wel een taak eindeloos laten herhalen. Twee leerlingen hadden veel hulp nodig bij het maken van een reeks en zagen hun eigen fouten niet. Hierdoor behaalden deze leerlingen fase nul bij algoritmes. In grafiek 5 is te zien dat drie leerlingen fase één behaalden, omdat ze hun eigen fouten eruit haalden en in een logische volgorde konden programmeren. Eén leerling zat al in fase drie, omdat hij helemaal zijn eigen reeks had gemaakt met behulp van een computercode. De helft van de leerlingen kon een eenvoudige reeks in een logische volgorde programmeren. De andere helft verdeelde een stap in andere deelstappen. Hierdoor werd het steeds complexer, omdat er verschillende functies en scripts werden gebruikt. De frequentietabel is te vinden in bijlage 12.



Grafiek 5: fasen computational thinking denkvaardigheden

Beoordeling niveau denkvaardigheid (fases van de leerlijn) effectmeting
In grafiek 6 is te af te lezen dat vier leerlingen een probleem konden oplossen met de computer en hier de toegevoegde waarde van inzagen. Twee leerlingen konden door middel van de vragen van de onderzoeker reflecteren op hun eigen aanpak en oplossingen. Hierbij gaven ze aan wat de meerwaarde was van de computer en wat ze de volgende keer nog meer wilden benutten. Twee leerlingen snapten de functie van herhalen, maar hebben dit niet toegepast. De overige vier leerlingen hebben de taak met het computerprogramma laten herhalen. Alle leerlingen konden hun taak in een logische volgorde plaatsen. Daarbij konden alle leerlingen hun fouten ontdekken en corrigeren. Drie leerlingen schreven ook een eenvoudige computercode. Twee leerlingen lieten zien dat ze hun taak konden opdelen in deelstappen. Echter waren dit eenvoudige en korte deelstappen. Vier leerlingen konden een grotere en complexere taak opdelen in deelstappen. De frequentietabel is te vinden in bijlage 12.



Grafiek 6: fasen computational thinking denkvaardigheden

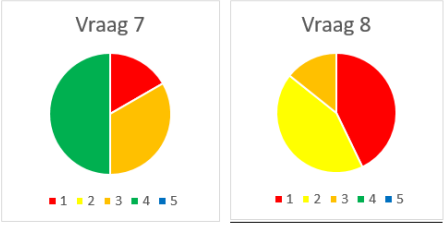
Observatie ontwikkeling complexiteit in de vier denkvaardigheden en samenwerking
De volledige observaties zijn uitgeschreven in bijlage 13. Ook de algemene resultaten zijn daar beschreven. De essentiële informatie waar in de conclusie en discussie op wordt teruggekomen, is hieronder beschreven.
Interventie 1 Microbit De leerlingen hebben kennis vergaard over herhalingen, maar sommige hebben dit niet gebruikt in hun eigen script. Deze leerlingen kozen ervoor om de alsdan functie te gebruiken (als er werd geschud of op een knop werd gedrukt dan…). Dit was recenter aan bod gekomen. De andere leerlingen lieten hem de hele tijd herhalen. De meeste leerlingen hielpen elkaar en haalden hun eigen fouten eruit. Twee meiden vonden het lastig om hun eigen fouten eruit te halen, waarbij deze leerlingen nog wat extra uitleg vroegen over de functies. Hierdoor snapten ze het wel, maar de reeks werd niet erg complex. De anderen zorgden voor een complexere reeks door veel verschillende deelstappen te gebruiken. Ze ontdekten zelf de mogelijkheden van het programma. Het was opvallend dat de twee meiden niet zelf op ontdekkingstocht uitgingen.
Interventie 2 Codemonster De leerlingen programmeerden individueel en drie leerlingen waren soms bang om fouten te maken. Hierdoor zochten ze steeds bevestiging bij de leerkracht. Vooral de jongens haalden snel hun eigen fouten eruit door te experimenteren. Dit programmeren was complex, omdat er gebruik werd gemaakt van computercodes, kleuren, transparantie, verschillende grootes en vormen (tekstueel). De meeste leerlingen doorliepen de eerste zes levels. Twee leerlingen kwamen twee levels verder, waarin ze zelf een computercode schreven en gebruik maakten van verschillende lijnen. Opvallend was dat dit twee jongens waren en deze jongens ontdekten veel zelfstandig. De leerlingen snapten de computercode en konden deze laten herhalen.
Interventie 3 bekers De leerlingen begonnen met een eenvoudig bouwsel, maar vervolgens kozen ze complexere bouwsels met verschillende afstanden, kleuren en hoogtes. De leerlingen bedachten zelf een oplossing voor deze problemen. De leerlingen gaven elkaar aanwijzingen en de bouwer liet gelijk zien wat de programmeur opschreef. Door het zelf uit te proberen, werden veel fouten voorkomen.

Twee leerlingen zochten af en toe nog bevestiging bij de onderzoeker of ze het goed deden. Dit waren weer dezelfde twee meiden als bij interventie één.

Deelvraag 6: In welke maten worden de computational thinking skills van de leerlingen in groep 3 bevorderd gedurende drie activiteiten?

Gestructureerde vragenlijst leerlingen nulmeting

In grafiek 7 is te zien dat de helft van deze leerlingen programmeren leuk vindt. Twee leerlingen vinden het een beetje leuk en één leerling vindt het helemaal niet leuk. De leerlingen denken over het algemeen dat ze niet goed zijn in programmeren. De frequentieverdeling en de vragenlijst zijn te vinden in bijlage 10.



Grafiek 7: hoe leuk is programmeren (vraag 7) en hoe goed kunnen ze programmeren (vraag 8)?

Gestructureerde vragenlijst leerlingen effectmeting

De leerlingen hebben vooral geleerd hoe je onder andere robots bewegingen kunt laten uitvoeren. Ze beweerden dat programmeren het besturen van een robot met stapjes betekent, zoals naar voren en naar achteren bewegen. Ze gaven aan dat programmeren belangrijk is om te leren voor later. Drie leerlingen gaven aan dat het belangrijk is voor een toekomstige baan en een leerling dacht dat je jezelf beter kan ontwikkelen in groep 8. Vijf leerlingen willen dit vaker doen in de klas, omdat ze het leuker vonden en ze zichzelf hebben ontwikkeld (af te lezen in grafiek 8). Ze verwachten dit nog meer te doen in de toekomst. Eén leerling vindt het niet leuker dan rekenen of lezen. De frequentieverdeling over vraag vijf en zes en de vragenlijsten zijn te vinden in bijlage 10.

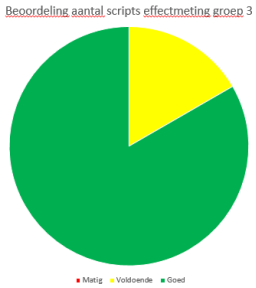
Grafiek 8: hoe leuk is programmeren (vraag 5) en hoe goed kunnen ze programmeren (vraag 6)?

Beoordeling aantal scripts nulmeting

In grafiek 9 is af te lezen dat twee leerlingen de scripts matig gebruikten, twee leerlingen de scripts voldoende gebruikten en twee leerlingen de scripts meer dan voldoende gebruikten. Vijf leerlingen hebben gebeurtenissen één keer ingezet, omdat dit nodig was om te beginnen. Een andere leerling zette het drie keer in. Bewegen en herhalen hebben de leerlingen nul tot drie keer ingezet of vijf tot tien keer. Hier zit een groot verschil in. Tekst en uiterlijk hebben alle leerlingen één of twee keer gebruikt. Audio is erg wisselend. Een leerling gebruikt het helemaal niet, twee leerlingen één keer, een leerling drie keer, een leerling vijf keer en een leerling zes keer. Stoppen en verder zijn helemaal niet gebruikt. De frequentieverdeling en alle cirkeldiagrammen zijn te vinden in bijlage 14.

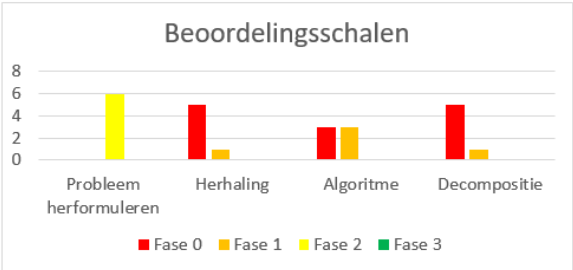
Grafiek 9: beoordeling aantal scripts

Beoordeling aantal scripts effectmeting
In grafiek 10 is af te lezen dat vijf leerlingen meer dan voldoende scripts hebben gebruikt en dit goed beheersten. Eén leerling beheerst dit voldoende. Gebeurtenissen gebruikten vijf leerlingen alleen op het begin om een reeks te starten. Eén leerling gebruikten het twee keer. Bij beweging zat er een groot verschil in aantallen. De helft van de leerlingen gebruikte deze vaardigheid tien tot zestien keer en de anderen twee tot vier keer. De leerlingen gebruikten teksten en uiterlijk en herhalingen heel wisselend. Teksten en uiterlijk gebruikten een leerling helemaal niet, één leerling één keer, één leerling twee keer, twee leerlingen twee keer en één leerling vier keer. Herhalingen gebruikten één leerling helemaal niet, één leerling één keer, twee leerlingen twee keer, één leerling drie keer en één leerling acht keer. Audio en stoppen gebruikten alle leerlingen nul, één of twee keer. De frequentietabel en alle cirkeldiagrammen worden weergegeven in bijlage 14.



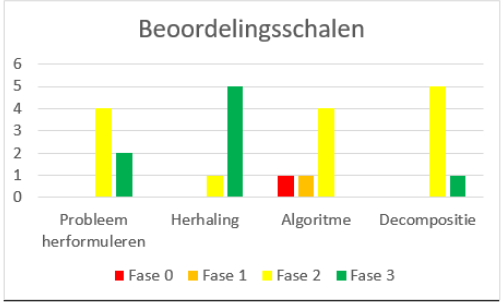
Grafiek 10: beoordeling aantal scripts

Beoordeling niveau denkvaardigheid (fases van de leerlijn) nulmeting
In grafiek 11 is af te lezen dat alle leerlingen fase twee behaalden bij het herformuleren van een probleem, omdat ze een probleem konden oplossen met behulp van een computer. Echter konden ze nog niet goed reflecteren op hun aanpak. Ze hadden veel hulp nodig van de leerkracht om te kijken naar hun aanpak. Vijf leerlingen hadden geen herhaling toegepast, omdat ze nog niet genoeg kennis hadden van deze functie. In grafiek 11 is te zien dat bij de algoritmes de helft van de leerlingen in fase nul zit. Deze leerlingen konden hun eigen fouten niet uit de reeks halen. Hierdoor ontstond er een onsamenhangende reeks. De andere helft heeft fase één bereikt, omdat ze een algoritme konden representeren met symbolen in een logische volgorde. Deze leerlingen haalden ook zelfstandig de fouten uit de reeks. Bij de decompositie zitten vijf leerlingen in fase nul in grafiek 11, omdat ze de deelstappen niet in een logische volgorde konden plaatsen. Eén andere leerling haalde fase één, omdat hij het proces had onderverdeeld in verschillende deelstappen en deze in een logische volgorde kon plaatsen. De frequentietabel is te vinden in bijlage 15.



Grafiek 11: fasen computational thinking denkvaardigheden

Beoordeling niveau denkvaardigheid (fases van de leerlijn) effectmeting
In grafiek 12 is af te lezen dat alle leerlingen met de computer een probleem opgelost hebben. Twee leerlingen konden aan de hand van de vragen van de leerkracht reflecteren op de genomen stappen. De vier andere leerlingen konden nog niet reflecteren op hun aanpak. Eén leerling gaf hierbij aan dat de computer de taken kan blijven uitvoeren, zodat het sneller gaat. Deze leerling kon alleen nog geen herhaling zelf toepassen (fase twee). De andere vijf leerlingen konden een taak meerdere keren of eindeloos laten herhalen. Eén leerling kon de deelstappen niet in een logische volgorde plaatsen (fase nul). Deze leerling kon zijn eigen fouten niet corrigeren. Eén andere leerling kon de deeltaken wel in een logische volgorde zetten. Deze leerling had af en toe hulp nodig van de leerkracht, maar kon een fout uit de reeks verwijderen. De overige vier leerlingen konden hun eigen fouten ontdekken en deze gelijk corrigeren. Ze konden een algoritme representeren met symbolen. Vijf leerlingen konden elke stap opdelen in verschillende deelstappen. Eén leerling kon een complexe taak uitvoeren, omdat deze leerling de taak kon opdelen in veel verschillende deeltaken. Hierbij gebruikte hij complexere functies. De frequentietabel is te vinden in bijlage 15.

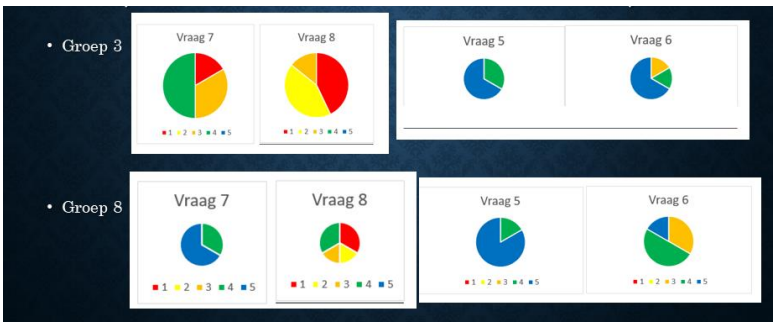


Grafiek 12: fasen computational thinking denkvaardigheden

Observatie ontwikkeling complexiteit in de vier denkvaardigheden en samenwerking
De volledige observaties zijn uitgeschreven in bijlage 16. Ook de algemene resultaten zijn daar beschreven. De essentiële informatie waar in de conclusie en discussie op wordt teruggekomen, is hieronder beschreven.
Interventie 1 Codewise Alle leerlingen vonden het lastig om met logische stappen te programmeren. Door aanwijzingen van de leerkracht kregen ze hier meer inzicht in. Er zat wel een verschil tussen de niveaus van de leerlingen. De complexiteit steeg door het gebruik van de gele, oranje en paarse kaarten. De leerlingen vonden het uitdagend om deze te gebruiken. Hierdoor gebruikten ze herhalingen (voor de hele reeks of kleine stukken), de alsdan functie en verschillende snelheden voor bewegingen. Alle leerlingen gaven elkaar aanwijzingen als er fouten werden gemaakt, waardoor ze samen hun fouten eruit konden halen. Twee leerlingen waren op het begin bang om fouten te maken, maar ervaarden dat fouten maken niet erg is. Dit lieten ze snel los. Dit was een jongen en een meisje.
Interventie 2 Blue-Bot De leerlingen vonden het lastig om de robot een andere kant op te laten lopen. Daarnaast dachten ze dat de robot zou lopen als ze hem lieten draaien. De leerlingen maakten hier eerst fouten in. Door het veel te proberen, haalden ze hun eigen fouten eruit. In bijlage 16 staat uitgelegd hoe de leerlingen dit deden. Ze hielpen elkaar en de complexiteit steeg door meerdere deelstappen/letters.
Interventie 3 bekers Eén leerling zat op de gang. De andere leerlingen testen hun bouwsel door zelf de rol van een robot in te nemen. Hierdoor werden er bijna geen fouten gemaakt of ze verbeterden hun fouten. De leerlingen bouwden regelmatig in een logische volgorde van dichtbij het startpunt naar verder weg. Door het zelf uit te voeren met behulp van blokken, kwamen ze erachter dat het handiger kon. De leerlingen gingen van eenvoudige, naar steeds complexere bouwwerken.

Deelvraag 7: Wat zijn de verschillen en overeenkomsten tussen de ontwikkeling van computational thinking skills in groep 3 en 8?

Vragenlijst

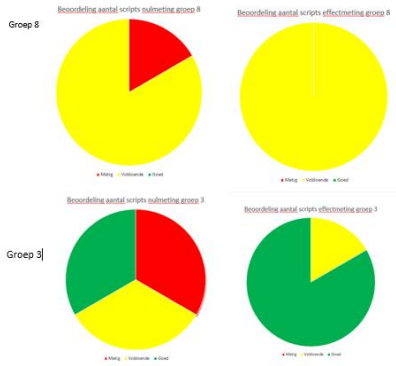


Grafiek 13: Hoe leuk is programmeren (vraag 7 en vervolgens vraag 5) en hoe goed kunnen ze programmeren (vraag 8 en vervolgens vraag 6) in groep 3 en 8?

Gestructureerde vragenlijst leerlingen

Door vergelijking van de nul- en effectmeting van de vragenlijst bij de leerlingen in groep 3 en 8 kan er geconcludeerd worden dat het merendeel van alle leerlingen vindt dat ze beter kunnen programmeren na de drie interventies, ongeacht hun leeftijd. De leerlingen van groep 3 beweren bij de laatste vraag wel (grafiek 13) dat zij er iets beter in zijn geworden dan groep 8.

Beoordeling aantal scripts



Grafiek 14: Beoordeling aantal scripts groep 3 en 8

De vergelijking hierboven laat zien dat de leerlingen van groep 3 nu meer scripts gebruiken dan voorheen. De volledige uitwerking van de gebruikte scripts is te vinden in bijlage 14. De gebeurtenissen gebruikten de leerlingen even vaak. De scripts beweging, teksten en uiterlijk, herhalingen en stoppen werden meer gebruikt. Opvallend was het script audio, want die werd bij de effectmeting minder vaak gebruikt. De concrete getallen staan hieronder genoemd.

Groep 3 leerling	**1**	**2**	**3**	**4**	**5**	**6**
Gebeurtenissen	1	3	1	1	1	1
Beweging	10	16	2	5	2	2
Teksten en uiterlijk	0	0	1	2	1	2
Audio	5	3	6	0	1	1
Herhalingen	0	0	3	6	0	1
Stoppen	0	0	0	0	0	0

Aantal keren dat scripts zijn gebruikt door elke individuele leerling nulmeting

Groep 3 leerling	**1**	**2**	**3**	**4**	**5**	**6**
Gebeurtenissen	1	2	1	1	1	1
Beweging	11	16	10	4	2	2

Teksten en uiterlijk	2	4	0	1	3	3
Audio	2	2	1	0	1	1
Herhalingen	8	2	3	2	0	1
Stoppen	1	0	1	1	0	1

Aantal keren dat scripts zijn gebruikt door elke individuele leerling effectmeting

De vergelijking hierboven laat zien dat vijf leerlingen van groep 8 evenveel scripts gebruikten bij de nulmeting als bij de effectmeting. Een leerling is meer scripts gaan gebruiken. De volledige uitwerking van de gebruikte scripts is te vinden in bijlage 11. De leerlingen gebruikten wel andere scripts. Ze benutten nu meer de scripts bewegen, geluid en gebeurtenissen. Bij gebeurtenissen zat er wel een groot verschil tussen de leerlingen. Ze zijn de scripts bewegen en geluid meer gaan gebruiken. Het script uiterlijk is opvallend, want eerst gebruikten twee leerlingen het niet, één leerling één keer en twee leerlingen twee keer. Bij de effectmeting gebruikten drie leerlingen het niet en één leerling vier, vijf en acht keer. Dus één leerling heeft dit niet benut bij de effectmeting en wel bij de nulmeting. De andere drie leerlingen hebben dit script meer gebruikt. Bij besturen zijn de resultaten wisselend. De concrete aantallen zijn hieronder beschreven.

Groep 8 leerling	**1**	**2**	**3**	**4**	**5**	**6**
Beweging	2	1	3	5	4	4
Uiterlijken	2	1	0	3	0	2
Geluid	0	1	1	0	1	2
Gebeurtenissen	1	1	1	4	4	2
Besturen	0	0	1	5	4	0

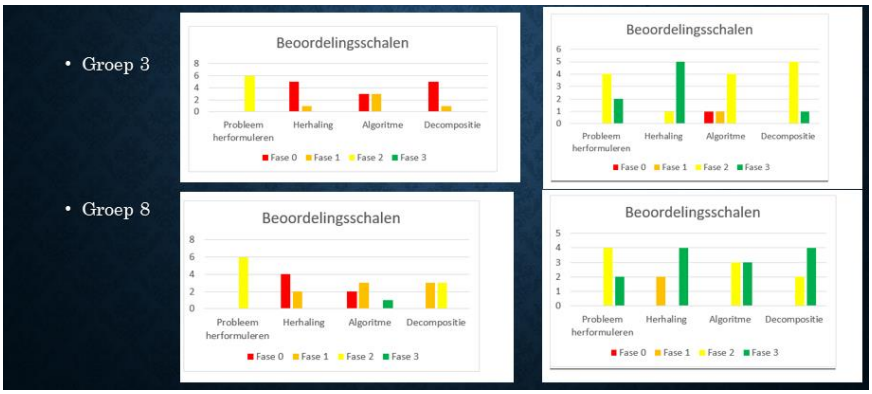
Aantal keren dat scripts zijn gebruikt door elke individuele leerling nulmeting

Groep 8 leerling	**1**	**2**	**3**	**4**	**5**	**6**
Beweging	4	5	2	9	17	3
Uiterlijken	0	4	0	8	0	5
Geluid	2	3	2	0	4	3
Gebeurtenissen	1	6	2	4	7	5
Besturen	2	0	2	4	7	0

Aantal keren dat scripts zijn gebruikt door elke individuele leerling effectmeting

Door dit onderzoeksinstrument kan er geconcludeerd worden dat de leerlingen van groep 3 in totaal meer scripts gebruikten, in tegenstelling tot groep 8. Bij groep 8 is er maar één leerling meer scripts gaan gebruiken en bij groep drie zijn er vier leerlingen meer scripts gaan gebruiken. Er zat wel een verschil in de complexiteit van de functies. Groep 3 had veel gebruikgemaakt van bewegingen, teksten en uiterlijk en herhalingen. Groep 8 had daarentegen meer gebruikgemaakt van het uiterlijk, geluid, gebeurtenissen, besturen en bewegen.

Niveau denkvaardigheid



Grafiek 15: Beoordeling niveau denkvaardigheid (fases van de leerlijn) groep 3 en 8

De vergelijking hierboven laat zien dat er twee leerlingen uit groep 3 één fase vooruit zijn gegaan bij het herformuleren van een probleem. De leerlingen werden meer bewust van hun eigen beslissingen. Alle leerlingen waren bij herhaling één, twee of drie fases gestegen. Dit geldt ook voor probleem decompositie, omdat de leerlingen hebben geoefend om een taak in deelstappen op te delen. Bij de algoritmes zijn vier leerlingen een of twee fases gestegen. Dit kwam, omdat ze een reeks konden programmeren in een logische volgordes met behulp van symbolen. Deze leerlingen konden hun eigen fouten eruit halen. Twee leerlingen zijn in dezelfde fase gebleven, omdat ze niet logisch konden programmeren en dit niet zelfstandig konden verbeteren. De volledige uitwerking van de denkniveaus van groep 3 is te zien in bijlage 15. De vergelijking hierboven laat zien dat er twee leerlingen van groep 8 één fase zijn gestegen bij het herformuleren van een probleem. Dit kwam, omdat ze op hun eigen aanpak konden reflecteren. Bij herhaling zijn vier leerlingen één fase gestegen en twee leerlingen zijn twee fases gestegen. Bij de algoritmes zijn vijf leerlingen één of meerdere fases gestegen en er is een leerling in dezelfde fase gebleven. Twee leerlingen zijn twee fases gestegen en drie leerlingen één fase. Daarnaast konden alle leerlingen de deelstappen in een logische volgorde zetten en hun fouten er uithalen. Bij probleem decompositie waren twee leerlingen één of geen fase gestegen en de andere vier waren twee fases gestegen. De leerlingen kunnen een complexere taak opdelen in verschillende deelstappen. Voorheen lukte dit nog niet. De volledige uitwerking van de denkniveaus van groep 8 is te vinden in bijlage 12.

Punten	*Probleem herformuleren*	*Herhaling*	*Algoritme*	*Decompositie*
Groep 3 nulmeting	*12*	*1*	*3*	*1*
Groep 3 effectmeting	*14*	*17*	*9*	*13*
Groep 8 nulmeting	*12*	*2*	*6*	*9*
Groep 8 effectmeting	*14*	*14*	*15*	*14*

Punten voor het niveau per denkvaardigheid

Door dit onderzoeksinstrument van groep 3 met groep 8 te vergelijken, is te zien dat de ontwikkeling verschilt per denkvaardigheid. Bij het herformuleren van een probleem hadden ze dezelfde ontwikkeling doorgemaakt, omdat ze beide twee punten vooruitgingen. Bij herhaling waren de leerlingen van groep 3 meerdere fases vooruitgegaan, in vergelijking met groep 8. Groep drie is zestien punten vooruitgegaan en groep 8 twaalf punten. Bij de algoritmes waren de leerlingen van groep 8 meerdere fasen vooruitgegaan, in vergelijking met groep 3. Groep 3 ging zes punten vooruit en groep 8 ging negen punten vooruit. Bij probleem decompositie waren de leerlingen van groep 3 meer vooruitgegaan dan de leerlingen van groep 8, omdat groep 3 twaalf punten vooruitging en groep 8 vijf punten.

Observatie ontwikkeling complexiteit in de vier denkvaardigheden en samenwerking

De overeenkomst tussen groep 3 en 8 is dat de leerlingen elkaar allemaal probeerden te helpen en leerden van de gemaakte fouten. Daarnaast wilden alle leerlingen programmeren uitproberen en konden alle leerlingen steeds complexere functies benutten. In beide groepen lag de beginsituatie van de leerlingen ver uit elkaar. Een opvallend verschil is dat de leerlingen van groep 3 bij bijna alle interventies gelijk aan de slag gingen. Het verschil met groep 8 was dat sommige leerlingen veel bevestiging zochten bij hun tweetal of bij de onderzoeker en niet gelijk aan de slag konden. Daarnaast vond groep 3 het lastig om in een logische volgorde te programmeren, terwijl dit bij de leerlingen van groep 8 goed ging. Daarnaast konden de leerlingen in groep 3 de draairichting niet bepalen en de leerlingen uit groep 8 wel. De volledige observaties zijn uitgeschreven in bijlage 13 en 16.

Conclusie en discussie

In dit hoofdstuk worden de resultaten geïnterpreteerd en wordt er een conclusie van het onderzoek beschreven op basis van de onderzoeksinstrumenten. Vervolgens wordt het onderzoek kritisch bekeken en worden er aanbevelingen gedaan voor de praktijk en een vervolgonderzoek.

De hoofdvraag is: Hoe beïnvloedt leeftijd van basisschoolleerlingen het bevorderen van computational thinking skills op de Howiblo?

Door vergelijking van de nul- en effectmeting van de vragenlijst bij de leerlingen in groep 3 en 8 kan er geconcludeerd worden dat het merendeel van alle leerlingen vindt dat ze beter kunnen programmeren na de drie interventies, ongeacht hun leeftijd. Door het aantal scripts en het niveau in denkvaardigheid van de twee groepen met elkaar te vergelijken, kan er geconcludeerd worden dat groep 8 meer verschillende functies gebruikt en groep 3 vooral meer functies gebruikt (Kennisnet, 2016.). Daarbij liet groep 8 een grotere vooruitgang zien, omdat meer leerlingen een ontwikkeling hadden doorgemaakt door andere scripts vaker te gebruiken. Als de leerlingen in de bovenbouw eerst tekstueel en vervolgens visueel leren programmeren, leidt dit tot betere resultaten (Jeuring, Corbalan, Es & Leeuwestein, 2016). Dit bevestigt het onderzoek. De leerlingen van groep 3 hadden bij de beoordelingsschalen van de denkniveaus een iets grotere ontwikkeling doorgemaakt op meer gebieden. Er kan dus geconcludeerd worden dat de leerlingen uit groep 3 meer scripts hadden gebruikt en in de vier fasen van computational thinking een grotere ontwikkeling hadden doorgemaakt, dan de leerlingen uit groep 8 (SLO curriculum van de toekomst, 2015). Dit komt, omdat groep 3 veel heeft gewerkt met blokken tijdens de drie interventies, waardoor ze de nieuwe functies goed konden onthouden. Dit lieten ze namelijk weer terugkomen bij de effectmeting. Leerlingen uit de onderbouw leren de programmeerconcepten snel door de blokken (Jeuring, Corbalan, Es & Leeuwestein, 2016). Dit bevestigt het onderzoek. De vooruitgang verschilde echter wel per denkvaardigheid. De leerlingen uit groep 8 lieten zien dat ze een grotere ontwikkeling hadden doorgemaakt in het gebruik van de verschillende vaardigheden.

Bij de observaties viel het op dat de meeste meiden uit groep 8 onzekerder waren dan de jongens. De jongens probeerden van alles uit en de meiden hadden meer bevestiging nodig van de leerkracht. Dit bevestigt de theorie. Hoe later leerlingen in aanraking komen met computers, hoe meer de leerlingen gelijk zeggen dat het niets voor hen is of dat ze het niet kunnen (Visser, 2017). Meisjes hebben op jonge leeftijd (al rond acht jaar) vooroordelen over hun eigen prestaties (Visser, 2017). In groep 8 was dit heel herkenbaar. Bij de meiden van groep 3 is dit echter niet herkent, omdat ze nog jong zijn.

Alle leerlingen zijn vooruitgegaan in computational thinking en alle theorie over de onderzoeksvraag komt overeen met de praktijk. Daarentegen is er een bron over creativiteit die het onderzoek tegenspreekt. Door computational thinking leren kinderen al jong om hun creatieve, oplossingsgerichte vermogen en kennis van de digitale wereld tot zich te nemen (Maes, 2015). De leerlingen hadden oplossingsgericht nagedacht en digitale kennis tot zich genomen. Echter het creatieve vermogen verschilde per opdracht. Codemonster vroeg weinig creativiteit, want ze moesten de stappen van het monster uitvoeren. De opdracht bij Scratch en het programmeren met de bekers deed wel een groot beroep op het creatieve vermogen van de leerlingen. Het is dus niet vanzelfsprekend dat computational thinking vraagt om creativiteit.

Daarnaast zijn er wel een paar leerlingen in dezelfde fase gebleven. Dit is te verklaren uit de vragenlijst en het interview uit de nulmeting, omdat sommige leerlingen hier al kennis van hadden en

anderen niet. Voor hen was dit nieuw en konden hier een grotere groei laten zien dan de leerlingen die deze vaardigheid al beheersten. Deze leerlingen bleven dus in dezelfde fase.

Discussie en aanbevelingen vervolgonderzoek

Een beperking van het onderzoek was dat het onderzoek gericht was op vier denkvaardigheden van computational thinking. Om uitspraken te doen over de vooruitgang van computational thinking moeten de andere denkvaardigheden ook onderzocht worden. Zelfs de onderzoeker is hier geïnteresseerd in geraakt, omdat de ontwikkeling van de leerlingen al verschilt bij vier denkvaardigheden. Dit is het advies van de onderzoeker voor een vervolgonderzoek, zodat de leerlingen zich in meerdere denkvaardigheden kunnen ontwikkelen. Bovendien was het onderzoek gericht op twee keer zes leerlingen. Dit is gering. Met zo'n kleine onderzoeksgroep zijn percentages in de analyse misleidend, maar bij een grotere groep zijn percentages sprekend en betrouwbaar (Harinck, 2013, p.173). Voor een betrouwbaarder onderzoek is het beter om een grotere doelgroep te nemen. Dit is aan te raden voor een vervolgonderzoek, zodat alle leerlingen van die klassen een ontwikkeling kunnen doormaken. Het onderzoek was gericht op groep 3 en 8, maar het advies van de onderzoeker voor een vervolgonderzoek is om dit breder te trekken, zoals de hele onderbouw in vergelijking met de bovenbouw. Dan kunnen alle leerlingen zich ontwikkelen en dit is ook de wens van de stageschool. Dit ligt in het verlengde van het doel van het onderzoek, omdat ze op de hele school innovatief onderwijs willen geven. Tijdens dit onderzoek was het belangrijk dat de populatie evenredig vertegenwoordigd was, zodat er een goede afspiegeling was van de populatie (Harinck, 2013, p. 120). Dit betekent dat er evenveel leerlingen uit elke jaargroep meedoen. Het onderzoek was geen representatieve afspiegeling van de school, omdat het onderzoek gaat over twee klassen. Dit is dus een advies voor een vervolgonderzoek.

De Howiblo was enthousiast geraakt over het feit dat de leerlingen zich na drie interventies zo goed hadden ontwikkeld. Daarom zou het team het leuk vinden als er in een vervolgonderzoek meer lessen worden gegeven in programmeren om te zien of de leerlingen dan blijven groeien of dat de ontwikkeling stagneert. Drie interventies geven immers weinig mogelijkheden om jezelf maximaal te kunnen ontwikkelen. Dit kon echter niet anders, omdat het onderzoek in een paar maanden uitgevoerd moest worden. Daarom is dit een goed advies voor een vervolgonderzoek. Tijdens het onderzoek hadden bepaalde leerlingen met elkaar samengewerkt. Sommige tweetallen konden elkaar goed helpen, maar anderen werkten liever niet samen. De vorming van deze tweetallen op basis van niveau was ook van invloed op de ontwikkeling van de leerlingen. In een vervolgonderzoek zouden de leerlingen zelf tweetallen kunnen maken op basis van hun niveau. Hierdoor blijft de motivatie hoog en kunnen de leerlingen laten zien wat ze kunnen. Daarnaast waren er twee leerlingen twee weken ziek, waardoor ze twee interventies achter elkaar in moesten halen. Hier zat geen tijd tussen. Voor een vervolgonderzoek is het aan te raden dat de leerlingen genoeg tijd krijgen tussen de interventies, zodat ze de vergaarde informatie kunnen laten bezinken. Dit vormt dan geen belemmering voor hun vooruitgang.

Sommige scripts werden minder vaak gebruikt bij de effectmeting dan bij de nulmeting. Dit kwam, omdat de leerlingen vaker voor andere scripts kozen. Deze scripts hadden zij nieuw aangeleerd en waren uitdagender om te gebruiken. De leerlingen waren door de onderzoeker gestimuleerd om de nieuw geleerde functies te gebruiken. De leerlingen bereikten uiteindelijk hetzelfde, maar hadden nieuwe mogelijkheden ontdekt. De leerlingen waren soms meerdere fases gestegen, omdat ze die vaardigheid nieuw hadden aangeleerd in de drie interventies. Deze vaardigheid lieten de leerlingen dus veel terugzien bij de effectmeting. Doordat ik de leerlingen had gestimuleerd om de nieuwe vaardigheden aan te leren, geeft het niet helemaal een objectief beeld. Voor een vervolgonderzoek

zou de onderzoeker de leerlingen niet stimuleren om deze scripts te gebruiken, zodat er een betrouwbaar beeld ontstaat.

Aanbevelingen voor de praktijk

De leerkrachten hebben kennis vernomen door het onderzoek. Bovendien hebben ze ideeën gekregen hoe ze het kunnen aanpakken. Het advies van de onderzoeker is voor hen om er gestructureerd en schoolbreed mee aan de slag te gaan. Dit sluit aan op het doel van het onderzoek, omdat ze vernieuwend onderwijs willen geven op de hele school. In de leerlijn staat beschreven hoe programmeren op een gestructureerde manier kan worden aangeboden in combinatie met andere vakken (SLO nationaal expertisecentrum leerplanontwikkeling, 2016). Zo kunnen alle scholen recht doen aan de vernieuwing en komt er geen nieuw vak bij (SLO nationaal expertisecentrum leerplanontwikkeling, 2016). Bijvoorbeeld tijdens een vast moment in de weektaak of in combinatie met techniek en de creatieve vakken. Het kan helpen om een werkgroep hedendaags onderwijs samen te stellen. Dit is nog een advies van de onderzoeker. Zij kunnen praktische ideeën geven om uit te voeren in de groep. Het onderzoek is maar een van de vele mogelijkheden. Dit sluit aan op het doel van het onderzoek, omdat de Howiblo innovatief onderwijs wil. Hierbij houden zij de leerlijn van het programmeren in de gaten, zodat er een logische opbouw in zit. Dit is essentieel, omdat de leerlingen door de leerlijn de vaardigheden voor later aanleren (SLO nationaal expertisecentrum leerplanontwikkeling, 2016). Volgens Remco Pijpers zijn er drie belangrijke leerpunten die we kunnen overnemen van de Britten om computational thinking in de school te integreren (2015). Ten eerste beweert Remco Pijpers dat de school een visie moet ontwikkelen (2015). Op de Howiblo wordt dit breder getrokken dan computational thinking, omdat ze innovatief onderwijs willen geven. Deze visie opstellen is een taak voor de werkgroep, waarin wel het hele team inspraak heeft. Ten tweede zegt Remco Pijpers dat een lesplan moeten worden opgesteld, bijvoorbeeld aan de hand van de leerlijn (2015). Hierbij kan er een stappenplan worden gemaakt, zodat er ook echt aan gewerkt wordt. Ten slotte moeten we volgens Remco Pijpers de Britse didactische ervaringen benutten en beginnen bij de praktijk (2015). Hierbij is het belangrijk dat leerlingen samenwerken, discussiëren en dat de lessen met veel vrijheid worden gegeven, zodat de zwakkere en de sterkere leerlingen uitloop hebben. Door deze drie tips kan het advies voor de praktijk eenvoudiger worden uitgevoerd, zodat de school er gestructureerd en schoolbreed aan de slag gaat. Voor de leerkrachten die zich er nog niet in thuis voelen, is een cursus aan te raden. Door één of twee dagen zelf te programmeren, ervaren ze hoe eenvoudig het kan zijn.

Tijdens de teamvergadering in maart zijn de uitkomsten van het onderzoek gepresenteerd. De grafische diagrammen hebben bijgedragen aan een helder beeld van de ontwikkeling van de leerlingen. In deze presentatie zijn er praktijksuggesties gegeven voor de hedendaagse onderwijspraktijk. Ook werd er een suggestie uitgevoerd, zodat ze konden ervaren hoe eenvoudig het toe te passen is in de praktijk. Dit sluit aan bij de tip van Remco Pijpers, om te beginnen bij de praktijk (2015). Hierdoor wordt de kans groter dat ze het gaan toepassen in de praktijk. Meerdere leerkrachten hebben deze programmeervorm al uitgevoerd in hun eigen klas, waardoor de theorie wordt bevestigd. De presentatie was in het bijzijn van het hele team, omdat het onderzoek over jonge leerlingen en oudere leerlingen gaat. De leerkrachten vonden de informatie zinvol, omdat ze hoorden wat het beste werkt en welke vaardigheden de leerlingen opdoen. Dit is belangrijk voor het onderwijs van de toekomst. De mentor uit groep 3 heeft de beoordeling geschreven, omdat zij het hele proces heeft gezien en aanwezig was bij de presentatie. De presentatie is opgenomen in bijlage 7.

Literatuurlijst

Harinck, F. (2013). *Basisprincipes praktijkonderzoek (14de herziene druk).* Apeldoorn: Garant. (Hoofdstuk 4, 5 en 6).

Jeuring, J., Corbalan, G., Es, N. van & Leeuwestein, H. (2016). Leren programmeren in het PO. Op 15 oktober 2017 ontleend aan https://www.nro.nl/wp-content/uploads/2016/09/003-en-036-Leren-programmeren-in-het-PO-een-literatuurreview.pdf

Jeuring, J., Corbalan, G., Montfort, J., Es, N. van. & Leeuwestein, H. (2016). Vorm en effect van programmeeronderwijs in het primair onderwijs. Op 13 oktober 2017 ontleend aan file:///C:/Users/Yvette/AppData/Local/Packages/Microsoft.MicrosoftEdge_8wekyb3d8bbwe/TempState/Downloads/Vormeneffect.pdf

Kennisnet (2016). Computational thinking in het Nederlandse onderwijs. Op 13 oktober 2017 ontleend aan https://www.kennisnet.nl/fileadmin/kennisnet/publicatie/Computational_Thinking_in_het_Nederlandse_onderwijs.pdf

Kennisnet (z.j.). Programmeren in het PO. Op 15 oktober 2017 ontleend aan https://maken.wikiwijs.nl/74282/Programmeren_in_het_PO#!page-1838175

Lucassen, M (2017). Leren programmeren: waarom eigenlijk? Op 21-01-2018 ontleend aan https://www.vernieuwenderwijs.nl/leren-programmeren-waarom-eigenlijk/

Maes, T. (2015). Computing-onderwijs in de praktijk. Wat kunnen we leren van de Britten? Op 15 oktober 2017 ontleend aan https://www.kennisnet.nl/fileadmin/kennisnet/publicatie/Computing_onderwijs_in_de_praktijk.pdf

Mediawijsheid (z.j.). Programmeren 21e -eeuwse vaardigheden. Op 13 oktober ontleend aan https://www.mediawijsheid.nl/programmeren/

NLDIT computerkennis (z.j.). Verschillen tussen Coding & Programmering. Op 10-11-2017 ontleend aan http://www.nldit.com/programmering/computer-programming-languages/201309/86705.html#.WgWivrpFzmJ

Onderwijs in de 21e eeuw (2016). Computational thinking. Op 15 oktober 2017 ontleend aan http://o21.nu/computationalthinking/

Pardoen, J. (z.j.). Wat is mediawijsheid? Over de waarde van mediawijsheidmodellen. Op 15 oktober 2017 ontleend aan https://mediawijs.be/dossiers/dossier-wat-mediawijs/wat-mediawijsheid-over-waarde-mediawijsheidmodellen

Pijpers, R. (2015). Wat wij kunnen leren van computing-onderwijs in de Britse praktijk. Op 04-04-2018 ontleend aan https://www.kennisnet.nl/artikel/wat-wij-kunnen-leren-van-computing-onderwijs-in-de-britse-praktijk/

PO-Raad (2017). Waarom de PO-Raad tegen verplícht programmeren is (maar voor digitale geletterdheid). Op 21-01-2018 ontleend aan https://www.poraad.nl/nieuws-en-achtergronden/waarom-de-po-raad-tegen-verplicht-programmeren-is-maar-voor-digitale

SLO curriculum van de toekomst (2015). Voorbeeldmatig leerplankader computational thinking. Op 13 oktober 2017 ontleend aan http://curriculumvandetoekomst.slo.nl/21e-eeuwse-vaardigheden/digitale-geletterdheid/computational-thinking/voorbeeldmatig-leerplankader

SLO nationaal expertisecentrum leerplanontwikkeling (2016). Leerlijn programmeren stoomt kinderen klaar voor de toekomst. Op 28-02-2018 ontleend aan http://www.slo.nl/nieuws/persbericht-leerlijn-programmeren/

Visser, J. (2017). Het belang van leren programmeren. Op 15 oktober 2017 ontleend aan https://decorrespondent.nl/7436/deze-onderzoeker-vindt-dat-ieder-kind-moet-programmeren/590812508-6aa25317

Woning, E. (2016). Leerlijn programmeren voor het basisonderwijs. Op 13 oktober 2017 ontleend aan https://www.mediawijsheid.nl/extern/?url=https%3A%2F%2Fwww.mediawijzer.net%2Fleerlijn-programmeren-basisonderwijs%2F

Ypenberg Moonen, D. (2017). Kinderen kansloos zonder programmeren! Op 15 oktober 2017 ontleend aan http://futurenl.org/tag/computational-thinking/

Bijlagen

Bijlage 1: Interview leerkracht

- Interview leerkracht groep 3 en 8 beginsituatie

1. Heeft u ooit programmeren toegepast in uw huidige klas?
   ……
   ……
   ……
2. Bij vraag 2 ja → Hoe heeft u dit toegepast?
   ……
   ……
   ……
3. Bij vraag 2 nee→ Waarom heeft u dit niet gedaan?
   ……
   ……
   ……
4. Hoe zou u programmeren graag in uw klas zien?
   ……
   ……
   ……
5. Waarom vindt u het wel/niet belangrijk dat programmeren wordt gedaan in de praktijk?
   ……
   ……
   ……
6. Hoe denkt u dat het niveau van programmeren van de leerlingen nu in uw groep is?
   ……
   ……
   ……
7. Wat voor ontwikkeling gaan ze volgens u doormaken als ze drie lessen programmeren hebben gevolgd? Leg uit.
   ……
   ……
   ……
8. Wat is het verschil tussen programmeren en computational thinking?
   ……
   ……
   ……

Bijlage 2: Vragenlijst leerling groep 3 en 8 nulmeting

1. Heb je ooit geprogrammeerd op de basisschool?
   ……
   ……
   ……
2. Bij vraag 2 ja → Wat heb je onthouden van die ervaring?
   ……
   ……
   ……
3. Bij vraag 2 nee→ Zou je dit graag willen doen? Leg uit.
   ……
   ……
   ……
4. Heb je ooit thuis geprogrammeerd (bijvoorbeeld met spelletjes op de computer of robots)? Leg uit hoe.
   ……
   ……
   ……
5. Wat betekent programmeren?
   ……
   ……
   ……
6. Waarom is programmeren volgens jou belangrijk op de basisschool? Leg uit.
   ……
   ……
   ……
7. Hoe leuk vind jij programmeren? Kies een score.
   1= helemaal niet leuk 2= niet leuk 3= een beetje 4= leuk 5= heel leuk

1 2 3 4 5

8. Hoe goed ben jij in programmeren? Kies een score.
   1= helemaal niet goed 2= niet goed 3= beetje 4= goed 5= heel goed

1 2 3 4 5

Bijlage 3: Vragenlijst leerling groep 3 en 8 effectmeting

1. Wat heb je geleerd van de afgelopen keren dat we hebben geprogrammeerd?
   ……
   ……
   ……
2. Wat betekent programmeren?
   ……
   ……
   ……
3. Waarom is programmeren volgens jou belangrijk op de basisschool? Leg uit.
   ……
   ……
   ……
4. Zou je dit vaker willen doen in de klas? Leg uit.
   ……
   ……
   ……
5. Hoe leuk vind jij programmeren? Kies een score.
   1= helemaal niet leuk 2= niet leuk 3= een beetje 4= leuk 5= heel leuk

1 2 3 4 5

6. Hoe goed ben jij in programmeren? Kies een score.
   1= helemaal niet goed 2= niet goed 3= beetje 4= goed 5= heel goed

1 2 3 4 5

Bijlage 4: Beoordeling aantal scripts

- Inhoudsanalyse nul- en effectmeting. Per leerling bekijken hoe vaak ze de begrippen van programmeren toepassen (vier begrippen uitgekozen). Achter elk script staat beschreven welke vaardigheid ze hier kunnen opdoen.

Groep 8

Vaardigheid		**Aantal keer**
Beweging	= algoritmes + probleem	
Uiterlijken	= algoritmes + probleem	
Geluid	= algoritmes + probleem	
Pen	= algoritmes + probleem	
Data	= algoritmes + probleem	
Gebeurtenissen	= algoritmes	
Besturen	= herhalen	
Waarnemen	= algoritmes + probleem	
Functies	= algoritmes + probleem	

Matig: minder dan vier scripts gebruikt

Voldoende: vier scripts gebruikt.

Goed: vijf of zes scripts meerdere malen gebruikt.

Groep 3

Vaardigheid		**Aantal keer**
Gebeurtenissen (starten actie)	= algoritme	
Beweging	=algoritme + probleem	
Teksten en uiterlijk	=algoritme + probleem	
Audio	=algoritme + probleem	
Herhalingen	= herhaling	
Stoppen	= algoritme	

Matig: minder dan vier scripts gebruikt

Voldoende: vier scripts gebruikt.

Goed: vijf of zes scripts meerdere malen gebruikt.

Bijlage 5: Beoordeling niveau denkvaardigheid (fases van de leerlijn)

Inhoudsanalyse nul- en effectmeting. Per leerling wordt bekeken in welke fase hij zit, hierbij worden de fases van SLO computational thinking en Kennisnet programmeren gebruikt.

1. Probleem herformuleren/ fouten

Fase 1	
Fase 2	
Fase 3	

Fase 1: verkennen van de mogelijkheden om problemen op te lossen met een computer
Fase 2: verkennen of een probleem opgelost kan worden met een computer
Fase 3: verkennen of een probleem opgelost kan worden met een computer en reflecteren op de genomen beslissingen/stappen en de gevonden oplossingen (bijv. Wat is de meerwaarde van de computer geweest?)

2. Automatisering/ herhaling

Fase 1	
Fase 2	
Fase 3	

Fase 1: beseffen dat een computer een taak eindeloos kan herhalen (bijv. in oefeningen)
Fase 2: vergelijken van resultaten van handelingen die handmatig of door apparaten zijn uitgevoerd
Fase 3: onderzoeken op welke manier een computerprogramma kan ondersteunen bij een telkens terugkerende taak of handeling

3. Algoritmes en procedures

Fase 1	
Fase 2	
Fase 3	

Fase 1: uitvoeren van een taak door stap voor stap een reeks handelingen uit te voeren en uit een reeks halen van een foute stap of instructie en deze vervangen door een juiste en plaatsen van opdrachten in een logische volgorde.
Fase 2: kennismaken met betekenis algoritme en ervaren dat een algoritme een lijst van instructies is die leidt tot een bepaald resultaat en onderzoeken van een reeks instructies of regels en oplossen van mogelijke fouten in deze reeks. Bovendien representeren en communiceren van een algoritme door middel van codes en symbolen.
Fase 3: oplossen van een probleem door het formuleren van een eigen set instructies en ontdekken van fouten in algoritmes door logisch te redeneren en verbeteren hiervan en ontwikkelen en schrijven van eenvoudige computercode.

4. Probleem decompositie

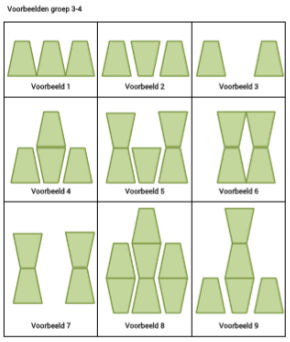
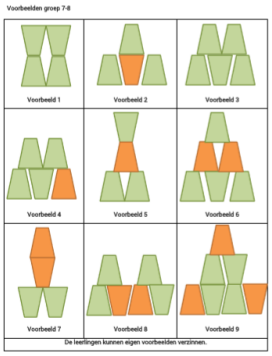
Fase 1	
Fase 2	

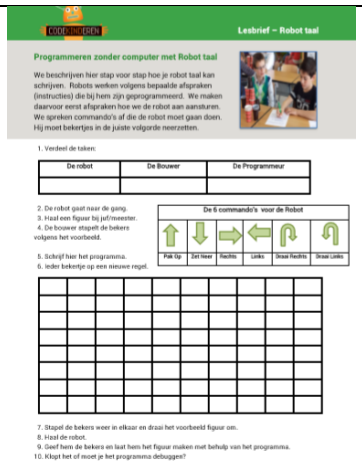
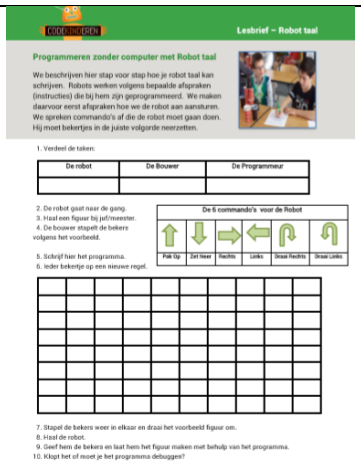
Fase 3	

Fase 1: opdelen van een eenvoudige taak in deeltaken en plaatsen in een logische volgorde
Fase 2: opdelen van een proces in verschillende stappen waar mogelijk elke stap weer in deelstappen
Fase 3: opdelen van een grotere en meer complexe taak in een aantal deeltaken

Bijlage 6: De nul- en effectmeting en de drie interventies uitgewerkt
Schuingedrukt staan de middelen die dat moment worden gebruikt (dit komt overeen met de theorie). Daaronder staat kort de opdracht toegelicht.

Meting	**Groep 3**	**Groep 8**
Nulmeting	*Scratch junior* De leerlingen krijgen een minimale instructie over de tools van het programma. De leerling moet één sprite laten lopen en praten. Dit is een app op de tablet.	*Scratch senior* De leerlingen krijgen een minimale instructie over de tools van het programma. De leerling moet twee sprites laten lopen en tegen elkaar laten praten. https://www.codekinderen.nl/leerling/programmeren/scratch/index.html
Interventie 1	*Codewise → blok* De verschillende kaarten en functies worden besproken met de leerlingen. Dan programmeert de leerling zijn maatje met zoveel mogelijk verschillende handelingen (herhalingen, boven, beneden, praten). De leerlingen krijgen uitdaging door steeds een moeilijkheidsgraad hoger te programmeren (andere kleur). 	*Microbit → tekstueel + plugged* De leerlingen krijgen een filmpje te zien over de werking van de Microbit. De leerlingen maken aan de hand van het filmpje één cijfer en zorgen dat hun naam wordt herhaald op de Microbit. Zelf mogen ze meerdere mogelijkheden uitproberen. https://www.skillsdojo.nl/microcomputer-uitvinden/programmeer-je-naam-in-licht/ https://makecode.microbit.org/?lang=nl
Interventie 2	*Blue bot= Robot* De leerlingen krijgen een korte uitleg over de robot en rijden naar de eerste letter van hun naam. Aan het einde moeten de leerlingen hun naam kunnen schrijven door in een keer alle stappen te programmeren, zodat hij naar alle letters rijdt.	*Codemonster = Plugged visueel* De leerlingen maken hier kennis met codes en symbolen. Dit wordt uitgelegd door de codemonster zelf. Ze proberen in een zo hoog mogelijk level te komen. http://www.crunchzilla.com/code-monster

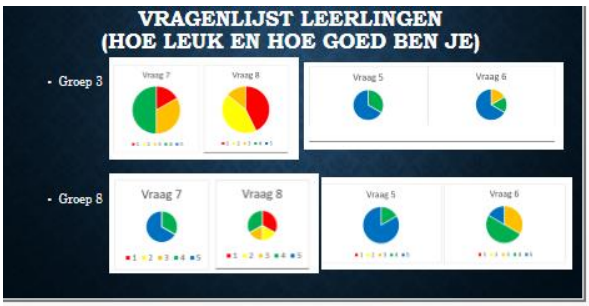
Interventie 3	*Bekertjes =Unplugged visueel+ blok* Op de lesbrief staat beschreven hoe de oefening werkt. De leerlingen gebruiken robottaal om te programmeren. Er zijn 3 rollen: de programmeur, de bouwer en de robot. De robot wacht op de gang, de programmeur schrijft de robottaal op en de bouwer probeert het vast uit. Doordat de robot het erna uitvoert, weten de leerlingen gelijk of ze het goed hebben geprogrammeerd. De leerlingen gebruiken de werkbladen voor groep 3/4. Als het te makkelijk is, mogen ze het zelf uitbreiden of de werkbladen voor groep 5/6 gebruiken.  	*Bekertjes =Unplugged visueel* Op de lesbrief staat beschreven hoe de oefening werkt. De leerlingen gebruiken robottaal om te programmeren. Er zijn 3 rollen: de programmeur, de bouwer en de robot. De robot wacht op de gang, de programmeur schrijft de robottaal op en de bouwer probeert het vast uit. Doordat de robot het erna uitvoert, weten de leerlingen gelijk of ze het goed hebben geprogrammeerd. De leerlingen gebruiken de werkbladen voor groep 7/8. Als het te makkelijk is, mogen ze het zelf uitbreiden. 

	 CODEKINDEREN Lesbrief – Robot taal Programmeren zonder computer met Robot taal We beschrijven hier stap voor stap hoe je robot taal kan schrijven. Robots werken volgens bepaalde afspraken (instructies) die bij hem zijn geprogrammeerd. We maken daarvoor eerst afspraken hoe we de robot aan aansturen. We spreken commando's af die de robot moet gaan doen. Hij moet bekertjes in de juiste volgorde neerzetten. 1. Verdeel de taken: De robot / De Bouwer / De Programmeur 2. De robot gaat naar de gang. 3. Haal een figuur bij juf/meester. 4. De bouwer stapelt de bekers volgens het voorbeeld. De 6 commando's voor de Robot Pak Op / Zet Neer / Rechts / Links / Draai Rechts / Draai Links 5. Schrijf hier het programma. 6. Ieder bekertje op een nieuwe regel. 7. Stapel de bekers weer in elkaar en draai het voorbeeld figuur om. 8. Haal de robot. 9. Geef hem de bekers en laat hem het figuur maken met behulp van het programma. 10. Klopt het of moet je het programma debuggen?	 CODEKINDEREN Lesbrief – Robot taal Programmeren zonder computer met Robot taal We beschrijven hier stap voor stap hoe je robot taal kan schrijven. Robots werken volgens bepaalde afspraken (instructies) die bij hem zijn geprogrammeerd. We maken daarvoor eerst afspraken hoe we de robot aan aansturen. We spreken commando's af die de robot moet gaan doen. Hij moet bekertjes in de juiste volgorde neerzetten. 1. Verdeel de taken: De robot / De Bouwer / De Programmeur 2. De robot gaat naar de gang. 3. Haal een figuur bij juf/meester. 4. De bouwer stapelt de bekers volgens het voorbeeld. De 6 commando's voor de Robot Pak Op / Zet Neer / Rechts / Links / Draai Rechts / Draai Links 5. Schrijf hier het programma. 6. Ieder bekertje op een nieuwe regel. 7. Stapel de bekers weer in elkaar en draai het voorbeeld figuur om. 8. Haal de robot. 9. Geef hem de bekers en laat hem het figuur maken met behulp van het programma. 10. Klopt het of moet je het programma debuggen?
Effectmeting	*Scratch junior* We herhalen de geleerde tools. Uiteindelijk moet de leerling één sprite verschillende bewegingen laten uitvoeren (omhoog/beneden, zijkanten, draaien, praten) door middel van herhalingen en door hun eigen fouten eruit te halen. De leerlingen worden door de onderzoeker uitgedaagd om hem zo goed mogelijk te maken. Na afloop kunnen ze reflecteren in een gesprek met de onderzoeker op hun genomen beslissingen. Dit is een app op de tablet.	*Scratch senior* We herhalen de geleerde tools. Uiteindelijk laat de leerling twee sprites verschillende bewegingen in deelopdrachten uitvoeren door middel van herhalingen, alsdan functie, alsdan-anders functie, wachten functie en wanneer ik kleur/toets aanraak dan…. De leerlingen worden door de onderzoeker uitgedaagd om hem zo goed mogelijk te maken. Na afloop kunnen ze reflecteren in een gesprek met de onderzoeker op hun genomen beslissingen. https://www.codekinderen.nl/leerling/programmeren/scratch/index.html

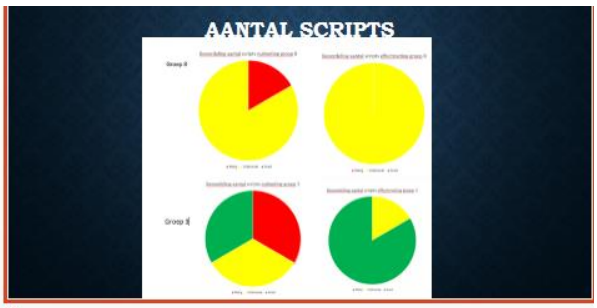
Bijlage 7 presentatie team

1

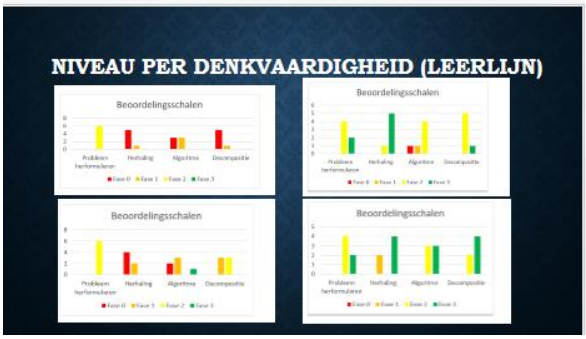
2



5



6



Bijlage 8 Matrix nulmeting interview leerkrachten groep 3 en 8

	Leraar groep 3	**Leraar groep 8**
Toegepast huidige klas	X	X Paar kinderen bij andere leerkracht
Ja → Hoe	X	X
Nee → Waarom niet?	Is er nog niet in thuis	Is er nog niet in thuis en is er niet aan toegekomen
Hoe graag zien?	Zelfstandig aan opdrachten/werkvormen met andere vakken (weektaak)	Eén keer per week in de weektaak voor begeleiding (niet klassikaal)
Wel/niet belangrijk	Wel, stap voor stap denken, plannen en toekomstige banen	Wel, met de tijd mee gaan en voor de toekomst
Niveau	Wisselend, van onwetendheid tot veel gebruik IPad	Wisselend, goed met computer leren snel en anderen minder inzicht dus meer hulp
Ontwikkeling drie lessen	Denken meer in stapjes en kennismaking	Weten wat het inhoudt en kennismaking. Sommige meer hulp voor zelfstandig programmeren.
Verschil programmeren/ computational thinking	X	X

X= kan de vraag niet beantwoorden of het is nooit gebeurd

8

Onderzoeksinstrumenten

- Interview leerkracht groep 3 en 8 beginsituatie

1. Heeft u ooit programmeren toegepast in uw huidige klas?
   Nee, wel bij ateliers andere leerkrachten. Een paar kinderen.
2. Bij vraag 2 ja → Hoe heeft u dit toegepast?
3. Bij vraag 2 nee → Waarom heeft u dit niet gedaan?
   Nog niet aan toegekomen en ik kan het zelf niet. We moeten daarvoor een cursus volgen.
4. Hoe zou u programmeren graag in uw klas zien?
   Beginnen in de weektaak, 1x in de week. Ik kan dan goed begeleiden. Niet stressvol.
5. Waarom vindt u het wel/niet belangrijk dat programmeren wordt gedaan in de praktijk?
   Belangrijk om met de tijd mee te gaan. Ze hebben het later nodig.
6. Hoe denkt u dat het niveau van programmeren van de leerlingen nu in uw groep is?
   een aantal die goed zijn met computer kunnen met tips snel leren. De rest zou het kunnen met meer uitleg. Sommige zagen inzicht lastiger.
7. Wat voor ontwikkeling gaan ze volgens u doormaken als ze drie lessen programmeren hebben gevolgd? Leg uit.
   Ze weten wat programmeren inhoudt. Ze proeven ervan. Beter leren meer lessen. Uiteindelijk worden ze beter, maar sommige kunnen ze heel dan wel meer hulp en uitleg.
8. Wat is het verschil tussen programmeren en computational thinking?

Groep 3

Onderzoeksinstrumenten

- Interview leerkracht groep 3 en 8 beginsituatie

1. Heeft u ooit programmeren toegepast in uw huidige klas?
   Nee
2. Bij vraag 2 ja → Hoe heeft u dit toegepast?
3. Bij vraag 2 nee → Waarom heeft u dit niet gedaan?
   Ze is er zelf nog niet in thuis.
4. Hoe zou u programmeren graag in uw klas zien?
   Opdrachten/werkvormen waar ze zelf mee aan de slag kunt, zoals de weektaak of gekoppeld aan andere vakken.
5. Waarom vindt u het wel/niet belangrijk dat programmeren wordt gedaan in de praktijk?
   Wel, je leert stap voor stap denken en het komt het leren plannen tegemoet en in de toekomstige banen kom je er niet onderuit.
6. Hoe denkt u dat het niveau van programmeren van de leerlingen nu in uw groep is?
   Heel wisselend. Sommige weten niks en ander weten wel door de Ipad (mindcraft).
7. Wat voor ontwikkeling gaan ze volgens u doormaken als ze drie lessen programmeren hebben gevolgd? Leg uit.
   Ze denken meer in stapjes, ie meer kennis en dramen ontwikkelen ze zich. Zeker als ze er niet mee bekend zijn.
8. Wat is het verschil tussen programmeren en computational thinking?
   Weet niet zo goed.

Bijlage 9 vragenlijst groep 8
Nulmeting

Score	1	2	3	4	5
Vraag 7 leuk	0	0	0	2	4
Vraag 8 goed	2	1	1	2	0

- Vragenlijst leerling groep 7 en 8 nulmeting

1. Heb je ooit geprogrammeerd op de basisschool?
   niet dat ik weet
2. Bij vraag 2 ja → Wat heb je onthouden van die ervaring?
3. Bij vraag 2 nee→ Zou je dit graag willen doen? Leg uit.
   het lijkt me wel leuk dus jes
4. Heb je ooit thuis geprogrammeerd (bijvoorbeeld met spelletjes op de computer of robots)? Leg uit hoe.
   nee
5. Wat betekent programmeren?
   iets met computers een spel maken ofzo
6. Waarom is programmeren volgens jou belangrijk op de basisschool? Leg uit.
   dat je ook leert om met computers omgaan
7. Hoe leuk vind jij programmeren? Kies een score.
   1= helemaal niet leuk 2= niet leuk 3= een beetje 4= leuk 5= heel leuk

   1 2 3 (4) 5
8. Hoe goed ben jij in programmeren? Kies een score.
   1= helemaal niet goed 2= niet goed 3= beetje 4= goed 5= heel goed

   (1) 2 3 4 5

Effectmeting

Score	1	2	3	4	5
Vraag 5 leuk	0	0	0	1	5
Vraag 6 goed	0	0	2	3	1

- Vragenlijst leerling groep [redacted] en 8 effectmeting

1. Wat heb je geleerd van de afgelopen keren dat we hebben geprogrammeerd?
   Soorten dingen programeren

2. Wat betekent programmeren?
   Dingen maken via verschillende stappen

3. Waarom is programmeren volgens jou belangrijk op de basisschool? Leg uit.
   lesen met de computer om te gaan en via stapjen leren

4. Zou je dit vaker willen doen in de klas? Leg uit.
   ja want het is leuk en je leert ook wat

5. Hoe leuk vindt jij programmeren? Kies een score.
   1= helemaal niet leuk 2= niet leuk 3= een beetje 4= leuk 5= heel leuk

   1 2 3 4 (5)

6. Hoe goed ben jij in programmeren? Kies een score.
   1= helemaal niet goed 2= niet goed 3= beetje 4= goed 5= heel goed

   1

- Vragenlijst leerling groep [redacted] en 8 effectmeting

1. Wat heb je geleerd van de afgelopen keren dat we hebben geprogrammeerd?
   dat er veel verschilende manieren zijn om te programeren en dat het eigenlijk best makkelijk is

2. Wat betekent programmeren?
   door verschillende stappen iets iets te laten doen

3. Waarom is programmeren volgens jou belangrijk op de basisschool? Leg uit.
   je kan er veel van leren

4. Zou je dit vaker willen doen in de klas? Leg uit.
   ja het is leuk om te de verschilende dingen te maken

5. Hoe leuk vindt jij programmeren? Kies een score.
   1= helemaal niet leuk 2= niet leuk 3= een beetje 4= leuk 5= heel leuk

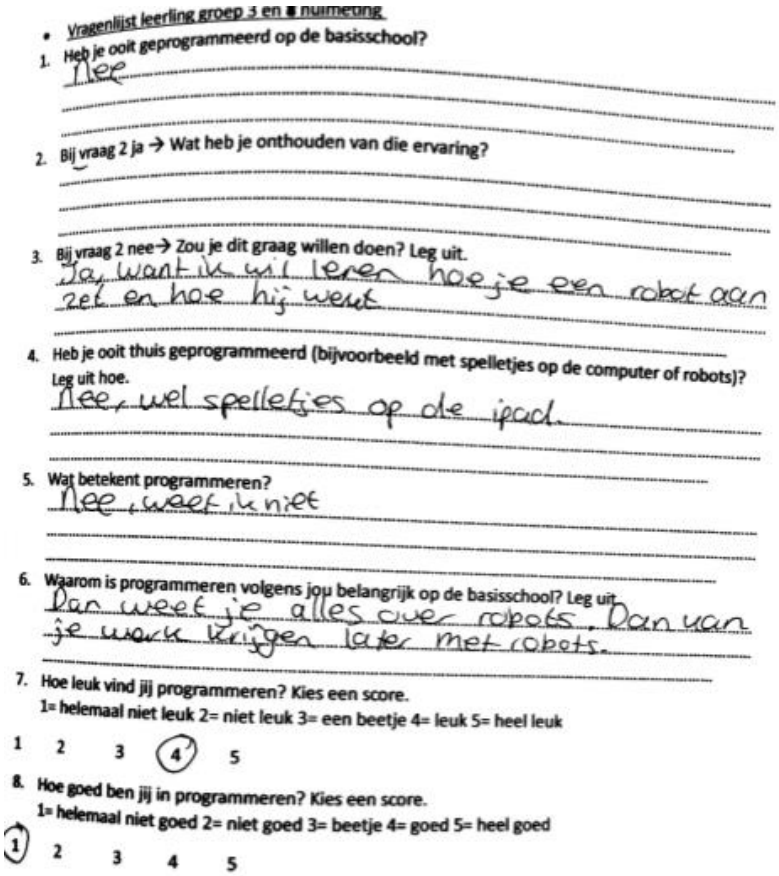
   1 2 3 4 (5)

6. Hoe goed ben jij in programmeren? Kies een score.
   1= helemaal niet goed 2= niet goed 3= beetje 4= goed 5= heel goed

   1 2 3 (4) 5

Bijlage 10 vragenlijst groep 3
Nulmeting

Score	1	2	3	4	5
Vraag 7 leuk	1	0	2	3	0
Vraag 8 goed	3	2	1	0	0



- Vragenlijst leerling groep 3 en 4 nulmeting

1. Heb je ooit geprogrammeerd op de basisschool?
   Nee
2. Bij vraag 2 ja → Wat heb je onthouden van die ervaring?
3. Bij vraag 2 nee → Zou je dit graag willen doen? Leg uit.
   Ja, want ik wil leren hoe je een robot aan zet en hoe hij werkt
4. Heb je ooit thuis geprogrammeerd (bijvoorbeeld met spelletjes op de computer of robots)? Leg uit hoe.
   Nee, wel spelletjes op de ipad.
5. Wat betekent programmeren?
   Nee, weet ik niet
6. Waarom is programmeren volgens jou belangrijk op de basisschool? Leg uit.
   Dan weet je alles over robots. Dan van je werk krijgen later met robots.
7. Hoe leuk vind jij programmeren? Kies een score.
   1= helemaal niet leuk 2= niet leuk 3= een beetje 4= leuk 5= heel leuk
   1 2 3 (4) 5
8. Hoe goed ben jij in programmeren? Kies een score.
   1= helemaal niet goed 2= niet goed 3= beetje 4= goed 5= heel goed
   (1) 2 3 4 5

Effectmeting

Score	1	2	3	4	5
Vraag 5 leuk	0	0	0	2	4
Vraag 6 goed	0	0	1	1	4

- Vragenlijst leerling groep 3 en 8 effectmeting

1. Wat heb je geleerd van de afgelopen keren dat we hebben geprogrammeerd?
   [illegible] van robots
2. Wat betekent programmeren?
   iets instructies met stapjes
3. Waarom is programmeren volgens jou belangrijk op de basisschool? Leg uit.
   probleemoplossen voor later
4. Zou je dit vaker willen doen in de klas? Leg uit.
   ja omdat ik dat [illegible]
5. Hoe leuk vindt jij programmeren? Kies een score.
   1= helemaal niet leuk 2= niet leuk 3= een beetje 4= leuk 5= heel leuk

   1 2 3 4 (5)

6. Hoe goed ben jij in programmeren? Kies een score.
   1= helemaal niet goed 2= niet goed 3= beetje 4= goed 5= heel goed

- Vragenlijst leerling groep 3 en 8 effectmeting

1. Wat heb je geleerd van de afgelopen keren dat we hebben geprogrammeerd?
   dat je verschillende beweegingen kan slepen in een vak, en met robot welke kant hij op kan gaan en met kaarten herhalen geleerd
2. Wat betekent programmeren?
   programmeren is dat je bijvoorbeeld robots naar voren en naar achteren kan laten gaan
3. Waarom is programmeren volgens jou belangrijk op de basisschool? Leg uit.
   voor werk met robots weet je wat je moet doen. Veel mensen kunnen het niet, dus dan is het belangrijk
4. Zou je dit vaker willen doen in de klas? Leg uit.
   ja, ik heb het nu gedaan en wil er verder mee en ik vond het leuk. Vooral alle stapjes die je een robot kan laten zetten.
5. Hoe leuk vindt jij programmeren? Kies een score.
   1= helemaal niet leuk 2= niet leuk 3= een beetje 4= leuk 5= heel leuk

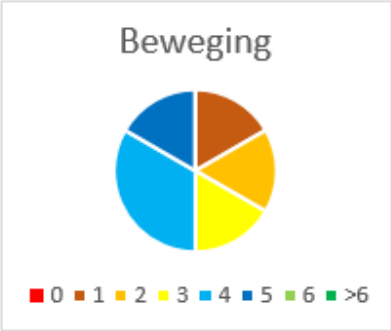
   1 2 3 4 (5)

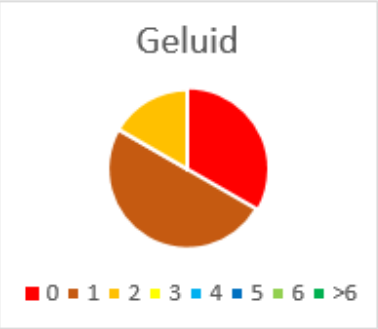
6. Hoe goed ben jij in programmeren? Kies een score.
   1= helemaal niet goed 2= niet goed 3= beetje 4= goed 5= heel goed

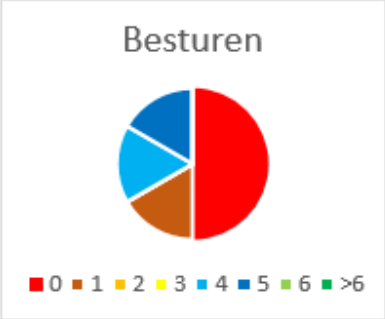
   1 2 (3) 4 5

Bijlage 11 Beoordeling aantal scripts groep 8
Nulmeting

Aantal keer	**0**	**1**	**2**	**3**	**4**	**5**	**6**	**>6**
Beweging	0	1	1	1	2	1	0	0
Uiterlijk	2	1	2	1	0	0	0	0
Geluid	2	3	1	0	0	0	0	0
Gebeurtenissen	0	3	1	0	2	0	0	0
Besturen	3	1	0	0	1	1	0	0

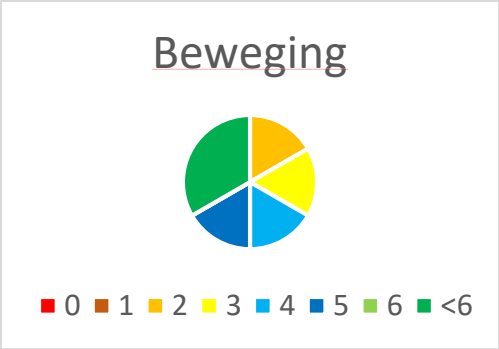


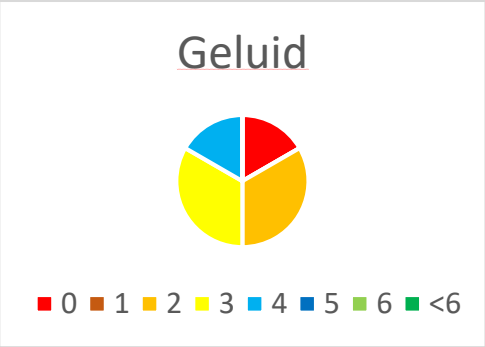


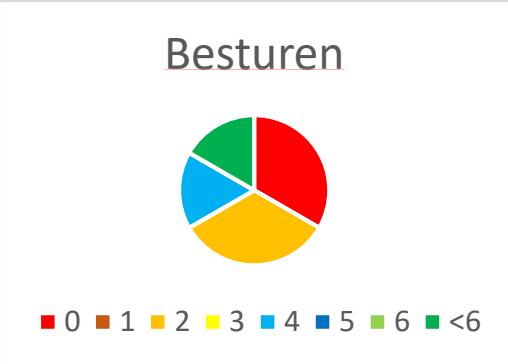


Effectmeting

Aantal keer	**0**	**1**	**2**	**3**	**4**	**5**	**6**	**>6**
Beweging	0	0	1	1	1	1	0	2
Uiterlijk	3	0	0	0	1	1	0	1
Geluid	1	0	2	2	1	0	0	0
Gebeurtenissen	0	1	1	0	1	1	1	1
Besturen	2	0	2	0	1	0	0	1







Bijlage 12 Beoordeling niveau denkvaardigheid (fases van de leerlijn) groep 8
Nulmeting

Aantal leerlingen	Fase 0	Fase 1	Fase 2	Fase 3
Probleem herformuleren	0	0	6	0
Herhaling	4	2	0	0
Algoritme	2	3	0	1
Probleem decompositie	0	3	3	0

Effectmeting

Aantal leerlingen	Fase 0	Fase 1	Fase 2	Fase 3
Probleem herformuleren	0	0	4	2
Herhaling	0	2	0	4
Algoritme	0	0	3	3
Probleem decompositie	0	0	2	4

Bijlage 13 Algemene resultaten drie observaties (cursief staat ook benoemd in de resultaten)
Interventie 1 Microbit
De leerlingen konden na het zien van de introductiefilm zelfstandig verschillende deelstappen maken in een script. *De leerlingen hebben kennis vergaard over herhalingen, maar sommige hebben dit niet gebruikt in hun eigen script. Deze leerlingen kozen ervoor om de alsdan functie te gebruiken (als er werd geschud of op een knop werd gedrukt dan…). Dit was recenter aan bod gekomen.* De andere

leerlingen lieten hem de hele tijd herhalen. De meeste leerlingen hielpen elkaar, gaven elkaar nieuwe inzichten en haalden hun eigen fouten eruit (hoe kan je twee keer dezelfde letter laten verschijnen). *Twee meiden vonden het lastig om hun eigen fouten eruit te halen, waarbij deze leerlingen nog wat extra uitleg vroegen over de functies. Hierdoor snapten ze het wel, maar de reeks werd niet erg complex.* Het leek op de reeks uit het introductiefilmpje. De anderen zorgden voor een complexere reeks door veel verschillende deelstappen te gebruiken. Ze ontdekten zelf de mogelijkheden van het programma. *Het was opvallend dat de twee meiden niet zelf op ontdekkingstocht uitgingen.* Iedereen kon logisch programmeren, omdat ze begonnen met een herhaling of de alsdan functie.

Interventie 2 Codemonster

De leerlingen programmeerden individueel en drie leerlingen waren soms bang om fouten te maken. Hierdoor zochten ze steeds bevestiging bij de leerkracht. Vooral de jongens haalden snel hun eigen fouten eruit door te experimenteren. Dit programmeren was complex, omdat er gebruik werd gemaakt van computercodes, kleuren, transparantie, verschillende groottes en vormen (tekstueel). De meeste leerlingen doorliepen de eerste zes levels: parameters en tekenen, variabelen, kleur, transparantie, operators, alsdan functie en rondingen. Twee leerlingen kwamen twee levels verder, waarin ze zelf een computercode schreven en gebruik maakten van verschillende lijnen. *Opvallend was dat dit twee jongens waren en deze jongens ontdekten veel zelfstandig.* Herhalingen waren pas level twaalf. Door deze levels zat er een logische opbouw in het programmeren. De leerlingen snapten de computercode en konden deze gebruiken (kopiëren en plakken). Bovendien maakten de leerlingen gebruik van herhalingen als ze precies dezelfde kleur en grootte vierkant moesten programmeren.

Interventie 3 bekers

De leerlingen begonnen met een eenvoudig bouwsel, maar vervolgens kozen ze complexere bouwsels met verschillende afstanden, kleuren en hoogtes. De leerlingen bedachten zelf een oplossing voor deze problemen. Voor een halve stap opzij gebruikten ze een halve pijl met 0,5 of ze schreven de H van een halve stap. Voor een andere kleur beker kleurden ze de pijl oranje of schreven de G van groen of de O van oranje erbij. Door twee kleuren bekers te gebruiken, kregen ze twee verschillende startpunten. Het ene groepje had gelijk door dat de bekers meer omhoog moesten en een stapje meer naar de zijkant. Het andere groepje heeft hulp gekregen door te laten zien wat er gebeurd als de robot uitvoerde wat er stond (de beker werd niet hoog genoeg opgetild, waardoor alles omviel). Daarnaast had dit groepje de bekers voor elkaar staan, waardoor de bekers niet op elkaar kwamen, maar voor elkaar. Dit losten ze zelf op. Vervolgens kwamen ze niet ver genoeg en ook dit losten ze zelf op. De leerlingen gaven elkaar aanwijzingen en de bouwer liet gelijk zien wat de programmeur opschreef. Door het zelf uit te proberen, werden veel fouten voorkomen. *Twee leerlingen zochten af en toe nog bevestiging bij de onderzoeker of ze het goed deden. Dit waren weer dezelfde twee meiden als bij interventie één.* De volgorde van bouwen was verschillend. Sommige begonnen dicht bij het startpunt te bouwen en de anderen juist verder weg. Het was niet logisch dat de leerlingen een groot stuk verder begonnen met bouwen. Hierdoor moesten ze veel pijlen extra gebruiken, waardoor er fouten konden ontstaan. Herhaling kwam terug door steeds een pijl naar dezelfde richting op te schrijven.

Volledige uitwerking drie interventies

Interventie 1 Microbit

Leerling 1 en 2

Deze leerlingen snapten het introductiefilmpje en maakten toen echt hun eigen reeks. Ze gebruikten de hele tijd herhalingen of een herhaling als de Microbit heen en weer werd geschut. De leerlingen hadden een lange tijd beide reeksen op hun programmeerveld staan en konden niet hun eigen fouten eruit halen. Hierbij hadden de leerlingen uitleg nodig, want ze snapten niet waarom je het niet tegelijk kan gebruiken (schudden en de hele tijd herhalen). De volgorde snapten de leerlingen,

want ze begonnen een actie als de Microbit werd geschut. Deze reeks was uiteindelijk niet erg complex, want het bestond uit twee deelstappen en drie scripts.

Leerling 3 en 4

De leerlingen snapten alles uit het introductiefilmpje. Ze konden zelfstandig verschillende deelstappen maken in een script. Eén leerling had hier wel meer inzicht in dan de ander. De leerling lieten hun naam zien als ze knop A indrukten en ze gebruikten voor de rest geen herhalingen, zoals de hele tijd. De leerlingen zagen hun eigen fouten en hielpen elkaar om deze te verbeteren. Ook konden de leerlingen logisch programmeren, omdat ze eerst een knop lieten indrukken, voordat er iets mee ging gebeuren. De leerlingen gebruikten vier scripts, maar de complexiteit viel mee. De leerlingen durfden weinig zelf te experimenteren, waardoor het script leek op die van het filmpje.

Leerling 5 en 6

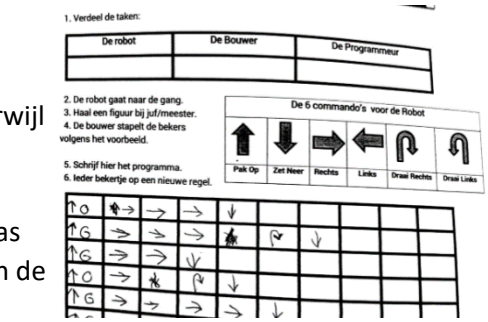
Deze leerlingen wezen elkaar op hun fouten en ze hielpen elkaar om deze fouten te verbeteren. Door elkaar kwamen ze echt tot andere inzichten. Ze hadden met lichtjes hun naam geprogrammeerd, maar één leerling had twee keer achter elkaar de u in zijn naam. Hierdoor kon je niet zien dat er twee keer een u werd getoond. De leerlingen bedachten samen om de u één kolom naar links te programmeren, zodat hij van plaats wisselde. Hierdoor werd hij twee keer getoond. Deze leerlingen losten hun eigen fouten op. De leerlingen snapten het introductiefilmpje en maakten daarna hun eigen reeks met herhalingen (de hele tijd). Ook was deze reeks complex, omdat ze verschillende deelstappen gebruikten. Ook probeerden deze leerlingen er beelden bij te voegen. Ze gingen echt op onderzoek uit. Bovendien gebruikten ze veel scripts. Ze snapten de volgorde van programmeren, omdat ze begonnen met de hele tijd of met wanneer knop A wordt ingedrukt. Ook gebruikten deze leerlingen de alsdan functie. Deze leerlingen wilden maar doorgaan, omdat ze het leuk vonden om alle mogelijkheden te ontdekken.

Interventie 2 Codemonster

De leerlingen hebben individueel achter de computer gezeten. Over het algemeen waren de leerlingen op het begin bang om fouten te maken en zochten ze steeds bevestiging bij de onderzoeker. Ook vroegen ze veel kleine vragen, zoals moeten de vierkanten elkaar aanraken of een groter gedeelte bedekken. De jongens haalden snel hun eigen fouten eruit en twee van de drie meiden vroegen eerst de onderzoeker om hulp. Het andere meisje was zelfstandig en leerde van haar eigen fouten. Deze opdrachten waren complexer, omdat er werd gewerkt met codes, kleuren, transparantie, grootte en meerdere vormen. Echter beleefden alle leerlingen hierdoor meer succeservaringen. Herhaling kwamen pas aan bod bij level twaalf. De meesten hebben de eerste zes levels doorlopen: parameters en tekenen, variabelen, kleur, transparantie, operators, als dat en rondingen. Twee leerlingen waren twee levels verder gekomen, waarin ze zelf een code schreven en gebruikmaakten van verschillende lijnen. De leerlingen moesten bijvoorbeeld van meerdere rechte lijnen een driehoek maken. Doordat de leerlingen steeds in een volgend level kwamen, kwam er een logische volgorde in het programmeren. De leerlingen pakten dit snel op en ze leerden elk level meer. Ze hadden door dat er codes waren voor de grootte en de plaats en ze namen deze snel over. Als het programma vroeg om dezelfde code uit te voeren, gingen de leerlingen dit uit zichzelf kopiëren en plakken. Dit ging sneller en foutloos (herhalingen). Bovendien maakten de leerlingen gebruik van herhalingen als ze precies dezelfde kleur en grootte vierkant moesten programmeren.

Interventie 3 eigen robot met bekers

Leerling 1, 3 en 5



Deze leerlingen gebruikten de volgende kaarten van groep 7/8: één, twee, vijf en negen. De complexiteit steeg door de verschillende afstanden, kleuren, hoogtes en een ander beginpunt. Op het begin vroegen de leerlingen steeds aan de onderzoeker of het wel kon, terwijl de leerlingen het goed deden. Ze gaven elkaar aanwijzingen door te zeggen dat twee pijltjes naar links even ver was als dat ze net voor hadden gedaan met een reeks erboven. De volgorde van bouwen was verschillend. De ene begon dichtbij en bouwde steeds verder weg en de ander deed dit precies andersom. Wel bouwden ze allemaal eerst de onderkant en dan de bovenkant. De leerlingen tekenden een halve pijl met 0,5 en anderhalve pijl met 1,5. De complexiteit steeg, omdat de afstand veranderde. Dit ging wel gelijk goed. Daarnaast bedachten de leerlingen ook een oplossing voor de gekleurde bekertjes. Als de robot een andere kleur beker moest pakken, dan kleurden ze die pijl geel. Deze oranje bekers stonden achter het startpunt van de zwarte bekers. De leerlingen dachten er gelijk aan dat de beker omhoog moest en je twee keer naar links moest, voor je het startpunt bereikte. Hier kwam dus ook de herhaling aan bod. Doordat de bouwer gelijk uitvoerde wat de programmeur opschreef, ontstonden er weinig fouten. Na deze bouwsels hadden de leerlingen het heel goed onder de knie. De volgende keer zou ik ze meer uitdagen door meer commando's te geven en de kaarten voor groep 7/8 aan te passen.

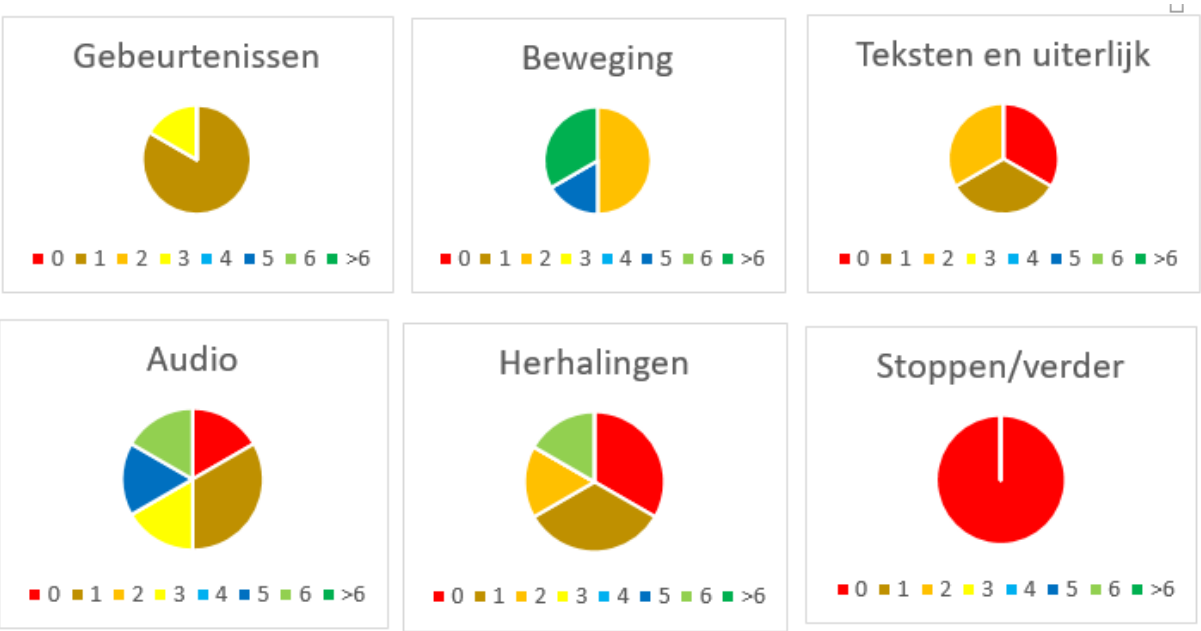
Leerling 2, 4 en 6

De leerlingen bouwden tegelijkertijd met het programmeren, zodat ze wisten dat er geen fouten in de reeks zaten. Eerst pakten ze alleen de beker op en draaiden ze de beker om zijn as. De leerlingen maakten geen gebruik van recht en links. Daarna zei één leerling er wat van en gaf de onderzoeker de leerlingen wat tips. Hierdoor konden ze gelijk aan de slag. De leerlingen bouwden niet in een logische volgorde, omdat ze van ver weg naar dichtbij bouwden. Het was ook niet logisch, omdat de leerlingen de robot lieten bouwen een heel stuk van het beginpunt af. De robot moest eerst drie stappen naar links zetten, waardoor de kans op fouten groter werd. Op het begin wilden de leerlingen van boven naar beneden bouwen. Dit was niet logisch. Een leerling gaf aan dat het makkelijker was om van beneden naar boven te bouwen. De leerlingen wilden als eerste voorbeeld twee bouwen, maar kozen uiteindelijk voor een makkelijkere optie, namelijk bouwwerk één. Toen de robot de reeks ging uitproberen, kwamen ze erachter dat de robot het bekertje niet neer kon zetten. Ze verbeterden zelf hun fout. De complexiteit steeg, omdat ze steeds een moeilijker bouwwerk kozen. Eerst bouwwerk één, vervolgens bouwwerk zeven en daarna bouwwerk acht. Hierbij moesten ze denken aan een ander beginpunt en aan de verschillende kleuren bekertjes. Hierbij hadden de leerling een oplossing bedacht voor de verschillende kleuren bekertjes. Bij het eerste blokje schreven ze een G van groen of een O van oranje. Daarnaast tekenden ze een halve pijl voor een halve stap naar links, maar dit vonden ze niet duidelijk. Uiteindelijk hebben ze een H geschreven van half. De robot snapten dit. De groene bekers stonden voor en de oranje bekers stonden achter, hierdoor kwamen de bekers niet op elkaar. Ze kwamen echter voor elkaar. Ze verbeterden hun fout en zetten het startpunt van de bekers naast elkaar. Vervolgens kwamen de bekers niet ver genoeg, omdat de verste beker nu een stapje extra moest. Ze verbeterden zelf hun reeks door er een pijl tussen te schrijven voor een extra stap. Hierbij heeft de onderzoeker de leerlingen wel geholpen voor

verbetering. Bijvoorbeeld door te laten zien wat een robot zou doen bij die reeks, waardoor de bekers niet hoog genoeg kwamen en tegen de eerste rij kwamen. Hierdoor vielen alle bekers om. De leerling bedachten zelf een oplossing door een extra pijl omhoog te schrijven. Herhaling kwam terug door steeds een pijl naar links te schrijven. De leerlingen twijfelden aan het begin of ze het wel goed deden, maar aan het einde lukte het om het moeilijkste bouwwerk te maken.

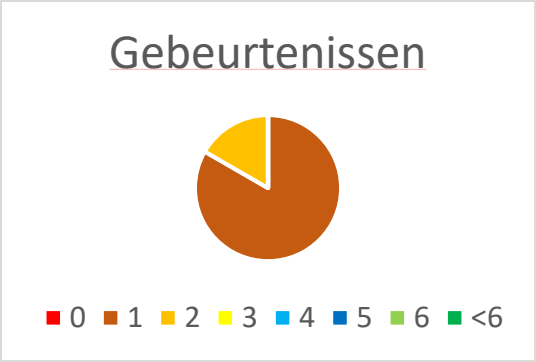
Bijlage 14 beoordeling aantal scripts groep 3
Nulmeting

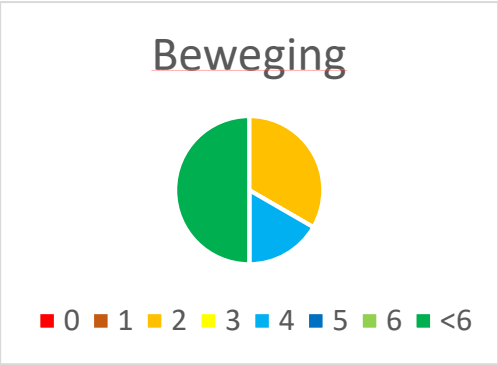
Aantal keer	**0**	**1**	**2**	**3**	**4**	**5**	**6**	**>6**
Gebeurtenis	0	5	0	1	0	0	0	0
Beweging	0	0	3	0	0	1	0	2
Teksten en uiterlijk	2	2	2	0	0	0	0	0
Audio	1	2	0	1	0	1	1	0
Herhalingen	3	1	0	1	0	0	1	0
Stoppen	6	0	0	0	0	0	0	0

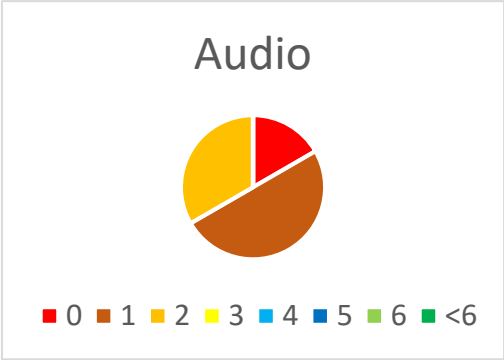


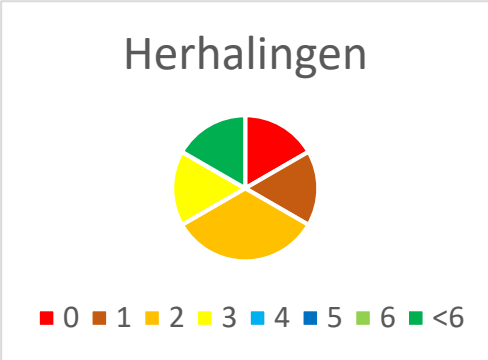
Effectmeting

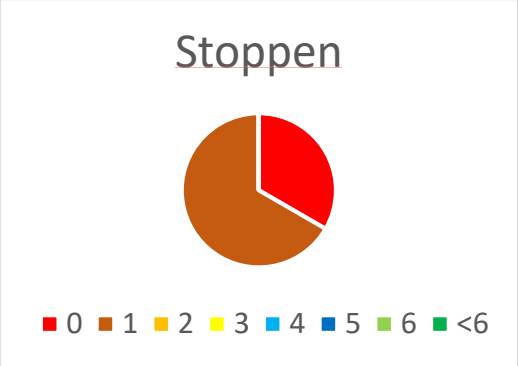
Aantal keer	**0**	**1**	**2**	**3**	**4**	**5**	**6**	**>6**
Gebeurtenis	0	5	1	0	0	0	0	0
Beweging	0	0	2	0	1	0	0	3
Teksten en uiterlijk	1	1	1	2	1	0	0	0
Audio	1	3	2	0	0	0	0	0
Herhalingen	1	1	2	1	0	0	0	1
Stoppen	2	4	0	0	0	0	0	0











Bijlage 15 Beoordeling niveau denkvaardigheid (fases van de leerlijn) groep 3
Nulmeting

Aantal leerlingen	Fase 0	Fase 1	Fase 2	Fase 3
Probleem herformuleren	0	0	6	0
Herhaling	5	1	0	0
Algoritmes	3	3	0	0
Probleem decompositie	5	1	0	0

Effectmeting

Aantal leerlingen	Fase 0	Fase 1	Fase 2	Fase 3
Probleem herformuleren	0	0	4	2
Herhaling	0	0	1	5
Algoritmes	1	1	4	0
Probleem decompositie	0	0	5	1

Bijlage 16 Algemene resultaten drie observaties groep 3
Interventie 1 Codewise
Alle leerlingen vonden het lastig om met logische stappen te programmeren, want ze gingen eerst naar rechts en gelijk weer naar links of eerst staan en dan pas zitten. Door aanwijzingen van de leerkracht kregen ze hier meer inzicht in. Er zat wel een verschil tussen de niveaus van de leerlingen. De ene leerling had aan het einde van de les meer inzicht dan de ander. Op het begin waren het korte reeksen. De complexiteit steeg door het gebruik van de gele, oranje en paarse kaarten. *De leerlingen vonden het uitdagend om deze te gebruiken. Hierdoor gebruikten ze herhalingen (voor de*

hele reeks of kleine stukken), de alsdan functie en verschillende snelheden voor bewegingen. Het werden complexere reeksen met veel verschillende deelstappen. Alle leerlingen gaven elkaar aanwijzingen als er fouten werden gemaakt, waardoor ze samen hun fouten eruit konden halen. *Twee leerlingen waren op het begin bang om fouten te maken, maar ervaarden dat fouten maken niet erg is. Dit lieten ze snel los. Dit was een jongen en een meisje.*

Interventie 2 Blue-Bot

Op het begin konden de leerlingen het niet onthouden dat ze de robot leeg moesten maken. Echter na een paar keer uit te leggen waarom ze het moesten doen, werd dit een gewoonte voor de leerlingen. De leerlingen vonden het lastig om de robot een andere kant op te laten lopen, omdat ze het verschil niet wisten tussen rechtsom en linksom draaien. Daarnaast dachten ze dat de robot zou lopen als ze hem lieten draaien, maar ze moesten de robot ook nog rechtdoor laten lopen. *De leerlingen maakten hier eerst fouten in. Door het veel te proberen, haalden ze hun eigen fouten eruit.* Een ander groepje verbeterde een fout (rechtsom draaien) door twee keer op linksom te drukken. Zij begonnen geen nieuwe reeks. Een ander groepje liet de robot achteruitlopen, waardoor het draaien zo min mogelijk werd toegepast. Ze hielpen elkaar. De ene leerling drukte de pijltjes in en de ander wees de letter aan waar de robot zou staan. De meeste leerlingen dachten logisch na, door de kortste route te nemen. Alle leerlingen reden eerst naar een letter en uiteindelijk nam de complexiteit toe, omdat ze in een keer (een gedeelte van) hun naam konden programmeren. Dit was complex, omdat het veel deelstappen waren.

Interventie 3 bekers

Eén leerling zat op de gang. *De andere leerlingen testen hun bouwsel door zelf de rol van een robot in te nemen. Hierdoor werden er bijna geen fouten gemaakt of ze verbeterden hun fouten.* De leerlingen bouwden in een logische volgorde van dichtbij het startpunt naar verder weg. Soms programmeerden ze minder logisch door van boven naar beneden te programmeren. *Door het zelf uit te voeren met behulp van blokken, kwamen ze erachter dat het handiger kon.* De leerlingen gingen van eenvoudige, naar steeds complexere bouwwerken. Hierbij hielden ze rekening met verschillende hoogtes door twee pijltjes omhoog op te schrijven. Het werd ook complexer, omdat er meer gedraaid moest worden en er meer bekertjes gebruikt werden. Herhalingen kwamen alleen aan bod als ze dezelfde pijl meerdere keren gebruikten.

Volledige uitwerking drie interventies groep 3

Interventie 1 Codewise kaarten

Leerling 1 en 2

Ze begonnen niet logisch te programmeren, zoals pijltje heen en dan weer terug. Zo bleven ze op dezelfde plek staan. De leerlingen hebben alle kaarten gebruikt in opbouw van moeilijkheid, namelijk van geel, naar geel en oranje en ten slotte ook paars. De leerlingen gebruikten de moeilijke kaart van herhalen, zoals de hele reeks twee keer herhalen of een deel van de reeks vier keer herhalen. De onderzoeker heeft de leerlingen uitgelegd dat je ook eindeloos kan laten herhalen. De leerlingen zien fouten door aanwijzingen van hun maatje. Dit tweetal heeft veel kaarten en veel verschillende deelstappen gebruikt. De reeks van hen was complex.

Leerling 3 en 4

De leerlingen snapten niet hoe ze logisch moesten programmeren. Ze hadden bijvoorbeeld eerst opstaan en dan zitten en vervolgens moesten alle stappen zittend worden uitgevoerd. Hierbij heeft de onderzoeker de leerlingen laten zien wat logischer was, namelijk eerst zitten en dan weer staan. Ook had dit tweetal niet logisch geprogrammeerd bij het lopen (ook het pijltje heen en weer terug, waardoor hij op dezelfde plek bleef staan). Hierbij heeft de onderzoeker aanwijzingen gegeven. De leerlingen lieten hem ook twee keer draaien, waardoor hij omgedraaid was. Eén leerling was bang om fouten te maken. Na uitleg van de herhalingskaart snapten ze het gelijk. De leerlingen konden een deel van de reeks of de hele reeks laten herhalen. Ze pakten de kaart van het herhalen er zelf bij.

Aan het einde bleven de leerlingen sommige stappen niet logisch neerzetten, zoals een hoge snelheid bij het zitten. Ook deze leerlingen wezen elkaar op hun fouten en ze konden beide hun fouten corrigeren. Uiteindelijk was deze reeks ook complex, omdat ze verschillende deelstappen gebruikten. Ze gebruikten bijvoorbeeld verschillende snelheden, de alsdan functie en herhalingen. Deze leerlingen gingen aan het eind logischer programmeren, zoals een boek neerleggen en weer oppakken. De ene leerling had hier wel meer inzicht in dan de ander. Die was ook bang om het fout te doen.

Leerling 5 en 6

Ook deze leerlingen snapten niet hoe ze logisch moesten programmeren. Ook deze groep had het pijltje heen en dan weer terug, zodat ze op dezelfde plek bleven staan. Bovendien ging dit tweetal eerst staan en dan zitten, terwijl ze al stonden. Hier heeft de onderzoeker voorbeelden bij gegeven. Op het begin maakten ze korte reeksen zonder enige complexiteit. De complexiteit werd gestimuleerd door het gebruik van de gele, oranje en paarse kaarten. Hierbij vonden de leerlingen het uitdagend om deze te gebruiken, waardoor ze snel de kaarten voor herhalen en de alsdan functie gebruikten. Ze konden ook de hele reeks of een deel van de reeks meerdere keren laten herhalen. Wel zat er een verschil in het niveau van de leerlingen, want de ene leerling snapte de snelheid en de alsdan functie beter dan de andere. Deze leerling haalde ook snel haar eigen fouten eruit. Dit tweetal ervaarden dat ze fouten mogen maken en dat ze die mochten veranderen. Dit was voor hun heel leerzaam. De laatste reeks maakten ze samen en deze werd erg lang. De complexiteit nam toe, omdat ze veel verschillende kleuren gebruikten.

Interventie 2 Blue-bot

Leerling 1 en 2

Op het begin vonden de leerlingen het lastig om de robot leeg te maken, maar naar verloop van tijd werd dit een gewoonte. Daarnaast dachten de leerlingen dat de robot naar de zijkant ging lopen als er op draaien werd gedrukt. Echter draait de robot dan alleen en verzet hij nog geen stap. De leerlingen maakten hier foutjes in, maar bleven proberen. Deze leerlingen toonden doorzettingsvermogen. Ze hielpen elkaar door met de vinger aan te wijzen waar de robot nu zou staan (de ander drukte de pijltjes in). Eerst deden ze van letter naar letter rijden en ten slotte konden ze in één keer naar hun hele naam rijden. Dit was complex, omdat het veel stappen achter elkaar waren (vijf letters in een naam). De ene leerling toonde wel meer inzicht dan de ander. De leerlingen vonden het lastig om te weten welke kant de robot op zou draaien (rechtsom of linksom). Hier kwamen de leerlingen achter door te testen. Het was mooi om te zien dat de leerlingen het daarna nog een keer fout deden (rechtsom) en vervolgens twee keer op linksom drukten, zodat hij uiteindelijk naar links reed. Ze herstelden hun eigen fouten. Dit gebeurde ook bij voor- en achteruit. De leerlingen snapten de herhalingen (meerdere keren drukken op rechtdoor). De leerlingen dachten logisch na en ze probeerden de kortste route te nemen naar al hun letters.

Leerling 3 en 4

Deze leerling ervaarden dat ze mochten blijven proberen en dat fouten maken niet erg is, zoals het kiezen van de verkeerde draairichting. De leerlingen bleven dit testen. Daarnaast dachten de leerlingen dat de robot naar de zijkant ging lopen als er op draaien werd gedrukt. Echter draait de robot dan alleen en verzet hij nog geen stap. De leerlingen maakten hier fouten in, maar bleven proberen. Beide leerlingen bedachten een logische volgorde om van de ene naar de andere letter te rijden. Bovendien wezen de leerlingen op elkaars fouten, zoals een verkeerde draairichting. Het was mooi om te zien dat ze elkaar aanwijzingen gaven. Eén leerling snapte er eerst niets van en nu kon hij zijn hele naam programmeren. Af en toe vergat hij nog het kruisje in te drukken, maar door het gewoon uit te voeren, drukten ze alsnog op het kruisje. De andere leerling had veel inzicht en schreef ook zijn naam. Op het einde begonnen ze te spelen met de robot en lieten ze de robot honderd

rondjes draaien door op de draairichting te drukken. De leerlingen snapten wat herhaling was.
Leerling 5 en 6
Eén leerling dacht niet logisch na, omdat ze met een hele omweg naar haar eerste letter reed. De andere leerling had hier meer inzicht in, want die nam steeds de kortste route. Daarnaast dachten beide leerlingen dat de robot naar de zijkant ging lopen als er op draaien werd gedrukt. Echter draait de robot dan alleen en verzet hij nog geen stap. De leerlingen maakten hier fouten in, maar bleven proberen totdat het lukte. De leerlingen hadden alleen geen inzicht in de draairichting, waardoor hij steeds de verkeerde kant op liep. De leerlingen bedachten hier een slimme oplossing voor, door de robot zo min mogelijk te laten draaien, maar meer achteruit te laten rijden. Het was moeilijk om meerdere deelstappen achter elkaar te laten uitvoeren, maar naar één letter rijden ging goed. Doordat de leerlingen minder inzicht hadden, pakten ze regelmatig de robot op om te kijken waar hij zou staan en welke kant hij op zou staan. Ze konden niet hun fout herstellen in de reeks, maar verwijderde de hele reeks en deden het opnieuw. De leerlingen waren enorm enthousiast als het was gelukt om naar de goede letter te rijden. Ze wilden de robot zelfs hebben.

Interventie 3 eigen robot met bekers
Leerling 1, 2 en 3
Eén leerling zat op de gang. De andere leerlingen testen hun bouwsel door zelf de rol van een robot in te nemen. Hierdoor werden er bijna geen fouten gemaakt. Als er een fout werd gemaakt, zagen ze zelf waar dat aan lag en konden ze hun eigen fout verbeteren. Hierbij was geen hulp van de onderzoeker nodig. Het was een logische volgorde, omdat ze begonnen met bouwen dicht bij het startpunt (dichtstbijzijnde beker en steeds verder weg). Bovendien zorgden de leerlingen ervoor dat de bekertjes eerst op de goede plek stonden en zorgden er dan voor dat het bekertje eventueel gedraaid werd. De leerlingen gingen van eenvoudig naar steeds complexere bouwwerken van de kaarten. Ze kozen de volgende bouwwerken van groep 3/4: één, twee, vier en acht. De leerlingen tekenden twee pijltjes naar boven, zodat hij nog een plek omhoogging. Hierdoor viel de toren niet om. Het werd ook complexer, omdat er meer gedraaid moest worden en er meer bekers gebruikt werden. Hierbij maakten ze een fout, maar ze verbeterden zichzelf. Herhalingen kwamen naar voren door meerdere keren een pijl dezelfde richting op te schrijven.
Leerling 4, 5 en 6
De leerlingen hebben in de volgende volgorde de kaarten van groep 3/4 gebruikt: één, twee, zeven en acht. Hierdoor nam de complexiteit steeds toe. Als de leerlingen een fout maakten, lieten ze de pijl wel staan. Ze zagen dat ze een fout maakten, maar haalden deze dus niet weg. Echter begonnen ze eronder een nieuwe reeks, waarin ze hun eigen fouten verbeterden. Deze leerlingen kwamen achter hun fouten als de bouwer precies deed wat de programmeur opschreef. Dit was dus voor de uitvoering. Herhaling gebeurden hier ook door meerdere keren een pijl in dezelfde richting te schrijven. De leerlingen deden het in een logische volgorde, omdat ze van dichtbij naar ver weg bouwden. Echter waren sommige gebeurtenissen niet logisch. De leerlingen programmeerden eerst van boven naar onder. Door het zelf te testen, kwamen ze erachter dat dit niet logisch was. Ze kwamen niet tot een bouwwerk, want er viel van alles om. Ze hebben geleerd van hun fouten. Beide groepjes vonden het leuk en wilden met zijn drieën met de onderzoeker als robot programmeren.