ONDERZOEK

Hoe kinderen computational thinking skills inzetten als zij een probleem oplossen met de Ozobot



LECTORAAT ONDERWIJSINNOVATIE & ICT

ZWOLLE, 2017

UNIVERSITEIT VAN AMSTERDAM

Windesheim

Auteurs	:	Diane van der Linde, Nicole van Aar, Hogeschool Windesheim Prof. Dr. Joke Voogt, Hogeschool Windesheim/Universiteit van Amsterdam
Studenten	:	Kelly Bakkenes, Roos van Goor, Tony Gruntjes, Inge Hollak, Josien Hulleman, Myrthe Spanjaard, Pascal Stoel, Hogeschool Windesheim
In opdracht van	:	Nationaal Regieorgaan Onderwijsonderzoek (NRO)
Lectoraat	:	Onderwijsinnovatie & ICT, Windesheim Zwolle
Website	:	www.windesheim.nl

©Copyright, 2017: lectoraat Onderwijsinnovatie & ICT, Christelijke Hogeschool Windesheim

Inhoud

1. Samenvatting 3

2. Inleiding 4

Onderzoeksvragen 6

3. Methode 7

Opzet 7
Ozobot 7
Deelnemers 7
Observatie-instrument 8
Procedure 11
Data-analyse 13

4. Resultaten 14

4.1 Kwantitatieve resultaten 14

Algemeen 14
Probleemdecompositie, abstraheren, testen en gegevens ordenen 15
Fouten opsporen en oplossen 18

4.2 Illustratieve casussen 20

Isa en Mette uit groep 7 20
Fred en Gijs uit groep 7 21
Ronald en Inge uit groep 8 22
Gavin en Doortje uit groep 6 23
Ella en Brenda groep 6 23
Mirjam en Joris uit groep 5 24

5. Conclusie en discussie 26

Referenties 29

Bijlage 1 31

Bijlage 2 34

1. Samenvatting

Computational thinking skills zijn denkvaardigheden die je kunt inzetten om problemen op te lossen met behulp van computertechnologie. In dit onderzoek is nagegaan welke denkvaardigheden leerlingen tussen 6 en 12 jaar gebruiken als zij in tweetallen een Ozobot (een kleine robot) programmeren om een gegeven probleem (de simpele programmeertaak) op te lossen. Tweeënveertig duo's uit de groepen 4 tot en met 8 hebben de programmeertaak uitgevoerd, waarbij ze tijdens het maken van de programmeertaak hardop uitspraken wat ze dachten. Het werken van de duo's aan de programmeertaak is op film vastgelegd. Vijf deelvaardigheden van Computational Thinking skills kwamen in het kader van de programmeertaak aan de orde: probleemdecompositie, abstraheren, ordenen van gegevens, testen en fouten opsporen en oplossen. De data zijn geanalyseerd op deze vijf deelvaardigheden. Uit de analyse van de data bleek dat iets minder dan de helft van de duo's de taak succesvol oploste. Dit waren vooral de duo's uit de hogere groepen. Duo's van jongens uit de lagere groepen scoorden gemiddeld hoger op de onderscheiden deelvaardigheden dan meisjes duo's en gemengde duo's uit de lagere groepen. In de hogere groepen was er geen verschil in gemiddelde score tussen jongens -, meisjes - en gemengde duo's. Voor 3 van de 5 deelvaardigheden (probleemdecompositie, abstraheren, testen) is de grootste verbetering in de aanpak zichtbaar tussen het plaatsen van code 1 en code 2. Alleen voor gegevens ordenen is het verschil pas significant tussen code 2 en 3. Bij fouten opsporen en oplossen is dit niet per code te beoordelen. Succesvolle duo's werken gestructureerd, zij verzamelen alle informatie voordat ze beginnen, elimineren opties, nemen de tijd om de (mogelijke) oplossing door te nemen en werken nauwkeurig.

2. Inleiding

In 2006 bracht Janet Wing de term Computational Thinking onder de aandacht. Volgens Wing gaat het bij Computational Thinking om “het oplossen van problemen, het ontwerpen van systemen en het begrijpen van menselijk gedrag met gebruikmaking van concepten die fundamenteel zijn in de informatica” (Wing 2006, p33). Computational Thinking is een universele attitude en vaardigheid die, volgens Wing, iedereen, en niet alleen informatici, moet ontwikkelen en gebruiken. Als het gaat over de focus van Computational Thinking in de context van het funderend onderwijs dan ligt de nadruk op de vaardigheden en disposities die nodig zijn om complexe problemen op te lossen, met behulp van de computer (Barr & Stephenson, 2011; Grover & Pea, 2013; Lee et al., 2011). In Nederland werkt de SLO in opdracht van het Ministerie van Onderwijs Cultuur en Wetenschappen aan de uitwerking van een leerplankader voor de ontwikkeling van computational thinking skills in het funderend onderwijs. Gebaseerd op het werk van de International Society of Technology in Education (ISTE) (nd) (in samenwerking met de Computer Science Teacher Association (CSTA)) hanteert de SLO (nd) de volgende definitie van Computational Thinking Skills: ”Computational thinking is het procesmatig (her)formuleren van problemen op een zodanige manier dat het mogelijk wordt om met computertechnologie het probleem op te lossen. Het gaat daarbij om een verzameling van denkprocessen waarbij probleemformulering, gegevensorganisatie, -analyse en -representatie worden gebruikt voor het oplossen van problemen met behulp van ICT-technieken en gereedschappen.” Vanuit deze definitie kan CT bijdragen aan het verwerven van vaardigheden die nodig zijn bij het oplossen van problemen. De onderwerpen die bij CT aan de orde komen zijn volgens SLO en ISTE/CSTA: probleemdecompositie, gegevens verzamelen, analyseren en visualiseren, abstraheren, algoritmes en procedures, automatiseren, simuleren en modelleren en de gelijktijdige uitvoering van taken. In de benadering van de SLO en ISTE gaat het overigens niet alleen om kennis en vaardigheden, maar ook om een attitude gericht op het om kunnen gaan met complexiteit en ambiguïteit, doorzettingsvermogen bij het oplossen van lastige problemen en samenwerken met anderen bij het zoeken naar de oplossing. Men is het erover eens dat aspecten van CT in verschillende leerdomeinen aan de orde kunnen komen, zoals bijvoorbeeld in het stelonderwijs en de natuurwetenschappen (Sengupta, Kinnebrew, Basu, Biswas & Clark, 2013; Wolz, Stone, Pearson, Pulimood, & Switzer, 2011).

Toch worden Computational Thinking skills vaak in verband gebracht met, of soms zelfs gelijk gesteld aan programmeervaardigheden. Een belangrijke reden is dat veel onderzoek naar Computational Thinking is uitgevoerd in de context van programmeren bijv. (Hambrusch, Hoffmann, Korb, Haugan, & Hosking, 2009; Lee et al., 2011; Lu & Fletcher, 2009). Ook Papert (1980) legde deze relatie. In zijn boek “Mindstorms: Children, Computers, and Powerful Ideas” beargumenteert hij dat kinderen door te programmeren in met de speciaal voor jonge kinderen ontwikkelde programmeertaal LOGO denkvaardigheden ontwikkelen die overdraagbaar zijn naar andere contexten. Ook in recente populaire initiatieven wordt het verband gelegd

tussen programmeervaardigheden en Computational Thinking Skills (bijv. http://www.codekinderen.nl/leerling/programmeren/hour-of-code/). Onderzoek naar deze assumptie is vooral uitgevoerd rond 1980 toen relatief veel onderzoek werd gedaan met LOGO. Dat onderzoek laat echter geen eenduidige resultaten zien (bijv. (Klahr & Carver, 1988; Pea, Kurland, & Hawkins, 1985). Salomon en Perkins, (1989) geven als mogelijke verklaring voor deze bevindingen dat de overdraagbaarheid van vaardigheden mede afhankelijk is van de instructie, waarin het werken met LOGO is ingebed. Voogt, Brand-Gruwel en van Strien (2017) lieten op basis van een recente review van de literatuur zien dat programmeeronderwijs wel degelijk kan bijdragen aan generieke probleemoplosvaardigheden, evenals aan inzicht in de begrippen en vaardigheden die van belang zijn om een computer te programmeren. De begrippen en vaardigheden die al op jonge leeftijd kunnen worden aangeleerd zijn bijvoorbeeld *control flow* – de vaardigheid om instructies in een bepaalde volgorde uit te laten voeren of *loop* - het herhaaldelijk uitvoeren van instructies onder bepaalde condities en het *debuggen* – het opsporen van fouten. Wel liet deze review zien dat het onderzoek naar het effect van programmeeronderwijs op CT nauwelijks heeft plaatsgevonden in een reguliere onderwijscontext.

Lye, Hwee en Koh (2014) hebben een reviewstudie verricht naar het ontwikkelen van Computational Thinking skills door middel van programmeertools. Zij onderscheiden computational concepts, waar de kennis over CT centraal staat, computational practices dat zich richt op het aanleren van de denkvaardigheden die belangrijk zijn, en computational perspectives dat de attitude component van CT benadrukt. Het zijn het vooral de computational practices die volgens Lye et al. (2014) met behulp van programmeertools kunnen worden ontwikkeld. Computational practices zijn volgens Lye et al. (2014) “problem-solving practices that occur in the process of programming” (p.53), zoals het opsplitsen van het probleem in deelproblemen, het gebruiken van loops, iteraties, testen en debuggen van (deel)oplossingen, hergebruik van algoritmes en abstraheren. Ook Lye et al. (2014) leggen dus een verband tussen CT en strategieën die kunnen worden gebruikt om problemen op te lossen. Uit de review blijkt dat er weinig recente studies beschikbaar zijn die de relatie tussen computational thinking skills en programmeertools in het funderend onderwijs onderzoeken, maar dat uit het beperkte onderzoek dat beschikbaar is aanwijzingen naar voren komen dat de visuele output van de tool helpt bij het testen van de oplossing. Ook waren er aanwijzingen voor het belang van reflectie op de oplossing voor het ontwikkelen van probleemoplosvaardigheden met behulp van programmeertools. Om meer inzicht te krijgen in de denkvaardigheden die kinderen inzetten als zij werken aan programmeertaken om problemen op te lossen bevelen Lye et al. (2014) aan om studies uit voeren waarin via hardop denken het denkproces van kinderen wordt onderzocht als zij werken aan het oplossen van problemen waaraan zij werken met programmeertools. Het onderzoek waarover we rapporteren sluit aan bij deze aanbeveling.

In navolging van Lye et al. (2014) en SLO/ISTE zien we CT als de denkvaardigheden die je kunt inzetten om problemen op te lossen met behulp van computertechnologie.

Deze denkvaardigheden bestaan uit verschillende deelvaardigheden. In dit onderzoek kijken we met name naar de deelvaardigheden:

- probleemdecompositie- analyseren van het probleem in deeltaken,
- abstraheren – vaststellen wat de kern van het probleem is,
- ordenen en met elkaar in verband brengen van gegevens,
- testen of de instructies op de gewenste manier worden uitgevoerd,
- fouten opsporen en oplossen.

We gebruiken de Ozobot (een kleine robot die werkt op basis van kleurencodes) om inzicht te krijgen in het handelen en de denkprocessen van kinderen in de leeftijd van 6 en 12 jaar als zij samen een probleem oplossen waarbij zij gebruik maken van de mogelijkheden van de Ozobot (http://ozobot.com/). In het probleem komen met name de hierboven benoemde deelvaardigheden aan de orde.

Onderzoeksvragen

De onderzoeksvraag voor deze studie luidt:
Welke denkvaardigheden gebruiken leerlingen tussen 6 en 12 jaar uit als zij in tweetallen de Ozobot programmeren om een gegeven probleem (de programmeertaak) op te lossen?

De hoofdvraag beantwoorden we met behulp van de volgende deelvragen:

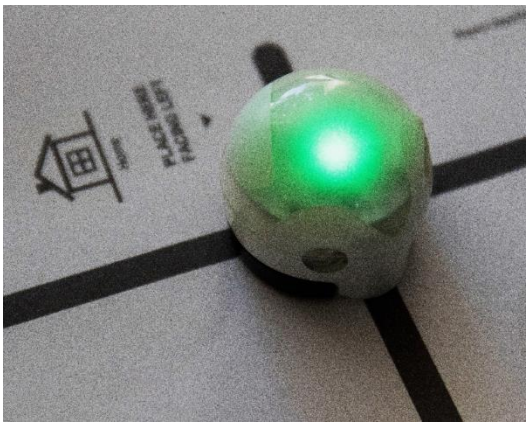
1. Welke activiteiten voeren de tweetallen uit als zij de Ozobot programmeren om het probleem op te lossen en leiden de activiteiten tot een oplossing van het probleem?
2. Welke overwegingen geven zij bij de uitvoering van deze activiteiten?
3. Hoe kan de kwaliteit van de gehanteerde aanpak voor de oplossing van het probleem worden geduid in termen van computational thinking skills?
4. Maakt leeftijd uit in de wijze waarop het tweetal de activiteiten uitvoert en de overwegingen die ze hebben? En zo ja, in welk opzicht?
5. Maakt groepssamenstelling (jongensgroepjes, meisjesgroepjes, gemengde groepjes) uit in de wijze waarop het tweetal de activiteiten uitvoert en de overwegingen die zij hebben. En zo ja, in welk opzicht?

3. Methode

Opzet

Er is een observatiestudie uitgevoerd bij leerlingen in het basisonderwijs. Om het hardop denken van leerlingen te bevorderen voeren zij de programmeertaak uit in tweetallen van gemengde samenstelling. De observatiedata zijn kwalitatief en kwantitatief geanalyseerd.

Ozobot



De Ozobot ((http://ozobot.com/) is een kleine robot welke programmeerbaar is door lijnen en/of kleurcodes binnen de lijnen te tekenen. De Ozobot rijdt over een lijn en voert de kleurcode die hij hierbij tegenkomt uit. Een kleurcode bestaat uit een combinatie van kleuren (groen, blauw, zwart of rood) en leidt tot een actie, zoals bijvoorbeeld het aanpassen van snelheid of richting of het maken van leuke bewegingen (zie de voorbeeldcodes – Figuur 1).

Een kleurcode onderbreekt als het ware de lijn. Na de kleurcode loopt de lijn weer verder. Slordig inkleuren van de codes leidt tot fouten, zoals ook slordig programmeren tot fouten leidt.

Het lampje van de Ozobot toont de kleur, welke op dat moment herkend wordt. Wanneer de Ozobot over een kleurcode rijdt, dan worden de kleuren van de kleurcode achtereenvolgens getoond. De Ozobot kan gebruikt worden om op een leeg vel vrij lijnen en codes te tekenen, bij het plaatsen van codes in gedefinieerde opdrachten en met een app op een tablet.



Figuur 1 Voorbeeldcodes Ozobot

Deelnemers

Aan het onderzoek hebben 84 leerlingen deelgenomen uit de groepen 4 tot en met groep 8 van diverse basisscholen in de regio rondom Zwolle. De leeftijd van de leerlingen varieerde tussen 6 en 12 jaar, met een gemiddelde leeftijd van 9.2 jaar. De

leerlingen hebben in tweetallen (42 paren) gewerkt aan de programmeertaak. De tweetallen zijn samengesteld door de onderzoeksassistent op basis van leeftijd en sekse (jongensgroepen, meisjesgroepen, gemengde samenstelling). Zie Tabel 1. In het onderzoek is de variabele 'groep' gehanteerd als een indicator voor leeftijd. De leerlingen hebben geen ervaring met programmeertools en kennen de Ozobot niet.

duo's	groep					*totaal*
	4	*5*	*6*	*7*	*8*	
jongen-jongen	0	2	4	3	2	*11*
meisje-jongen	2	2	4	4	3	*15*
meisje- meisje	4	2	5	3	2	*16*
totaal	*6*	*6*	*13*	*10*	*7*	*42*

Tabel 1: Samenstelling duo's per groep (aantal)

Observatie-instrument

CT definiëren we als denkvaardigheden die nodig zijn bij het oplossen van problemen met behulp van computertechnologie. Om de CT skills van de leerlingen vast te leggen hebben we duo's gestimuleerd hardop te denken bij het oplossen van de programmeertaak (zie procedure voor een beschrijving van de programmeertaak). Voor de ontwikkeling van het observatie-instrument waarmee het hardop denken van de duo's vast kan worden gelegd is aansluiting gezocht bij de rubric 'Cognitive skills in collaborative problem solving' (Hesse, Care, Buder, Sassenberg & Griffin, 2015). Zij operationaliseren samenwerkend probleemoplossen in vier indicatoren die we kunnen relateren aan een aantal CT skills te weten: probleemdecompositie- analyseren van het probleem in deeltaken, abstraheren – vaststellen wat de kern van het probleem is, testen of de instructies op de gewenste manier worden uitgevoerd, ordenen en met elkaar in verband brengen van gegevens en fouten opsporen en oplossen. De rubric is geschikt gemaakt voor de context van dit onderzoek en de omschrijvingen hebben we geconcretiseerd in termen van de programmeertaak met de Ozobot (cursief). Het observatie-instrument is beschreven in Tabel 2. Voor een uitgebreide beschrijving van de totstandkoming van de rubric verwijzen we naar Bijlage 1.

Het gebruik van de rubric is gepilot door de onderzoekers bij duo's uit groep 4 en individuele leerlingen in de leeftijd van 10-12 jaar. Op basis van de bevindingen is het observatie-instrument bijgesteld. Het definitieve instrument wordt gepresenteerd in Tabel 2.

Indicator	Zeer Laag: 1	Laag: 2	Middel: 3	Hoog 4	Zeer hoog 5
Analyseert en beschrijft het probleem in eigen woorden.	Probleem wordt niet onderkend zoals het zich voordoet.	Probleem is onderkend zoals het zich voordoet.	Probleem is onderverdeeld in subtaken.	Herkent de noodzakelijke sequentie van subtaken.	Herkent relaties en verbindingen van subtaken en generaliseert de oplossing.
CT skill: Probleem-decompositie	*De leerlingen snappen bijv. niet dat er een code gebruikt kan worden.*	*De leerlingen snappen bijv. dat Ozobot over water moet, maar weten niet dat hier een code voor gebruikt kan worden*	*De leerlingen snappen bijv. dat Ozobot over water moet en begrijpen dat dit een spring code moet zijn*	*De leerlingen snappen bijv. dat Ozobot over water moet en beredeneren de benodigde springcode(s)*	*De leerlingen beredeneren de volledige juiste plaatsing van codes.*
Stelt een duidelijk doel voor een taak.	Ze stellen geen doelen.	Stelt een algemeen doel zoals taak afronding.	Stelt doelen voor subtaken.	Stelt doelen waarbij de relatie tussen subdoelen erkend wordt.	Stelt doelen waarbij de gehele relatie tussen alle subdoelen erkend wordt.
CT skill: abstraheren	*De leerlingen weten niet goed hoe ze moeten beginnen aan deze taak*	*De leerlingen beginnen bijv. zonder voorbereiding*	*De leerlingen lezen eerst de codes door en beginnen met een willekeurige code*	*De leerlingen lezen de codes door en beginnen gericht met een eerste code*	*De leerlingen zoeken eerst alle benodigde codes op en beredeneren de volgorde*
Implementeert mogelijke oplossingen voor een probleem en bekijkt de voortgang.	Ze beginnen, maar passen de opdracht niet toe.	Trial & Error handelingen: uitproberen.	Doelgerichte opeenvolging van handelingen.	Systematisch alle mogelijke oplossingen verkennen.	Past de gekozen probleemoplossing systematisch toe op alle subtaken en test de werking

Indicator	Zeer Laag: 1	Laag: 2	Middel: 3	Hoog 4	Zeer hoog 5
CT skill: testen	*De leerlingen gebruiken bijv. niet geselecteerde codes, tekenen bijv. zelf op het blad*	*De leerlingen proberen uit de geselecteerde codes in volstrekte willekeur welke past en testen deze (soms)*	*De leerlingen plaatsen doelgericht een code en testen deze altijd*	*De leerlingen redeneren verder vanaf de geplaatste code en testen deze*	*Alle codes worden direct ingekleurd en de werking wordt getest.*
Herkent relaties en verbindingen tussen elementen.	Focust niet op de informatie	Focust op losse stukjes informatie.	Verbindt stukjes informatie.	Formuleert verbindingen tussen verschillende stukken informatie.	NVT
CT skill: gegevens ordenen	*De leerlingen doen maar wat*	*Pakt een willekeurige code en gaat hiermee aan de slag*	*Pakt een willekeurige code, gaat hiermee aan de slag en herkent dan een vervolgcode*	*Beredeneert welke code logisch volgt op de startcode en vandaaruit een volgende code*	

Tabel 2: Het observatie-instrument (gebaseerd op Hesse et al., 2015)

In de rubric van Hesse et al. (2015) ontbreekt de CT skill fouten opsporen en oplossen. Bij het analyseren van de filmpjes concludeerden wij dat deze CT skill interessant is om mee te nemen. In dit onderzoek bedoelen we met fouten opsporen en oplossen de handelingen die een leerling uitvoert om te zorgen dat de Ozobot de ingevulde code wel uitvoert. Hierbij is niet gekeken of de code goed of fout is. We hebben 'fouten opsporen en oplossen' als indicator toegevoegd aan het instrument waarbij wij ons gebaseerd hebben op de beschrijving van de SLO (nd) . In Tabel 3 wordt de uitbreiding van het observatie-instrument beschreven. Alle filmpjes zijn opnieuw bekeken om te bezien of leerlingen de foutoplossing gestructureerd aanpakken.

Indicator:	*Zeer Laag: 1*	*Laag: 2*	*Middel: 3*	*Hoog 4*	*NVT*
CT skill: fouten opsporen en oplossen	*Snapt niet waarom het fout gaat*	*Herkent één mogelijke reden van de fout. De aanpak om tot foutoplossing te komen is nog niet gestructureerd*	*Herkent meerdere of zelfs alle mogelijke redenen van de fout. De aanpak om tot foutoplossing te komen is nog niet gestructureerd*	*Doorloopt stap voor stap alle mogelijke oorzaken voor de fout en probeert deze uit te sluiten*	*Er is geen foutoplossing nodig. De programmeer taak is juist gemaakt.*

Tabel 3: Observatie-instrument: fouten opsporen en oplossen

Procedure

Pabo-studenten hebben de rol van onderzoeksassistent gekregen en zij zijn daar voorafgaand aan het onderzoek in een korte training op voorbereid. In deze training zijn ze geschoold in het werken met de Ozobots (de meesten hadden hier al eerder mee gewerkt), er is uitgelegd hoe de klassikale les verzorgd moest worden en welke elementen in deze les aan de orde moesten komen. Er is tevens stil gestaan bij de wijze waarop het onderzoek rondom de duo's plaats zou moeten vinden. De pabo-studenten hebben een instructietekst voor de leerlingen gekregen om voor aanvang van het onderzoek voor te lezen. Er is uitgelegd hoe de studenten het observatie-instrument moesten gebruiken en ze hebben voorbeeld filmpjes gezien aan de hand waarvan ze geoefend hebben in het werken met het observatie-instrument.

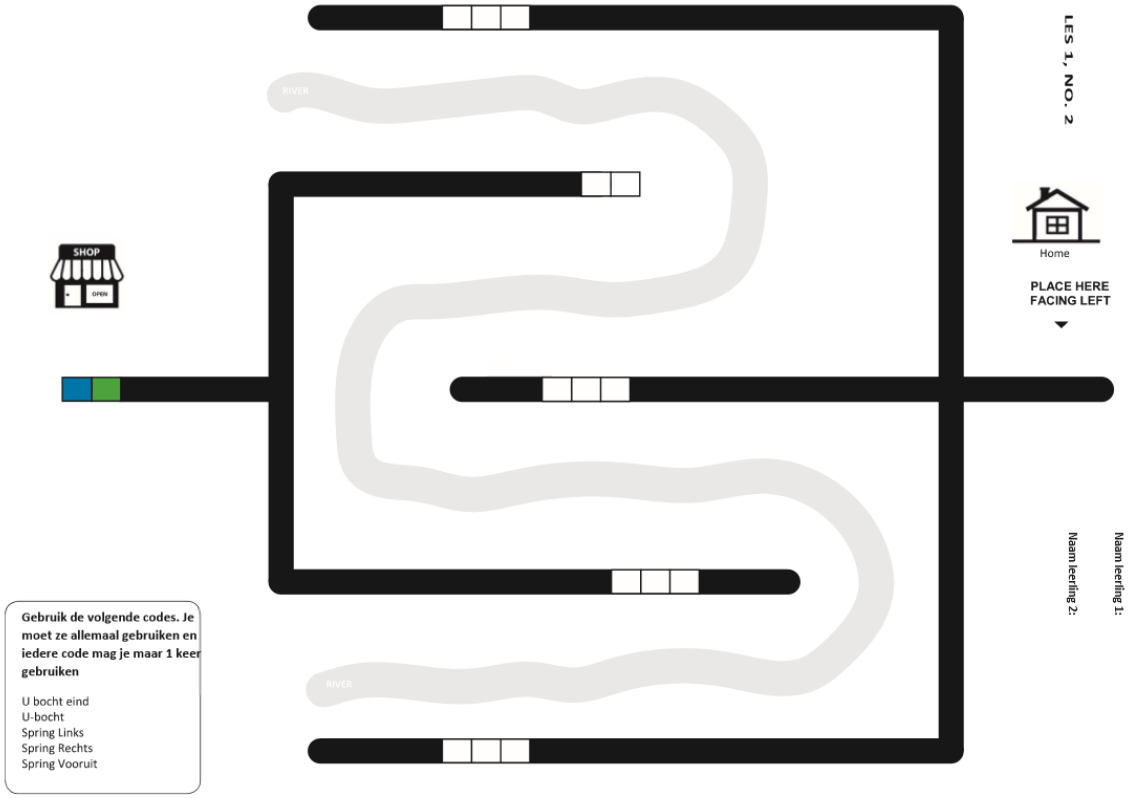
De pabo studenten hebben op hun stageschool zelf het moment gepland om de les te verzorgen en aan de slag te gaan met de duo's. Op voorhand hadden de studenten, middels een toestemmingsformulier, toestemming aan de ouders gevraagd om kinderen te mogen filmen en deze beelden te gebruiken voor het onderzoek. De pabo-studenten hebben vervolgens deze leerlingen ingedeeld in groepjes rekening houdend met leeftijd en sekse.

Zij hebben in hun stageklas de Ozobot geïntroduceerd in een klassikale les. In deze klassikale les zijn de leerlingen in viertallen aan de slag gegaan met een kleine programmeertaak in het werken met de Ozobot. De leerlingen mochten een parcours vullen met codes en aansluitend op een leeg vel vrij lijnen en kleurcodes tekenen. De studenten hadden de opdracht om de kleurcodes – die voor de programmeertaak nodig waren – in ieder geval te laten oefenen. De introductie les en programmeertaak zijn al eerder gebruikt in onderwijs aan leerlingen van deze leeftijd.

Vervolgens hebben de leerlingen in tweetallen onder toezicht van de onderzoeksassistent in een afzonderlijke ruimte gewerkt aan de programmeertaak.

De bedoeling van deze programmeertaak is dat de leerlingen de Ozobot van huis naar de winkel sturen, waarbij ze met 5 kleurcodes (U BOCHT, U BOCHT EIND, SPRING LINKS, SPRING RECHTS en SPRING VOORUIT) -die ieder maar één keer gebruikt mogen worden- moeten zorgen dat de Ozobot binnen het parcours blijft, over de rivier komt en de winkel bereikt. De voor deze programmeertaak benodigde kleurcodes staan op codenaam benoemd op het opdrachtvel. Met deze codenaam konden de leerlingen op een codevel kijken welke kleursequenties gebruikt moesten worden.

Onderstaande afbeelding (Figuur 2) toont de programmeertaak die de duo's hebben moeten maken.



Figuur 2: De programmeertaak voor de duo's

De pabo-student heeft de activiteiten die het tweetal uitvoert in de tijd geobserveerd en het hardop denken indien nodig gestimuleerd. Alle tweetallen kregen een identieke instructie. Sommige studenten lieten de tweetallen deze instructietekst zelf lezen. Anderen hebben de instructietekst voorgelezen. De tweetallen kregen maximaal 15 minuten om de programmeertaak af te ronden. Gedurende het maken van de programmeertaak zijn de leerlingen gefilmd en de pabo-studenten hebben tijdens de programmeertaak het observatieformulier per code ingevuld en geen verdere hulp geboden.

Data-analyse

Alle filmpjes zijn aan de hand van een observatieformulier beoordeeld door de pabo-student en nogmaals door de onderzoeker (eerste auteur). Eventuele afwijkingen (vaak werden leerlingen uit een klas in het geheel te hoog ingeschaald) zijn op basis van de observaties van de onderzoeker gecorrigeerd. De onderzoeker heeft een groot aantal filmpjes uitgeschreven (tekst en bewegingen). Op basis daarvan zijn enkele (representatieve) filmpjes uitgewerkt tot beschrijvingen waaruit blijkt waarom bepaalde duo's al dan niet succesvol zijn.

De resultaten van het observatie-instrument zijn geanalyseerd met SPSS. Gemiddelden en standaarddeviaties zijn berekend en significantie is getoetst met non-parametrische statistiek. Er is nagegaan of er sprake is van interactie tussen groep en de samenstelling van het duo. Met behulp van Excel is een scatterplot gemaakt van de ruwe data en is een regressielijn toegevoegd.

Het opsporen en oplossen van fouten gebeurt door de duo's per code of soms met meerdere codes tegelijkertijd, maar als het werkt, dan is foutoplossing niet nodig. De foutoplossing bleek niet over alle duo's even goed te waarderen. De onderzoeksassistent heeft in de foutoplossing soms hulp geboden, afgekapt (ga maar door want de code is goed) of de oplossing is niet meer op video opgenomen. Hierdoor hebben veel duo's geen score per code op dit onderdeel. We hebben daarom alleen die duo's in de berekeningen meegenomen waarvan we een volledige score hebben. Het betreft hier 30 van de 42 duo's.

4. Resultaten

In deze paragraaf bespreken we eerst de kwantitatieve resultaten en daarna illustreren we onze bevindingen aan de hand van de beschrijving van de handelingen van zes duo's, 3 uit groep 5 en 6 en 3 uit groep 7 en 8, die resp. laag, gemiddeld en hoog scoren.

4.1 Kwantitatieve resultaten

Algemeen

Van de 42 duo's lossen 18 duo's (42.9 %) de casus op. Het aantal correct geplaatste codes neemt toe naarmate de duo's langer bezig zijn met de programmeertaak. Succesvolle duo's (zij hebben de casus opgelost):

- nemen de tijd om de plaatsing van de codes door te spreken. Ze beginnen niet direct (15 duo's);
- kijken of er een code bij zit die als enige op die plek kan. In dit voorbeeld U bocht eind (2 vakjes als enige) (16 duo's);
- halen door welke codes al gebruikt zijn (5 duo's);
- tekenen de codes zorgvuldig in; binnen de lijnen en niet te dik (15 duo's);
- zorgen dat alle informatie helder is en stellen vragen als de instructie niet duidelijk is (18 duo's)
    - Ozobot moet van huis naar de winkel;
    - Ozobot moet via iedere weg bij de winkel komen;
    - Iedere code uit de set van 5 codes mag je maar 1 x gebruiken en je mag alleen die 5 codes gebruiken.

Vrijwel alle succesvolle duo's wijzen met hun vinger of pen de weg aan die de Ozobot moet lopen of ze laten de Ozobot de weg bepalen.

De succesvolle duo's uiten de volgende overwegingen tijdens de uitvoering van de programmeertaak[1]:

- we mogen alleen deze 5 codes gebruiken (18 duo's);
- wacht even met inkleuren, dan kijken we eerst waar de andere codes moeten komen; 4 duo's beredeneren eerst de volledige plaatsing van de codes en gaan dan testen; de andere 14 duo's gaan na de plaatsing van 1 of enkele codes gelijk testen;
- als positie 1 'SPRING LINKS' is, dan is positie 5 'SPRING RECHTS' en dan moet positie 3 wel 'SPRING VOORUIT' zijn of afgeleiden hiervan (16 duo's);
- positie 2 moet 'U BOCHT EIND' zijn, want hier kunnen er maar 2 (16 duo's).

Meer oudere dan jongere duo's lossen de casus op. Geen enkel duo uit groep 4 lost de casus op. Relatief veel duo's uit groep 4 komen niet toe aan codes 4 en 5. Bijna 60%

[1] Zie de beschrijving van de opdracht in de methodesectie (onderdeel procedure)

van de duo's uit de groepen 7 en 8 lossen de casus op en ruim een derde van de duo's uit de groepen 5 en 6.
Iets minder dan 2/3 van de jongensduo's (n=7, 63,6%) en ongeveer de helft van de gemengde duo's (n=7, 46.7%) lost de casus op. De meisjesduo's doen het aanzienlijk slechter. Slechts vier meisjesduo's (25%) lossen de casus op. Relatief veel meisjesduo's komen niet toe aan codes 4 en 5. Dit is mede verklaarbaar door het relatief grote aantal meisjesduo's in groep 4.

Probleemdecompositie, abstraheren, testen en gegevens ordenen

Bij de *probleemdecompositie* is gekeken hoe de duo's de casus analyseren en in eigen woorden beschrijven. De score bij het plaatsen van de eerste code ligt rond de drie (gem. = 3.02, sd=1.09, N=42) en dat wijst erop dat de duo's herkennen dat het probleem onderverdeeld is in subtaken. Deze scores lopen langzaam op bij het plaatsen van de volgende codes. Bij het plaatsen van de laatste code is de gemiddelde score 3.74 (sd= 0.75, N=34). Nadere analyse laat zien dat dit verschil vooral in het begin plaatsvindt, tussen de plaatsing van code 1 en code 2. Dit verschil is significant met een groot effect (zie Bijlage 2, Tabel 1).

Bij het *abstraheren* is nagegaan in welke mate de duo's een duidelijk doel voor de taak formuleren. Tussen het plaatsen van de eerste en de laatste code loopt het gemiddelde op tussen 2.95 (sd=1.21, N=42) en 3.65 (sd=0.849, N=34). Dat betekent dat in de loop van het uitvoeren van de programmeertaak meer duo's doelen stellen om subtaken op te lossen. Het verschil in de gemiddelde score tussen het plaatsen van code 1 en code 2 is het grootst. Dit verschil is significant met een middel tot groot effect (zie Bijlage 2, Tabel 1).

Er is gekeken hoe de duo's de mogelijke oplossingen *testen* en de voortgang hiervan bekijken. Bij het plaatsen van de eerste code is de gemiddelde score 2.83 (sd= 1.17, N=42) en bij de laatste code is de gemiddelde score 3.38 (sd= 0.95, N=34). Dit betekent dat bij de start van de programmeertaak veel duo's via trial en error het probleem proberen op te lossen, maar dat gedurende het uitvoeren van de programmeertaak steeds meer duo's doelgericht handelen. Tussen het plaatsen van de eerste en de tweede code is het verschil in de gemiddelde score het grootst. Dit verschil is significant met een middel tot groot effect (zie Bijlage 2, Tabel 1).

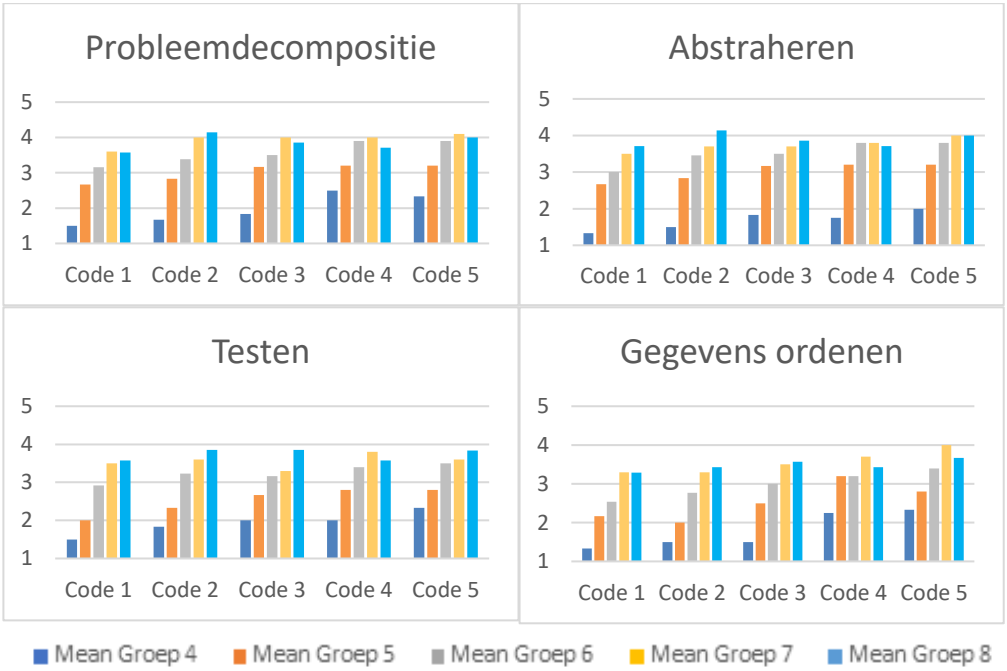
Er is nagegaan of de duo's *gegevens ordenen* en de verbinding tussen de elementen herkennen. Ook hier lopen de gemiddelde scores op naarmate de duo's meer codes plaatsen. Bij de eerste code is de gemiddelde score 2.62 (sd=1.17, N=42) en bij de laatste score 3.44 (sd=0.93, N=34). Dat betekent dat de duo's steeds beter verbindingen leggen tussen de beschikbare informatie. Hier is het grootste verschil tussen de gemiddelde score van de duo's tussen het plaatsen van code 2 en 3. Het verschil is significant met een middel tot groot effect (zie Bijlage 2, Tabel 1).

Op alle indicatoren is de standaarddeviatie hoog bij de start van de programmeertaak. Deze neemt weliswaar enigszins af, maar blijft relatief hoog. Dat betekent dat er grote verschillen zijn tussen de duo's.

Als we kijken naar het verschil in leeftijd dan zien we dat de jongste duo's op alle indicatoren lager scoren dan oudere duo's. Voor alle indicatoren ligt de gemiddelde score van de duo's uit groep 4 iets onder of ruim onder de 2. De jongste duo's hebben moeite om te bepalen wat ze moeten doen. Ze doen vaak maar wat en ze focussen zich niet op de informatie.

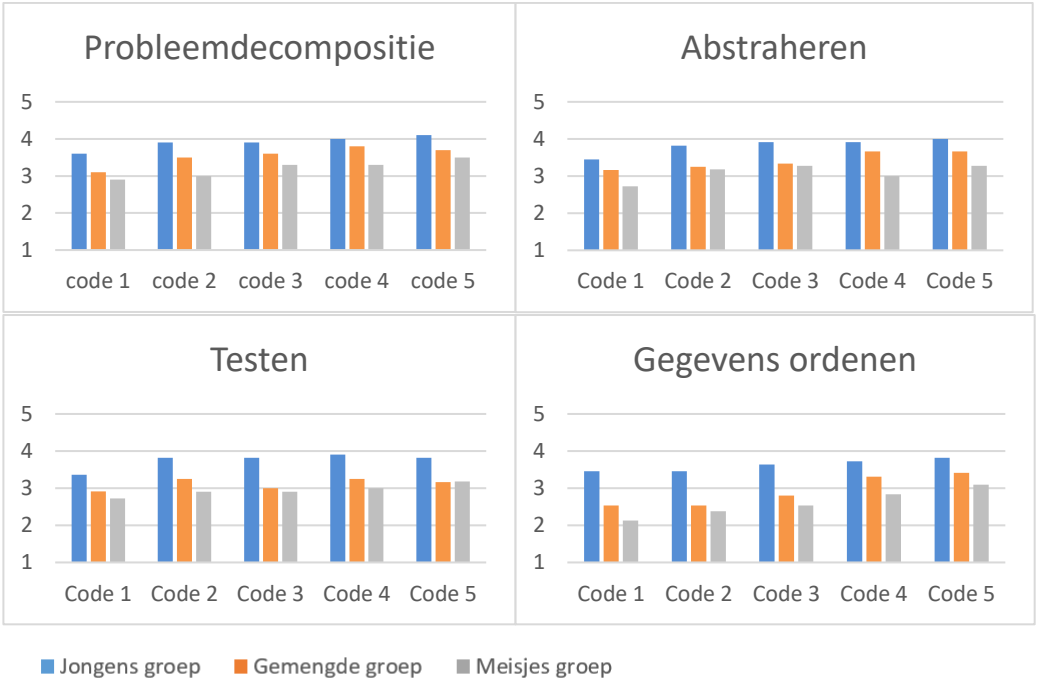
De leerlingen uit groep 5 scoren gemiddeld rond de 3 op probleemdecompositie en abstraheren. Zij benoemen subtaken en stellen daarvoor doelen. Op testen en gegevens ordenen scoren ze gemiddeld iets lager (rond de 2.5). Dat betekent dat sommige groepen vooral via trial en error het probleem proberen op te lossen en de informatie niet in relatie tot elkaar wordt gebruikt terwijl andere duo's meer doelgericht codes plaatsen en testen en de beschikbare informatie gebruiken.

Pas vanaf groep 6 onderkennen de duo's al bij de start dat de casus verdeeld is in subtaken en stellen de duo's doelen voor die subtaken (gemiddelde score is 3.5). Ze testen met een doelgerichte opeenvolging van handelingen en ze verbinden stukjes informatie (score rond de 3). Als ze langer bezig zijn onderkennen ze volgordelijkheid in die subtaken en stellen ze doelen waarbij de relatie tussen de subdoelen erkend wordt. Vanaf groep 7 en 8 liggen de scores op alle indicatoren (ruim) boven de 3.5. Als ze langer bezig zijn, scoren de duo's gemiddeld op alle indicatoren 4 of iets daaronder, wat betekent dat ze subtaken onderscheiden die in volgorde moeten worden opgelost, daarvoor gerichte doelen stellen en systematisch de mogelijke oplossingen verkennen door de beschikbare informatie met elkaar te verbinden. Figuur 3 tot 6 laat de gemiddelden per groep op de onderscheiden indicatoren zien.



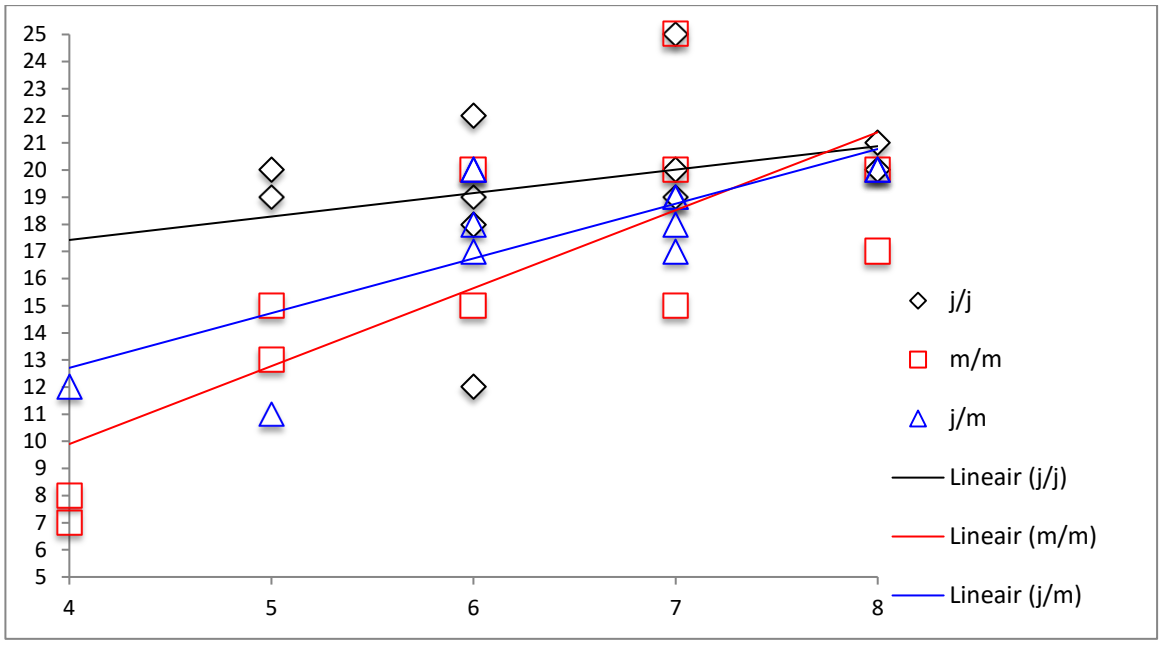
Figuur 3-6: Gemiddelde score op de indicatoren per code en per groep

Op alle indicatoren scoren de jongensduo's gemiddeld hoger dan de gemengde duo's en de meisjes duo's. Over de vier indicatoren heen scoren de jongensduo's gemiddeld tussen de 3.62 (gegevens ordenen) en 3.91 (probleemdecompositie). Voor de gemengde duo's liggen deze gemiddelden lager, tussen de 2.92 (gegevens ordenen) en 3.38 (probleemdecompositie), en voor de meisjesduo's nog lager, tussen de 2.59 (gegevens ordenen) en 3.09 (probleemdecompositie). In het algemeen eindigen de gemengde duo's en de meisjesduo's gemiddeld op het beginniveau van het gemiddelde van de jongensduo's of iets daaronder. Zie Figuur 7-10.



Figuur 7-10: Gemiddelde score op de indicatoren per code en per samenstelling van het duo

We zijn per indicator nagegaan of er sprake is van een interactie-effect tussen groep en de samenstelling van het duo. De resultaten laten zien dat jongensduo's voor alle indicatoren hoger scoren in groep 4-6 in vergelijking met gemengde en meisjesduo's. Echter in groep 7 en 8 is dit verschil weggewerkt en is de gemiddelde score van de duo's, ongeacht hun samenstelling ongeveer gelijk. Figuur 11 geeft het resultaat weer voor de indicator probleemdecompositie. De resultaten van de andere indicatoren zijn te vinden in de Bijlage 2, Figuur2-4.



Figuur 11: Probleemdecompositie: Behaalde score per duo (jongens/jongens (j/j); meisje/meisje (m/m) en gemengd (j/m)) per groep. Scatterplot en regressielijnen.

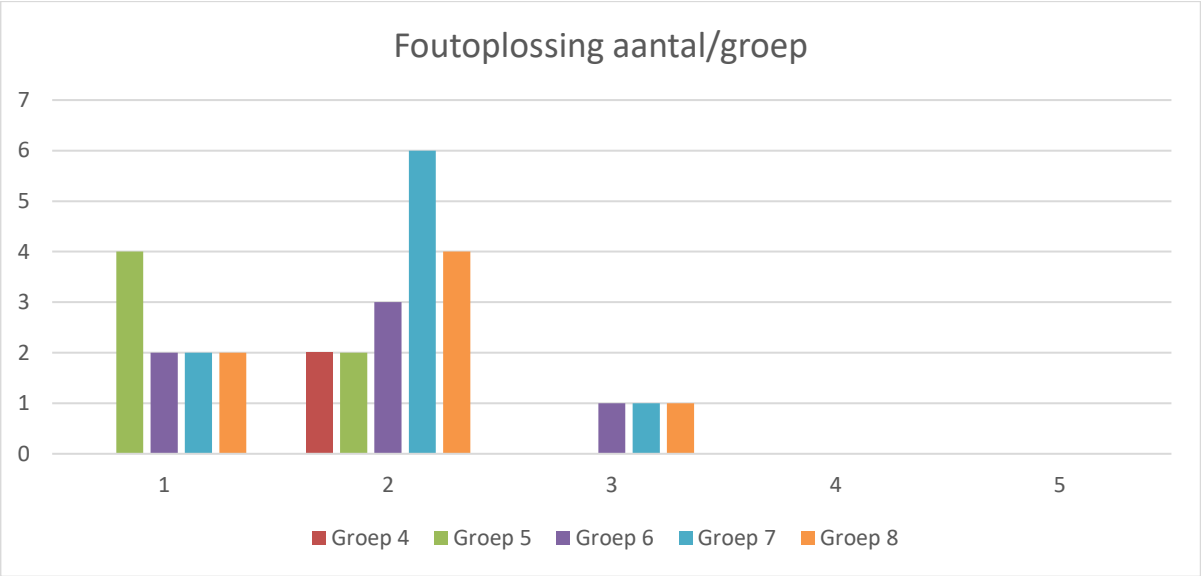
Fouten opsporen en oplossen

Bij de foutoplossing is gekeken in hoeverre leerlingen een verklaring konden vinden voor het niet werken van de Ozobot. Het testen is niet altijd per code te beoordelen. Sommige duo's testen pas na het plaatsen van meerdere codes en als het allemaal werkt, dan is foutoplossing niet nodig. Bij 30 van de 42 duo's hebben we de foutoplossing voor alle 5 codes kunnen scoren. 90% van deze 30 duo's herkent geen of maximaal één mogelijke reden van de fout. De aanpak om tot foutoplossing te komen is niet gestructureerd. 1 duo had geen foutoplossing nodig. De codes waren goed geplaatst en werden herkend door de Ozobot.

Bij het debuggen – waarom laat de Ozobot niet zien wat er verwacht wordt- komen de meeste duo's niet verder dan één mogelijke verklaring van het probleem. Er zijn weinig duo's die kijken welke kleuren de Ozobot toont wanneer er een kleurcode gelezen wordt. Veelal beperken ze zich tot het gebruik van een van onderstaande opties:

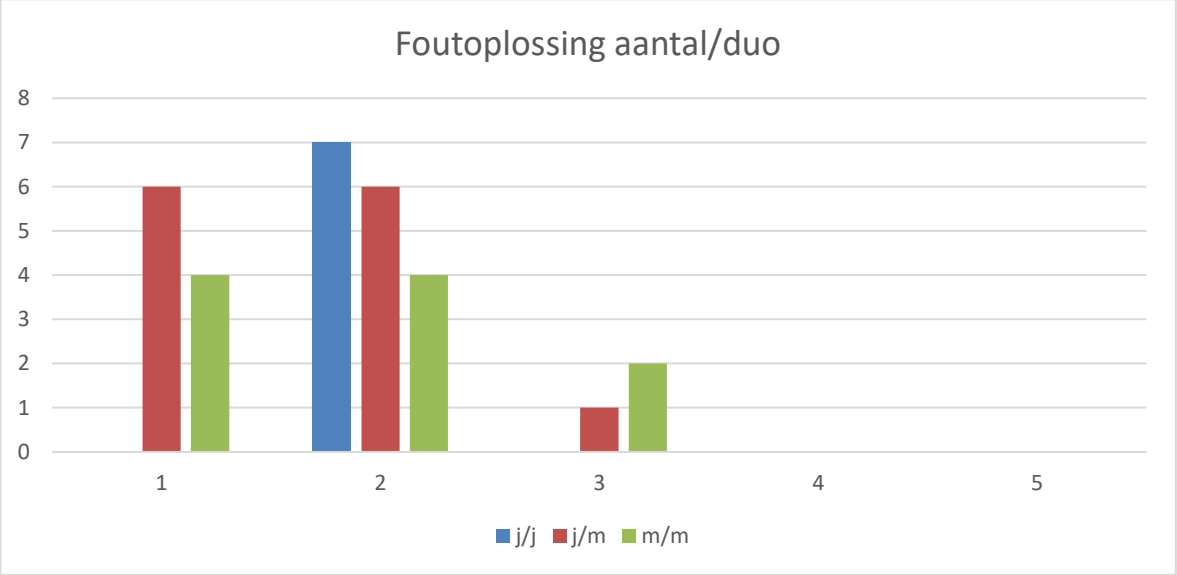
- controleren of de kleurcodes goed zijn ingekleurd. Binnen de vakjes, niet te licht of te donker;
- controleren of de Ozobot gekalibreerd moet worden;
- controleren of een andere Ozobot de code wel herkent.

Pas vanaf groep 6 komt 1 duo tot een score 3. Dit duo herkent meerdere of zelfs alle mogelijke redenen van de fout. De aanpak om tot foutoplossing te komen is nog niet gestructureerd. 1 duo (groep 6) heeft geen foutoplossing nodig (zie Figuur 12). De programmeertaak is juist uitgevoerd en de Ozobot doorloopt het parcours.



Figuur 12: Aantal duo's per aantal foutendiagnoses per groep.

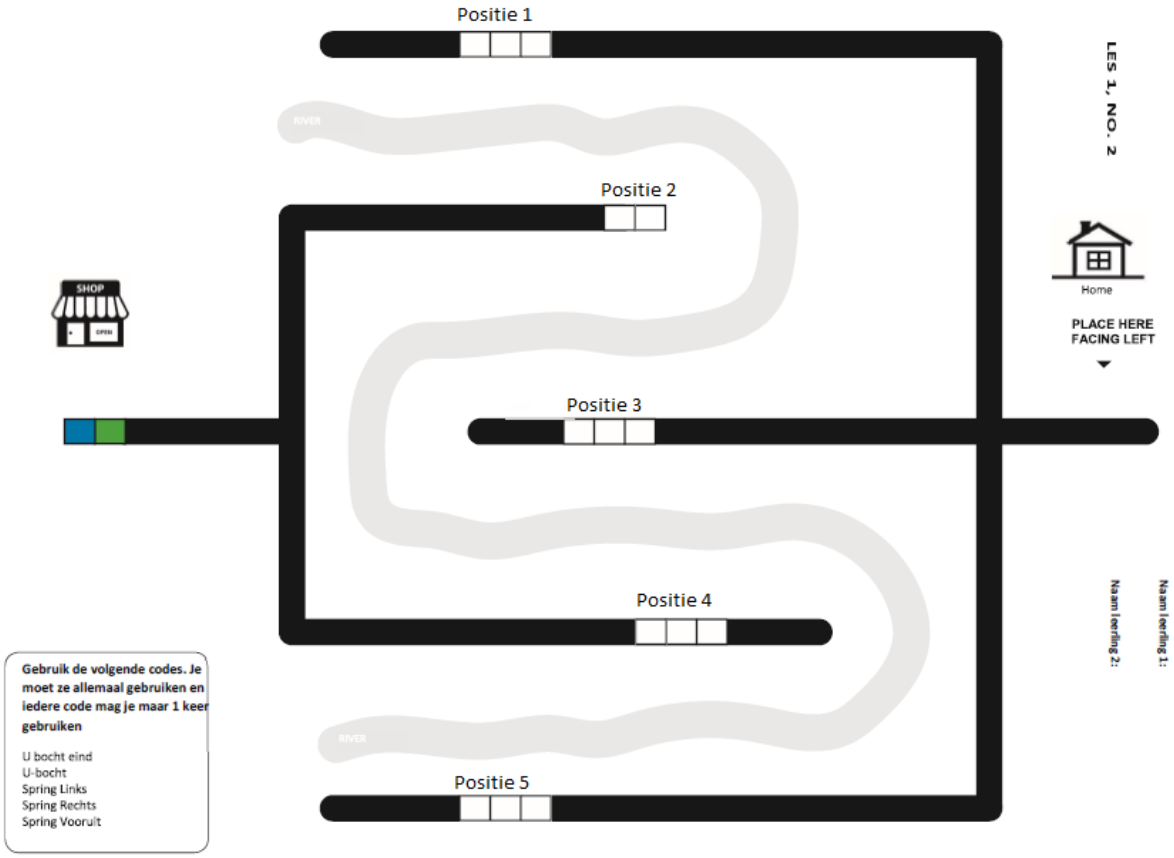
Bij de jongensduo's heeft 1 duo de programmeertaak opgelost en was foutoplossing niet nodig (score 5). Bij de meisjesduo's zien we 2 duo's met een score 3. Zij herkennen meerdere of zelfs alle mogelijke redenen van de fout. Gemiddeld scoren jongens duo's (1.80) en meisjes duo's (1.75) ongeveer gelijk. Gemengde duo's scoren iets lager (1.53). De aanpak om tot foutoplossing te komen is bij de meeste duo's nog niet gestructureerd. Zie Figuur 13.



Figuur 13: Aantal duo's per aantal foutendiagnoses (jongens duo's, meisjes duo's, gemengde duo's)

4.2 Illustratieve casussen

In deze paragraaf beschrijven we de handelingen van 6 duo's, 3 uit groep 5 en 6 en 3 uit groep 7 en 8. Iedere set van 3 beschrijvingen omvat een duo wat hoog, gemiddeld en laag scoort op het observatie-instrument. Ter herinnering is de programmeertaak die de duo's hebben moeten maken bijgevoegd (zie Figuur 14). De posities zijn later toegevoegd.



Figuur 14: de programmeertaak voor de duo's , inclusief posities voor codes

Isa en Mette uit groep 7

De onderzoeksassistent leest de instructie voor. Als de meisjes starten, proberen ze de route te visualiseren die de robot moet lopen. Ze bewegen met hun vinger langs de diverse routes. Ze kijken goed naar de 5 codes die ze mogen gebruiken en zoeken op de codetabel welke kleuren dat zijn.

Ze vragen nog wat verduidelijking van de programmeertaak. De vraag valt binnen de tekst die de onderzoeksassistent mag geven. Ze praten samen goed over de te gebruiken codes. Bv 'U BOCHT'. "Ja maar je mag hem maar 1 x gebruiken". Het is wel duidelijk dat Isa de leidende rol in dit duo heeft. Mette volgt Isa volledig.

Al uitproberend met de vinger beredeneren ze de volledige codeplaatsing. "Daar moet wel 'U BOCHT EIND', want dat is de enige met 2 vakjes" en "als hij hier links moet, dan moet hij daar rechts".

Ze starten met het inkleuren van de codes en kleuren gelijk de 5 verschillende codes in. De Ozobot wordt gepakt en goed gekalibreerd en hij wordt op het vel gezet. De Ozobot loopt naar de eerste code (‘SPRING RECHTS’) en herkent deze niet. Beide meisjes bedenken dat de Ozobot de kleuren niet goed leest. Alle andere codes worden wel door Ozobot herkend en hij komt van huis naar de winkel.
De meisjes tekenen vervolgens een nieuwe lijn met code naast al eerder ingekleurde code. Dit werkt niet en ze concluderen dat de lijn te dik is. Ze kleuren de code in op een nieuw vel en daar herkent de Ozobot de code wel.

Dit duo scoort zeer hoog op het observatie-instrument.

- Ze beredeneren de volledige juiste plaatsing van de codes.
- Ze zoeken eerst alle benodigde codes op en beredeneren de volgorde.
- Alle codes worden direct ingekleurd en de werking wordt getest.
- Ze doorlopen stap voor stap alle mogelijke oorzaken voor de fout en proberen deze uit te sluiten.

Fred en Gijs uit groep 7

De onderzoeksassistent leest de instructie voor. Gijs stelt een verduidelijkende vraag. De onderzoeksassistent mag hier niet op antwoorden. Het duo kleurt eerst om te oefenen op een leeg vel de code ‘SPRING VOORUIT’ in, voorafgegaan door een zwarte lijn en direct gevolgd door een zwarte lijn. Ze pakken de Ozobot en testen deze. Ze bespreken vervolgens uitgebreid hoe de Ozobot op het vel zal kunnen lopen. Ze lopen met hun vinger alle weggetjes langs. Ze noemen in dit proces veel goede codes. In dit proces noemen ze bijvoorbeeld voor positie 5 ‘SPRING RECHTS’ en voor positie 3 ‘SPRING VOORUIT’. Ze zien hierin nog niet de parallel met positie 1. Op positie 1 kleuren ze de ‘U BOCHT’ in. Deze code testen ze. Ozobot draait keurig om en loopt door naar positie 5. Hier stopt hij. Het duo bespreekt goed welke codes ze nog moeten plaatsen. Positie 2 weten ze zeker, dat is ‘U BOCHT EIND’. Uiteindelijk kleuren ze op positie 5 ‘SPRING RECHTS’ in. Op positie 4 plaatsen ze ‘SPRING LINKS’ en op positie 3 ‘SPRING VOORUIT’. Aansluitend kleuren ze positie 2 in met ‘U BOCHT EIND’. Ze gaan testen met de Ozobot. Deze loopt naar positie 1, keert om (‘U BOCHT’) en loopt terug en stopt bij de huispositie. Ze snappen niet goed waarom de Ozobot weer terugkeert op de huispositie. Fred bedenkt dat de ‘U BOCHT’ misschien omgewisseld moet worden met ‘SPRING LINKS'. De Ozobot loopt vanaf de huispositie iedere keer richting positie 1 of positie 5. Ze willen graag positie 3 testen. De oplossing die ze kiezen is om de Ozobot te kalibreren. Ze vragen vervolgens om een nieuw vel en kleuren op positie 3 de U bocht in. Ze testen deze. De Ozobot gaat richting positie 1. Ze bespreken goed welke weggetjes de Ozobot zou kunnen kiezen. Met hun vingers geven ze de bewegingen correct aan, maar ze benoemen de verkeerde code hierbij. Ze kleuren op positie 1 ‘SPRING LINKS’, op positie 2 ‘U BOCHT EIND’, op positie 4 ‘SPRING RECHTS’ en op positie 5 ‘SPRING VOORUIT’. Dit testen ze. Bij het testen benoemt Fred dat de codes op posities 4 en 5 omgewisseld moeten worden. Ze zien bij het testen dat de kleuren niet goed gelezen worden en kleuren deze bij.

Dit duo scoort middel tot hoog in het observatie-instrument:

- Ze snappen dat ozobot over het water moet en ze beredeneren de springcodes
- Ze lezen de codes door en beginnen gericht met een eerste code
- Ze redeneren verder vanaf de geplaatste code en testen deze
- Ze beredeneren welke code logisch volgt op een startcode en vandaaruit de volgende code.
- Ze herkennen meerdere of zelfs alle mogelijke redenen van de fout. De aanpak om tot foutoplossing te komen is nog niet gestructureerd.

Helaas zijn ze niet succesvol in het oplossen van de programmeertaak omdat ze wel de actie juist aanwijzen, maar hier in een aantal gevallen een verkeerde code bij kiezen.

Ronald en Inge uit groep 8

De onderzoeksassistent geeft de instructietekst. Het duo begint gelijk. “Dit is toch het begin? Ok. Nou zullen we eerst turbo doen“ en ze pakken de stiften. De docent wijst ze op de te gebruiken codes en vraagt of ze het instructievel al gelezen hebben. Ronald zegt dat hij daar nog geen tijd voor gehad heeft. Het duo leest de instructietekst door. Zodra Ronald klaar is met lezen zegt hij: “Ik denk dat we hier zo. Even kijken. Hier moeten we een kant opspringen, maar welke?” Inge maakt het niet uit. Ronald zegt “Hier kunnen we denk ik het beste rechtdoor” en hij plaatst de code ‘SPRING VOORUIT’ op positie 4. Hij test de code met de Ozobot. De Ozobot springt vooruit. Daarna kiest Ronald voor ‘U BOCHT EIND’. Deze plaatst hij op positie 2. “Hier zijn er 2 en ik zie er hier maar eentje waar er twee kunnen.” Hij test dit met de Ozobot, maar deze herkent de code niet. Ronald probeert het nog een keertje en gaat dan verder met positie 5, waar hij een U bocht inkleurt. Hij bedenkt gelijk dat deze code niet goed is. De onderzoeksassistent helpt het duo door aan te geven dat ze de code ‘SPRING RECHTS’ moeten proberen te tekenen, zodat ze kunnen zien wat er gebeurt. De onderzoeksassistent helpt verder door aan te geven dat “overkleuren” niet kan. Ze geeft aan dat ze de code boven de bestaande code moeten kleuren met een zwart lijntje ervoor en erna. Het duo kleurt deze lijn zo strak tegen de bestaande code aan dat de Ozobot deze niet herkent. De onderzoeksassistent besluit tot extra uitleg over lijndikte en het inkleuren van codes. Ze geeft daarna het duo een nieuw vel. Inge en Ronald kleuren op het nieuwe vel de code ‘SPRING RECHTS’ op positie 5. Ze testen met de Ozobot . Deze herkent de code en doet wat hij moet doen. Ronald kleurt vervolgens op positie 4 de code ‘SPRING VOORUIT’. Hij test dit met de Ozobot. In eerste instantie herkent de Ozobot de code niet. Daarna blijft hij rondjes lopen tussen positie 4 en positie 5. Het duo komt hier niet goed uit. De onderzoeksassistent adviseert ze om eerst naar de andere codes te kijken. Ronald: Wat is een ‘U BOCHT’?” De onderzoeksassistent vraagt hem dit op een ander vel uit te proberen. Het duo tekent een lijn met ‘U BOCHT’ en ze proberen dit uit. De Ozobot herkent de code niet. Dit keer bedenken ze dat de kleur misschien te donker is of dat de Ozobot bijna leeg is. Ze pakken een nieuwe Ozobot en testen. Ronald constateert: “dus een ‘U BOCHT’ is dat hij omdraait”. Onder druk van de tijd kleurt hij op positie 3 nogmaals ‘SPRING VOORUIT’ in.

Inge zegt: "'SPRING VOORUIT' hebben we al gehad.". Ronald gaf aan dat die het niet deed. Ook deze keer doet de Ozobot het niet.

Dit duo scoort laag op het observatie-instrument.

- Ze snappen dat Ozobot over water moet en dat dit een springcode moet zijn.
- Ze lezen eerst de codes door en beginnen met een willekeurige code.
- Ze plaatsen doelgericht een code en testen deze altijd.
- Ze zien de verbanden tussen de codes nog niet. Iedere code is een nieuw probleem (pakt een willekeurige code en gaat hiermee aan de slag).
- Ze herkennen één (bij code 4 en 5 meerdere) mogelijke reden van de fout. De aanpak om tot foutoplossing te komen is nog niet gestructureerd.

Gavin en Doortje uit groep 6

De onderzoeksassistent geeft instructie. Zodra ze beginnen laat Gavin met zijn vinger de mogelijke weggetjes zien. Doortje zegt dat alles maar 1 x gebruikt mag worden. Gavin vertelt Doortje hoe ze op positie 1 'SPRING LINKS' gaan kleuren. Hij wijst naar positie 5 en geeft aan dat dit 'SPRING RECHTS' moet zijn. "Nu moeten we het andersom doen". Gavin zegt dat positie 3 'SPRING VOORUIT' moet zijn. Doortje kleurt deze in. Gavin wijst naar de te gebruiken codes en geeft aan welke gebruikt zijn. Ze testen met de Ozobot. Deze pakt de codes op positie 1 en 3 niet. Positie 5 wordt wel herkend. "Wat is de 'U BOCHT'". De onderzoeksassistent legt uit dat de Ozobot dan omdraait. "O dat willen we juist" en ze kleuren 'U BOCHT' in op positie 4. De onderzoeksassistent vraagt wat er gebeurt als de Ozobot naar positie 2 gaat. "Ja die weten we juist niet". Ze kijken op de te gebruiken codes en roepen in koor 'U BOCHT EIND'. Ze snappen niet goed waarom de Ozobot het bij bepaalde codes wel doet en bij andere niet. De onderzoeksassistent geeft aan dat de codes goed zijn.

Dit duo scoort hoog tot zeer hoog op het observatie-instrument.

- Ze snappen dat Ozobot over water moet en beredeneren de benodigde springcodes. De springcodes worden helemaal beredeneerd. De U bocht (einde) niet.
- Ze lezen de codes door en beginnen gericht met een eerste code.
- Ze redeneren verder vanaf de geplaatste code en testen deze. De eerste 3 worden direct ingekleurd en daarna wordt de werking getest.
- Ze beredeneren welke code logisch volgt op de startcode en vandaar uit een volgende code. Dit voor wat betreft de Spring codes.

Ella en Brenda groep 6

De onderzoeksassistent leest de instructie voor. Het duo kijkt op de te gebruiken codes en kijkt welke kleuren hierbij horen. Ze plaatsen direct U bocht eind op positie 2. Ze strepen in het lijstje van te gebruiken codes de U bocht eind door. Met de vinger lopen ze de mogelijke weggetjes die de Ozobot kan nemen. Ze beredeneren dat bij positie 1 de Ozobot links moet. Ze kijken op de te gebruiken codes en vinden 'SPRING LINKS'. Deze plaatsen ze op positie 1. 'SPRING LINKS' wordt doorgehaald op het lijstje

codes. Op positie 3 plaatsen ze 'SPRING RECHTS'. 'SPRING RECHTS' wordt doorgehaald op het lijstje codes. Ze bedenken gelijk dat de code op positie 3 'SPRING VOORUIT' had moeten zijn. De onderzoeksassistent leest voor dat ze de code ernaast kunnen plaatsen. Ze tekenen 'SPRING VOORUIT' vast aan de al geplaatste code op positie 3. Ze vragen of iedere code maar 1 x gebruikt mag worden. Dit wordt bevestigd door de onderzoeksassistent. Ze plaatsen 'SPRING RECHTS' op positie 4. Op enig moment roept Brenda: "Ik weet het dit moet 'SPRING RECHTS' zijn (positie 5) en dan is dit 'U BOCHT' (positie 4)". De dames corrigeren dit in het vel. Als ze klaar zijn testen ze met de Ozobot. Deze wordt op de kalibreerstip gezet en gekalibreerd. Met 2 Ozobots testen ze de codes. Omdat de correcte codes strak tegen de andere codes aangekleurd zijn werken de codes niet. Ze pakken een nieuw vel waarop ze alle codes op de juiste positie overnemen. Daarna gaan ze testen. Ozobot wordt gekalibreerd. Ze zetten 2 Ozobots op het vel en gaan testen. Het testen gaat -ondanks wat botsingen omdat de Ozobots elkaar in de weg zitten- goed. Ozobot komt bij de winkel.

Dit duo scoort middel tot hoog in het observatie-instrument:

- Ze snappen dat Ozobot over het water moet en ze beredeneren de springcodes
- Ze lezen de codes door en beginnen gericht met een eerste code
- Ze redeneren verder vanaf de geplaatste code en testen deze
- Ze pakken een willekeurige code, gaan daar mee aan de slag en herkennen dan de vervolgcode.
- Ze doorlopen stap voor stap alle mogelijke oorzaken voor de fout en proberen deze uit te sluiten.

Mirjam en Joris uit groep 5

De onderzoeksassistent leest de instructie voor. Het duo begint met het kalibreren van de Ozobot. Joris zegt "U bocht is omkeren". Mirjam wil graag beginnen bij positie 2, 'U BOCHT EIND'. Joris kleurt op positie 3 de 'U BOCHT' in. Mirjam zet de Ozobot aan en plaatst hem op de huispositie. De Ozobot gaat naar positie 1. Joris kleurt ook hier de 'U BOCHT' in. Mirjam zegt "nu heb je 2 x de 'U BOCHT'". De geplaatste 'U BOCHT' op positie 3 werkt naar behoren. De 'U BOCHT' op positie 1 niet. Mirjam: "Je hebt hem te dik gekleurd. Oelewapper!". Tijdens het testen kleurt Joris positie 5 in, wederom met een 'U BOCHT'. Mirjam kijkt op de te gebruiken codes en kijkt welke ze gebruikt hebben. Joris wil gewoon lijnen bijtekenen. Hij tekent een lijn van positie 5 over het water. Dikke pret samen! Tijdens het testen komt de Ozobot bij positie 5 en keert om. Hij komt dus niet bij de zelfgetekende lijn. Ze krijgen een nieuw vel en de onderzoeksassistent benoemt nog een aantal zaken uit de instructie. Mirjam wil dat Joris een zwarte streep tekent. Ozobot komt via die lijn bij de winkel.

Dit duo scoort laag tot zeer laag op het observatie-instrument.

- Ze snappen bijvoorbeeld dat de Ozobot over het water moet, maar ze weten niet dat hier een code voor gebruikt kan worden.
- Ze beginnen zonder voorbereiding.

- Ze proberen uit de codes in volstrekte willekeur welke past en testen deze. Ook tekenen ze zelf bij op het blad.
- Ze doen maar wat.
- Ze snappen niet waarom het fout gaat.

5. Conclusie en discussie

In dit onderzoek hebben we onderzocht welke denkvaardigheden leerlingen inzetten als zij een simpele programmeertaak krijgen. De onderzoeksvraag luidde: “Welke denkvaardigheden gebruiken leerlingen tussen 6 en 12 jaar als zij in tweetallen de Ozobot programmeren om een gegeven probleem (de programmeertaak) op te lossen?” We hebben de denkvaardigheden van de leerlingen geoperationaliseerd in vijf indicatoren: probleemdecompositie, abstraheren, testen, gegevens ordenen en het fouten opsporen en oplossen. Deze denkvaardigheden zijn deelvaardigheden van computational thinking skills. In duo’s krijgen leerlingen de opdracht een eenvoudige programmeertaak met een Ozobot uit te voeren. De resultaten die wij hebben gepresenteerd bereiken leerlingen na een korte introductie in het gebruik van de Ozobot en is niet het resultaat van een langdurige interventie.
Uit ons onderzoek blijkt dat iets minder dan de helft van de duo’s het probleem oplost en dat meer duo’s uit de oudere groepen de programmeertaak oplossen dan de duo’s uit de jongere groepen. Voor drie van de vier indicatoren blijkt dat de grootste verbetering in de aanpak gebeurt tussen het plaatsen van code 1 en code 2. Dus relatief aan het begin van de programmeertaak. Alleen voor gegevens ordenen is het verschil pas significant tussen code 2 en 3. Uit ons onderzoek blijkt dat succesvolle duo’s vooral gestructureerd werken. Dit door alle informatie te verzamelen, opties te elimineren, nauwkeurig te werken en de tijd te nemen om de (mogelijke) oplossing door te nemen. De meeste duo’s (ook de succesvolle) volgen geen systematische aanpak voor het oplossen van fouten. Niet succesvolle duo’s hebben geen gestructureerde aanpak en overzien het probleem niet.
Op alle indicatoren, uitgezonderd fouten opsporen en oplossen, scoren de jongens duo’s uit groep 4 tot 6 gemiddeld hoger op de indicatoren dan meisjes duo’s en gemengde duo’s. Echter bij de groepen 7 en 8 is dit verschil weggewerkt.

Bij de duo’s uit de jongere groepen zien we dat ze bij het maken van de programmeertaak maar wat lijken te doen en soms via trial & error ergens komen. Diverse auteurs laten zien dat positieve effecten van programmeeronderwijs op probleemoplosvaardigheden (Fessakis, Gouli & Mavroudi, 2013) en creatief denken, reflectieve vaardigheden en metacognitieve taken (Clements & Gullo, 1984) bij jonge leerlingen worden bereikt als zij gerichte instructie krijgen en directe feedback op hun acties. De programmeertaak met de Ozobot was voor de jongste leeftijdsgroep in ons onderzoek waarschijnlijk te veelomvattend. Pas vanaf groep 6 zien we dat de duo’s al bij de start onderkennen dat de casus verdeeld is in subtaken en stellen de duo’s doelen voor subtaken. Ze testen met een doelgerichte opeenvolging van handelingen en ze verbinden stukjes informatie. Vanaf groep 7 en 8 formuleren leerlingen de verbindingen tussen de verschillende stukken informatie. De aanpak om fouten op te sporen en tot oplossing te brengen is bij de meeste duo’s nog niet gestructureerd. Klahr en Carver (1988) deden onderzoek naar het aanleren van strategieën om fouten op te sporen en op te lossen in een programmeeromgeving (LOGO). Zij vonden dat expliciete instructie in het opsporen en oplossen van fouten nodig is.

De verschillen tussen de gemiddelde score op de indicatoren van de jongens duo's en de meisjes - respectievelijk gemengde duo's in de jongere groepen is opvallend. Er is weinig onderzoek naar de verschillen tussen meisjes en jongens bij de aanpak van programmeertaken. In 1993 deed Yelland (1993) onderzoek naar leerlingen die gemiddeld 7.5 jaar waren. Zij liet de leerlingen ook in duo's werken en vond sekseverschillen in het voordeel van jongens duo's, maar observeerde ook dat deze verschillen wegvielen naarmate er langer aan de programmeertaak werd gewerkt. Uit deze studie bleek dat meisjes duo's geneigd zijn minder risico's te nemen. Sullivan en Bers (2013) vinden in een onderzoek waarin kleuters robots programmeren dat jongens op twee taken gemiddeld hoger scoren dan de meisjes: het gebruik van voorwaardelijke conditie (if) en het fysiek in elkaar zetten van de robot. Op de andere taken (waaronder fouten opsporen en oplossen) werden geen significante verschillen tussen jongens en meisjes gevonden. Atmatzidou en Demetriadis (2016) vinden in hun onderzoek onder leerlingen in het voortgezet onderwijs (gemiddelde leeftijd 15 jaar) ook verschillen tussen meisjes en jongens in het voordeel van jongens. Zij concluderen dat meisjes meer tijd nodig lijken te hebben om deze vaardigheden te ontwikkelen. Jeuring et al. (2016) vonden in hun reviewstudie een verschil tussen jongens en meisjes, welke grotendeels aan voorkennis en time-on-task toe te schrijven is. Om de verschillen tussen jongens en meisjes in de aanpak van een programmeertaak beter te begrijpen zou in een vervolgonderzoek de analyse van de filmpjes op dit punt wellicht meer zicht kunnen bieden.

In dit onderzoek hebben we gekeken naar de wijze waarop leerlingen een aantal CT skills inzetten om een programmeertaak aan te pakken. De vraag is echter of leerlingen dit type taken ook niet kunnen uitvoeren zonder programmeertools. Immers diverse onderzoekers (bijvoorbeeld Barr & Stephenson, 2011) geven aan dat deze vaardigheden ook zonder programmeertools kunnen worden opgelost. Vervolgonderzoek zou kunnen nagaan of het gebruik van programmeertools een meerwaarde biedt voor het aanleren van dit type vaardigheden.

Iets minder dan de helft van de duo's in onze studies losten het probleem zonder veel instructie op. Dit gold met name voor de duo's uit de hogere leerjaren. De verwachting is dat dit percentage hoger zou hebben gelegen en ook jongere leerlingen vaker het probleem hadden kunnen oplossen als er wel gerichte instructie was geweest (zie bijv. Jenson & Droumeva, 2016). Immers Voogt et al. (2017) vonden in hun review van de literatuur dat programmeeronderwijs een positief effect had als specifieke CT skills als deze worden aangeboden via gerichte instructie bij de introductie van de vaardigheid en het leerlingen de vaardigheid leren toepassen aan de hand van door de leraar geformuleerde opdrachten. Hoogstwaarschijnlijk is de noodzaak van gerichte instructie groter wanneer meer complexere CT vaardigheden moeten worden gebruikt, zoals voorwaardelijke uitvoering ('if' en 'if not'), en parallelle uitvoering ('while') (zie bijv. Atmatzidou & Demetriadis, 2016).

In ons onderzoek naar het gebruik van computational thinking skills bij het oplossen van een programmeertaak met behulp van de Ozobot hebben we gekeken naar een

aantal denkvaardigheden die leerlingen gebruiken als ze de programmeertaak oplossen. Succesvolle duo's pakken de programmeertaak gestructureerd aan:

- zij zorgen dat alle informatie helder is en stellen vragen als de instructie niet duidelijk is (abstraheren)
- ze kijken of er een code bij zit die als enige op die plek kan (probleemdecompositie)
- ze beginnen niet direct en nemen de tijd om de plaatsing van de codes door te spreken (gegevens ordenen)
- ze zien verbanden tussen codes (gegevens ordenen)
- ze elimineren de codes die al gebruikt zijn (testen)
- ze herkennen mogelijke fouten (fouten opsporen en oplossen)

Daarnaast werken succesvolle duo's zorgvuldig. Ze kalibreren de Ozobot en tekenen de codes zorgvuldig in. Leraren in het basisonderwijs kunnen bij het onderwijzen van programmeren leren van de aanpak van succesvolle duo's en deze aanpak integreren in hun didactiek.

Referenties

Atmatzidou, S., & Demetriadis, S. (2016). Advancing students' computational thinking skills through educational robotics: A study on age and gender relevant differences. *Robotics and Autonomous Systems*, *75*, 661–670. http://doi.org/10.1016/j.robot.2015.10.008

Barr, V., & Stephenson, C. (2011). Bringing computational thinking to K-12. *ACM Inroads*, *2*(1), 48. http://doi.org/10.1145/1929887.1929905

Clements, D. H., & Gullo, D. F. (1984). Effects of computer programming on young children's cognition. *Journal of Educational Psychology*, *76*(6), 1051–1058. http://doi.org/10.1037/0022-0663.76.6.1051

Fessakis, G., Gouli, E., & Mavroudi, E. (2013). Problem solving by 5-6 years old kindergarten children in a computer programming environment: A case study. *Computers and Education*, *63*, 87–97. http://doi.org/10.1016/j.compedu.2012.11.016

Grover, S., & Pea, R. (2013). Computational Thinking in K-12: A Review of the state of the field. *Educational Researcher*, *42*(1), 38–43. http://doi.org/10.3102/0013189X12463051

Hambrusch, S., Hoffmann, C., Korb, J. T., Haugan, M., & Hosking, A. L. (2009). A multidisciplinary approach towards computational thinking for science majors. *ACM SIGCSE Bulletin*, *41*, 183 - 187. http://doi.org/10.1145/1539024.1508931

Hesse, F., Care, E., Buder, J. Sassenberg, K. & Griffin, P. (2015). A framework for teachanble collaborative problemsolving skills. In P. Griffin & E. Care (Eds.). *Assessment and teaching of 21st century skills* (pp. 37-56). New York: Springer.

International Society for Technology in Education (ISTE) (nd). *Computational Thinking for all.* Retrieved from https://www.iste.org/explore/articleDetail?articleid=152&category=Solutions&article=Computational-thinking-for-all

Jenson, J., & Droumeva, M. (2016). Exploring media literacy and computational thinking: a Game Maker curriculum study. *Electronic Journal of e-Learning, 14*(2), 111-121.

Jeuring, J., Corbalan, G., Es, N. van, & Leeuwestein, H. (2016). *Leren programmeren in het PO – een literatuurreview.* Utrecht: Universiteit Utrecht.

Klahr, D., & Carver, S. M. (1988). Cognitive objectives in a LOGO debugging curriculum: Instruction, learning, and transfer. *Cognitive Psychology, 20*(3), 362–404.

Lee, I., Martin, F., Denner, J., Coulter, B., Allan, W., Erickson, J., Malyn-Smith, J., & Werner, L. (2011). Computational thinking for youth in practice. *ACM Inroads, 2*(1), 33–37. http://doi.org/10.1145/1929887.1929902

Lu, J. J., & Fletcher, G. H. L. (2009). Thinking about Computational Thinking. Categories and subject descriptors. *Sigcse*, 260–264. http://doi.org/10.1145/1539024.1508959

Lye, S. Y., Hwee, J., & Koh, L. (2014). Review on teaching and learning of computational thinking through programming: What is next for K-12? *Computers in Human Behavior, 41*, 51–61. http://doi.org/10.1016/j.chb.2014.09.012

Papert, S. (1980). *Mindstorms: Computers, children, and powerful ideas.* New York: Basic Books.

Pea, R. D., Kurland, D. M., & Hawkins, J. (1985). Logo and the development of thinking skills. *Children and Microcomputers: Research on the Newest Medium*, 194–212.

Salomon, G., & Perkins, D. N. (1989). Rocky roads to transfer: Rethinking mechanisms of a neglected phenomenon. *Educational Psychologist, 24*(2), 113–142.

doi:10.1207/s15326985ep2402_1.
Sengupta, P., Kinnebrew, J. S., Basu, S., Biswas, G., & Clark, D. (2013). Integrating computational thinking with K-12 science education using agent-based computation: A theoretical framework. *Education and Information Technologies, 18*, 351–380.
SLO (nd). *Een voorbeeldmatig leerplankader.* Retrieved from http://curriculumvandetoekomst.slo.nl/21e-eeuwse-vaardigheden/digitale-geletterdheid/computational-thinking/voorbeeldmatig-leerplankader
Sullivan, A., & Bers, M. U. (2013). Gender differences in kindergarteners' robotics and programming achievement. *International Journal of Technology and Design Education*, *23*(3), 691–702. http://doi.org/10.1007/s10798-012-9210-z
Yelland, N. (1993). Young children learning with LOGO: An analysis of strategies and interactions. *Journal of Educational Computing Research, 9*(4):465–486.
Voogt, J., Brand-Gruwel, S., & Strien, J. van. (2017). *Effecten van programmeeronderwijs op computational thinking - reviewstudie.* Zwolle, Heerlen, Amsterdam: Windesheim, Open Universiteit, Universiteit van Amsterdam.
Wing, J. M. (2006). Computational thinking. *Communications of the ACM, 49*(3), 33–35. doi:10.1145/1118178.1118215
Wolz, U., Stone, M., Pearson, K., Pulimood, S., & Switzer, M. (2011). Computational thinking and expository writing in the middle school, *ACM Transactions on Computing Education, 11*, 2, article 9.

Bijlage 1

Het in dit onderzoek gebruikte observatie-instrument is tot stand gekomen op basis van een vergelijking van het voorbeeldmatig leerplankader van de SLO met de rubric 'Cognitive skills in collaborative problem solving' (Hesse, Care, Buder, Sassenberg & Griffin, 2015).

In het voorbeeldmatig leerplankader van SLO zijn de indicatoren geselecteerd die van toepassing zijn op de programmeertaak die binnen dit onderzoek centraal staat. Deze indicatoren zijn:

Computational thinking	De leerling…
Problemen (her)fomuleren	Kan op een zodanige manier problemen formuleren dat het mogelijk wordt om het probleem op te lossen door gebruik van een computer of ander gereedschap
	Kan mogelijke oplossingen analyseren om de meest kansrijke richting te bepalen
Gegevens analyseren	Kan gegevens logisch ordenen en begrijpen
	Kan patronen vinden en conclusies trekken
Probleem decompositie	Kan een taak opdelen in kleinere taken
	Kan een aantal taken combineren tot één taak
Abstractie	Kan complexiteit reduceren en algemene concepten overbrengen
	Kan oplossingen automatiseren door middel van algoritmisch denken
Algoritmes en procedures	Kan een proces om problemen op te lossen generaliseren, zodat het ook bij andere problemen toegepast kan worden
Automatisering	Kan door het opstellen van een serie van geordende stappen een probleem oplossen of een bepaald doel bereiken
	Kan mogelijke oplossingen identificeren, analyseren en implementeren met als doel de meest effectieve en efficiënte oplossing te vinden
	Kan repetitieve taken laten uitvoeren door computers
Simulatie en modellering	Kan een probleemoplossing generaliseren en toepassen op andere problemen

Vervolgens hebben wij in kleuren gemarkeerd in hoeverre de indicatoren in de relevante onderdelen van de rubric 'Cognitive skills in collaborative problem solving' (Hesse, Care, Buder, Sassenberg & Griffin, 2015) overeenkomen met de indicatoren in het voorbeeldmatig leerplankader van SLO. Deze indicatoren hebben dezelfde kleur gekregen.

Zoals de kleurendekking laat zien ondervangen de geselecteerde onderdelen uit de rubric 'Cognitive skills in collaborative problem solving' (Hesse, Care, Buder, Sassenberg & Griffin, 2015) de indicatoren uit het voorbeeldmatig leerplankader van SLO die voor ons onderzoek van belang zijn.

Indicator: Analyseert en beschrijft het probleem in eigen woorden	**Laag: 0** Probleem is onderkend zoals het zich voordoet.	**Middel: 1** Probleem is onderverdeeld in subtaken.	**Hoog 2** Herkent de noodzakelijke sequentie van subtaken.
Stelt een duidelijk doel voor een taak.	Stelt een algemeen doel zoals taak afronding.	Stelt doelen voor subtaken.	Stelt doelen waarbij de relatie tussen subdoelen erkend wordt.
Implementeert mogelijke oplossingen voor een probleem en bekijkt de voortgang.	Trial & Error handelingen: uitproberen.	Doelgerichte opeenvolging van handelingen.	Systematisch alle mogelijke oplossingen verkennen.
Herkent relaties en verbindingen tussen elementen.	Focust op losse stukjes informatie.	Verbindt stukjes informatie.	Formuleert verbindingen tussen verschillende stukken informatie.

Na testen met een groep leerlingen is de rubric uitgebreid met 2 categorieën en is er een indicator fouten opsporen en oplossen aan toegevoegd. Tevens hebben wij in de rubric per regel benoemd welke CT skill beoordeeld wordt. Voor de herkenbaarheid hebben wij in het observatie-instrument dezelfde kleuren gehanteerd als hierboven in het vergelijk.

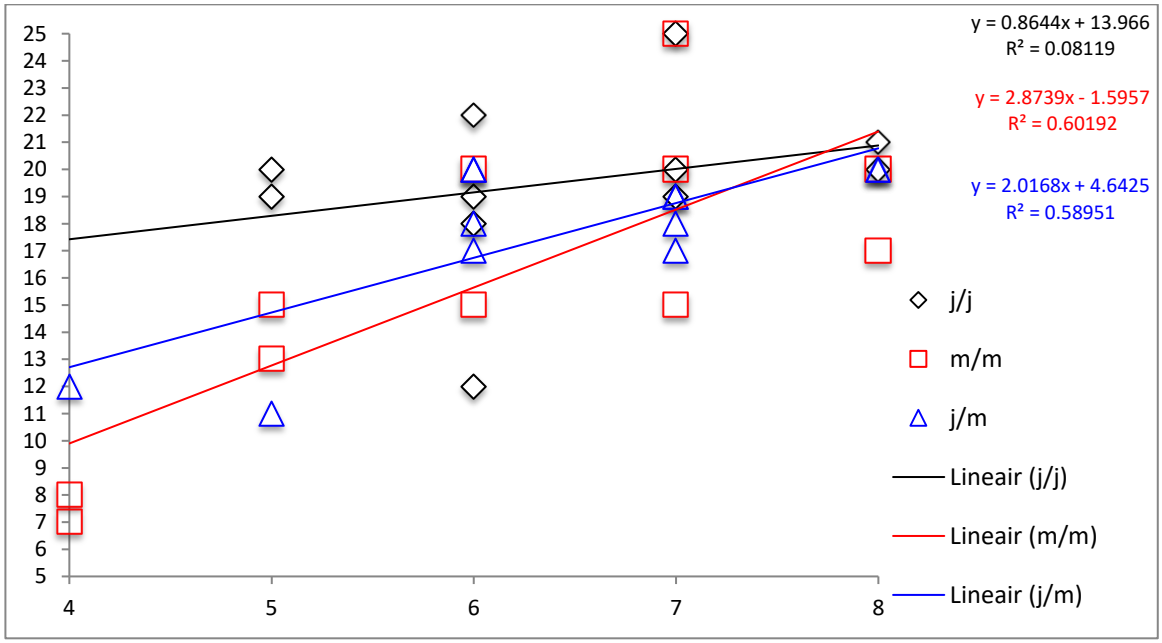
Indicator	Zeer Laag: 1	Laag: 2	Middel: 3	Hoog 4	Zeer hoog 5
Analyseert en beschrijft het probleem in eigen woorden.	Probleem wordt niet onderkend zoals het zich voordoet.	Probleem is onderkend zoals het zich voordoet.	Probleem is onderverdeeld in subtaken.	Herkent de noodzakelijke sequentie van subtaken.	Herkent relaties en verbindingen van subtaken en generaliseert de oplossing.
CT skill: Probleem-decompositie	*De leerlingen snappen bijv. niet dat er een code gebruikt kan worden.*	*De leerlingen snappen bijv. dat Ozobot over water moet, maar weten niet dat hier een code voor gebruikt kan worden*	*De leerlingen snappen bijv. dat Ozobot over water moet en begrijpen dat dit een spring code moet zijn*	*De leerlingen snappen bijv. dat Ozobot over water moet en beredeneren de benodigde springcode(s)*	*De leerlingen beredeneren de volledige juiste plaatsing van codes.*
Stelt een duidelijk doel voor een taak.	Ze stellen geen doelen.	Stelt een algemeen doel zoals taak afronding.	Stelt doelen voor subtaken.	Stelt doelen waarbij de relatie tussen subdoelen erkend wordt.	Stelt doelen waarbij de gehele relatie tussen alle subdoelen erkend wordt.
CT skill: abstraheren	De leerlingen weten niet goed hoe ze moeten beginnen aan deze taak	De leerlingen beginnen bijv. zonder voorbereiding	De leerlingen lezen eerst de codes door en beginnen met een willekeurige code	De leerlingen lezen de codes door en beginnen gericht met een eerste code	De leerlingen zoeken eerst alle benodigde codes op en beredeneren de volgorde

Indicator	Zeer Laag: 1	Laag: 2	Middel: 3	Hoog 4	Zeer hoog 5
Implementeert mogelijke oplossingen voor een probleem en bekijkt de voortgang.	Ze beginnen, maar passen de opdracht niet toe.	Trial & Error handelingen: uitproberen.	Doelgerichte opeenvolging van handelingen.	Systematisch alle mogelijke oplossingen verkennen.	Past de gekozen probleemoplossing systematisch toe op alle subtaken en test de werking
CT skill: testen	De leerlingen gebruiken bijv. niet geselecteerde codes, tekenen bijv. zelf op het blad	De leerlingen proberen uit de geselecteerde codes in volstrekte willekeur welke past en testen deze (soms)	De leerlingen plaatsen doelgericht een code en testen deze altijd	De leerlingen redeneren verder vanaf de geplaatste code en testen deze	Alle codes worden direct ingekleurd en de werking wordt getest.
Herkent relaties en verbindingen tussen elementen.	Focust niet op de informatie:	Focust op losse stukjes informatie.	Verbindt stukjes informatie.	Formuleert verbindingen tussen verschillende stukken informatie.	NVT
CT skill: gegevens ordenen	De leerlingen doen maar wat	Pakt een willekeurige code en gaat hiermee aan de slag	Pakt een willekeurige code, gaat hiermee aan de slag en herkent dan een vervolgcode	Beredeneert welke code logisch volgt op de startcode en vandaaruit een volgende code	
CT skill: fouten opsporen en oplossen	Snapt niet waarom het fout gaat	Herkent één mogelijke reden van de fout. De aanpak om tot foutoplossing te komen is nog niet gestructureerd	Herkent meerdere of zelfs alle mogelijke redenen van de fout. De aanpak om tot foutoplossing te komen is nog niet gestructureerd	Doorloopt stap voor stap alle mogelijke oorzaken voor de fout en probeert deze uit te sluiten	Er is geen foutoplossing nodig. De programmeer-taak is juist gemaakt.

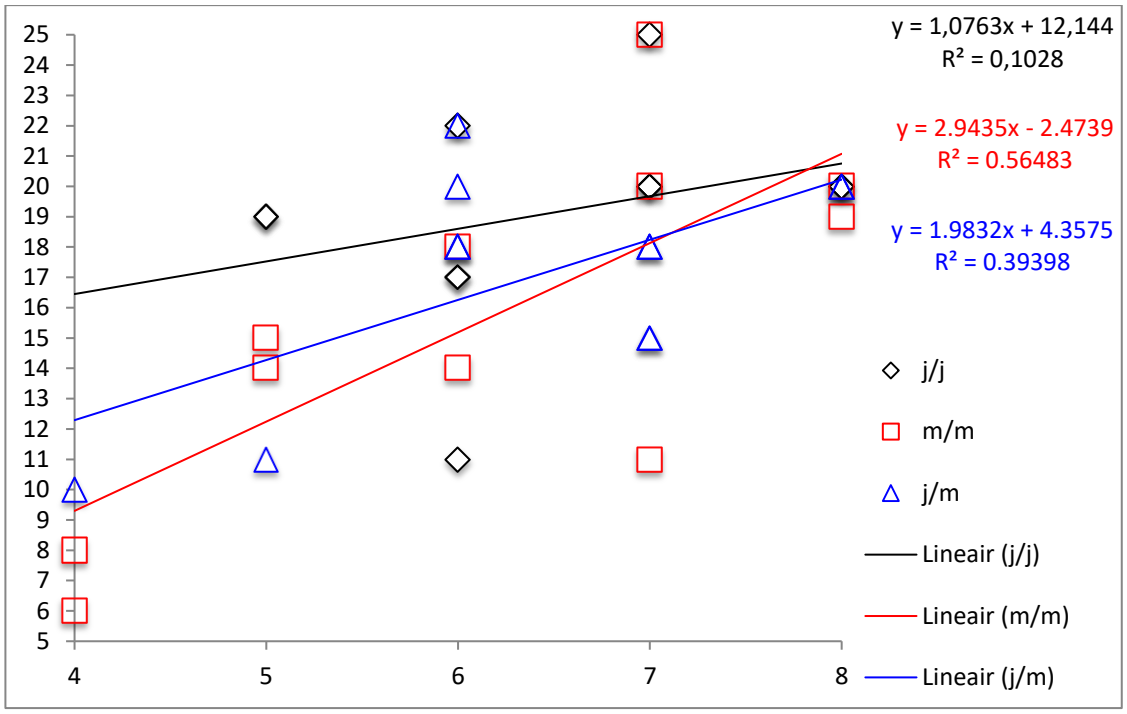
Bijlage 2

Indicator	toetswaarde (Z)	p	effectgroote (r)	verschillen
Probleemdecompositie	3.153	0.002	0.54	code 1 – code 2
Abstraheren	2.707	0.007	0.46	code 1 – code 2
Testen	1.984	0.047	0.34	code 1 – code 2
Gegevens ordenen	2.530	0.011	0.43	code 2 en 3

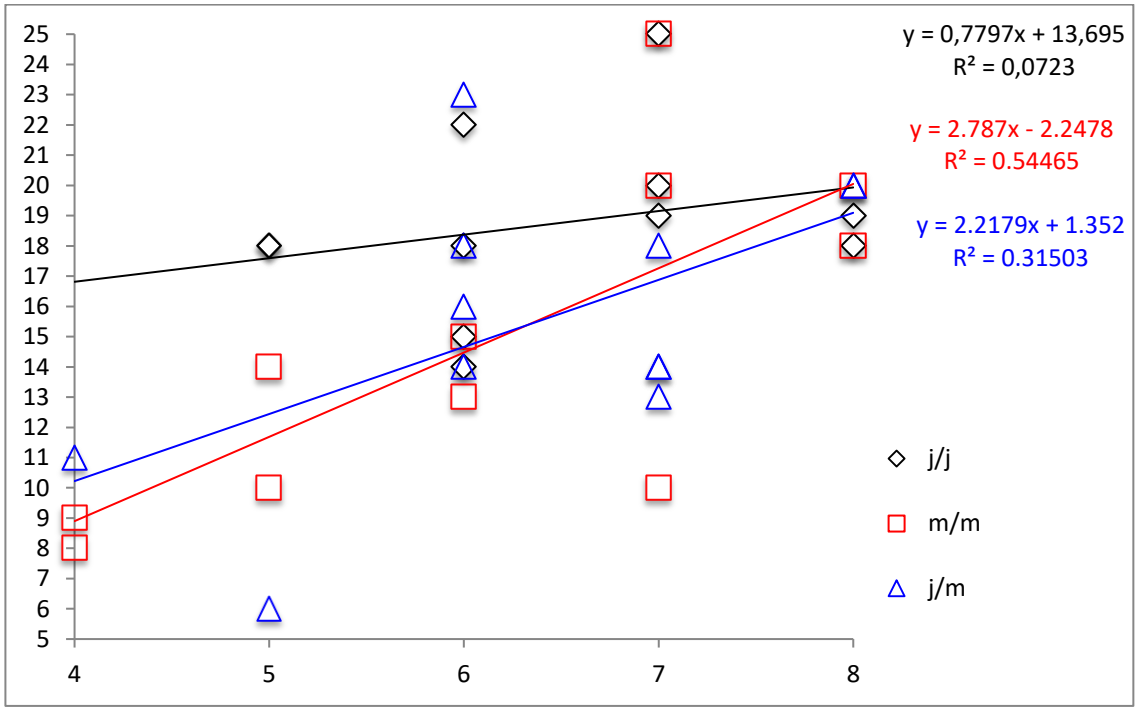
Tabel 1: Significante verschillen in de gemiddelde score op opeenvolgende codes. Gebaseerd op de Wilcoxon signed rank test.



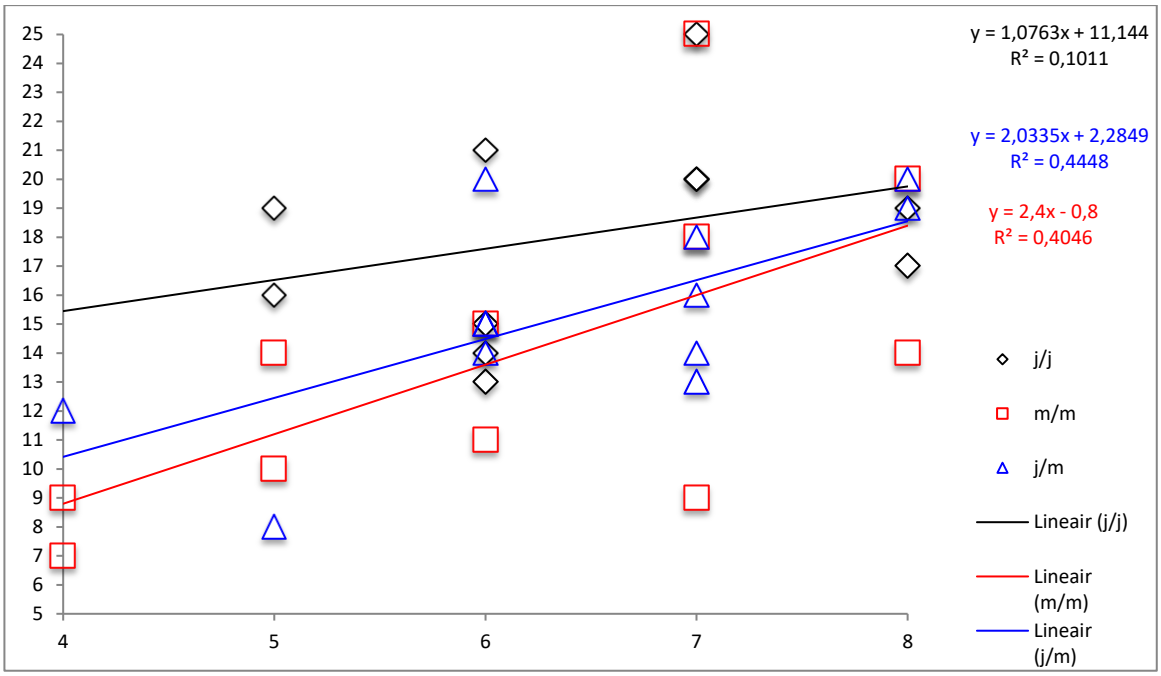
Figuur 1: Probleemdecompositie: Behaalde score per duo (jongens/jongens (j/j); meisje/meisje (m/m) en gemengd (j/m)) per groep. Scatterplot, regressielijnen, regressievergelijking en verklaarde variantie



Figuur 2: Abstraheren: Behaalde score per duo (jongens/jongens (j/j); meisje/meisje (m/m) en gemengd (j/m)) per groep. Scatterplot, regressielijnen, regressievergelijking en verklaarde variantie



Figuur 3: Testen: Behaalde score per duo (jongens/jongens (j/j); meisje/meisje (m/m) en gemengd (j/m)) per groep. Scatterplot, regressielijnen, regressievergelijking en verklaarde variantie



Figuur 4: Gegevens ordenen: Behaalde score per duo (jongens/jongens (j/j); meisje/meisje (m/m) en gemengd (j/m)) per groep. Scatterplot, regressielijnen, regressievergelijking en verklaarde variantie

WWW.WINDESHEIM.NL

Christelijke Hogeschool Windesheim
Campus 2-6 | Postbus 10090 | 8000 GB Zwolle
T 0900 8899 | www.windesheim.nl