Kiezen voor slecht informatica-onderwijs

J.B. van Meurs/ B.J. Koers
R.U. Groningen

Samenvatting

In een artikel van mei 1983 werd een nogal ontmoedigend beeld gegeven van de stand van zaken in het Nederlandse computer-onderwijs. Betreurd werd met name het feit dat bijna iedere school zeer eenvoudige apparatuur aanschafte, waar bijna niets mee te beginnen is.
Hermsen koos voor het UNIX Time-sharing Systeem, maar stelde tegelijk voor maar slecht te beginnen met de al aanwezige apparatuur.
De auteurs van dit artikel delen die mening niet, zij vinden: "Beter geen computer-onderwijs dan slecht computer-onderswijs".

Summary

In an earlier article (May, 1983) Hermsen proposed to use the UNIX Time-sharing System to teach computer-literacy. Regrettably, most schools have bought very simple micro computers, which are economical, but unsuitable for educational purposes. Hermsen proposed to try to make the best of it under those circumstances.
The present authors disagree with this: they prefer no computer-education at all if it is not possible to teach it the correct way.

Een ontmoedigende conclusie die wel getrokken moet worden, na het lezen van met name het themanummer van de Nieuwe Wiskrant over microcomputers (mei 1983). Dit themanummer vormde de aanleiding tot het schrijven van dit artikel.
Het is niet onze bedoeling de schrijvers van te noemen artikelen persoonlijk te bekritiseren, maar we constateren, dat hun meningen helaas veel algemener voorkomen bij docenten en andere mensen uit de onderwijswereld. In dit artikel leveren we kritiek op de huidige gang van zaken met betrekking tot het computeronderwijs en geven we tevens aan in welke richting onzes inziens oplossingen gezocht moeten worden. Wordt er in Nederland gekozen voor slecht onderwijs, of kiezen we voor goed onderwijs, centraal ontwikkeld en getest door een team van mensen uit de onderwijswereld en informatici? Natuurlijk is dit laatste geen absolute garantie voor goed onderwijs, maar in de huidige situatie, waarbij lokaal allerlei ad hoc lespakketten worden samengesteld, is die garantie zeker niet te geven.

Basic?

Allereerst willen we opmerken, dat we van harte instemmen met de veranderende visie op het informatica-onderwijs. Waren docenten en onderwijsdeskundigen enkele jaren geleden nog zeer moeizaam te overtuigen van de beperkingen van en bezwaren tegen wat we maar "BASIC-onderwijs" zullen noemen, nu zien we gelukkig, dat in vrijwel alle publikaties over informatica-onderwijs gesproken wordt over toepassingsprogrammatuur, gestructureerd programmeren en simuleren. BASIC lijkt het als onderwijsobject te gaan verliezen.
Enkele citaten uit de Nieuwe Wiskrant van mei 1983:

[Vonk 83]
"Computers in de hand te houden zal een belangrijke taak zijn van deze en de komende generatie...".
"Men zou dus datgene moeten onderwijzen dat onze leerlingen, na het verlaten van de school, geschikt maakt om met automatisering te leven en deze in de hand te houden".

[Herm 83a]
Over zeer eenvoudige apparatuur, die tegenwoordig helaas vaak door scholen wordt gekocht:
"Nooit en te nimmer valt daarmee iets anders te beginnen dan het verwerken van programma's in een bij de machine behorende BASIC-variant".

[Herm 83b]
"Omdat programmeren het beste kan gebeuren in talen die een goede uitdrukkingsmogelijkheid hebben voor het beschrijven van algoritmen, is BASIC een slechte kandidaat voor onderwijskundige doeleinden".

Duidelijk is ook het advies dat een werkgroep ter introductie van de burgerinformatica op de middelbare school heeft uitgebracht: "vermijdt gebruik BASIC in burgerinformatica". Aldus luidt de titel van een artikel over dit advies in de Automatiseringsgids van 29 juni 1983. Wij zijn het hier volledig mee eens.

Teleurstellend is echter, dat men ondanks goede onderwijsdoelstellingen, onderwerpkeuze en machinekeuze, zeer gemakkelijk afwijkt van zijn ideaal en toegeeft aan (meestal) financiële beperkingen. Hermsen doet in het bijzonder in zijn artikel "Een computer kopen, maar welke?" [Herm 83c] veel water bij de op zichzelf prima wijn. Na het opsommen van onderwijskundige en technische eisen, te stellen aan een computersysteem zegt hij: "Er is mijns inziens slechts een systeem dat het best aan deze alleszins redelijke eisen voldoet: het UNIX Time-sharing Systeem". Volgens hem moeten scholen niets kopen zolang het Ministerie van Onderwijs nog niet gekozen heeft voor een Operating System dat voor het onderwijs geschikt is en op alle scholen gebruikt zou moeten worden, nl. dit UNIX systeem. Iets wat we volledig onderschrijven, zie "Kennismaking met de computer" [Koers 82].
Maar zodra de financiën om de hoek komen kijken lijkt het of voor Hermsen dit ideale operating system niet meer zo belangrijk is. Het is geen haalbare kaart, want het is te duur. Zo komt hij zelfs tot een advies voor aanschaf van goedkope BASIC apparatuur ter overbrugging. Terwijl hij even te voren gezegd heeft, dat het slecht is en derhalve ongeschikt voor het onderwijs. [Herm 83a]: "het al te gemakkelijk accepteren van technische beperkingen grijpt *te veel* op het onderwijs in".

Het lijkt er op, dat we de huidige ontwikkelingen in het onderwijs als volgt moeten samenvatten: "Beter slecht onderwijs dan geen onderwijs". De vraag kan zelfs gesteld worden, of men wel de leerlingen voor ogen heeft als er met computeronderwijs begonnen wordt. Of gaat het er misschien om dat een leraar zijn hobby kan uitoefenen? Gaat het om het imago van de school, die leerlingen vreest kwijt te raken als er geen computeronderwijs is?

Het belang van de leerling is primair. Slecht onderwijs is schadelijk. Tegenover het "Beter slecht onderwijs dan geen onderwijs" poneren wij dan ook de stelling "Beter geen onderwijs dan slecht onderwijs". Laten de enthousiastelingen liever een jaar of twee wachten en ondertussen werken aan de ontwikkeling van goed onderwijs. Ongetwijfeld zal het bedrijfsleven in die tijd goede apparatuur en programmatuur kunnen ontwikkelen en tegen betaalbare prijzen kunnen aanbieden. Ondertussen is het onzes inziens heel goed mogelijk al vast met computeronderwijs te beginnen zonder dat de school eigen apparatuur heeft. Het project computerkunde bv. van het Onderwijs Computercentrum functioneert goed [Vonk 83].

Aspecten van computeronderwijs

Als een school iets aan computeronderwijs wil doen, komen daar in het algemeen drie aspecten aan de orde:

A. het demonstreren van en laten werken met standaard toepassingsprogrammatuur;
B. het gebruik van de computer bij het oplossen van voor de gebruiker nieuwe problemen;
C. het leren programmeren.

Het onderwerp toepassingsprogrammatuur kunnen we de *aardrijkskunde* van het computeronderwijs noemen. Bij dit onderdeel leert men de toepassingsmogelijkheden van de computer kennen, van spelletjes tot administratieve systemen, van robots tot betalingsverkeer, van auto tot wasmachine. Kenmerk van de onderwerpen is, dat ze sterk aan verandering onderhevig zijn. Een goed boek hierbij is het dit jaar bij Stenfert Kroese verschenen "Basisboek Informatica" geschreven door prof. dr. A.J. van der Pool. Het is beschrijvend van karakter en geeft uitstekend weer waar en hoe de computer in onze maatschappij aanwezig is. Opgemerkt dient te worden dat het bij dit onderdeel gaat om *passieve* kennis van het gebruik van computers en dat er geen voorkennis voor nodig is. Het kan gegeven worden in de vorm van verhalen, films of demonstraties. Het laatste kan het onderwijs wel duur maken. Een goed alternatief is dan het opgeven van leeswerk.

Bij het tweede aspect, de probleem-analyse en het algoritmisch denken, wordt de *actieve* kennis van de leerling op computergebied vergroot. Noem dit maar de *natuurkunde* van het computeronderwijs. De leerlingen zullen hierbij standaardprogrammatuur ter beschikking moeten krijgen en met behulp hiervan problemen oplossen. Typische onderwerpen zijn hier tekstverwerking, simulatie, spreadsheets en grafische toepassingen als turtle geometry en Karel de robot. Een voorbeeld hiervan is te vinden in [Koers 82], waar een practicum van ongeveer acht bestedingsuren beschreven wordt.

Het derde aspect, het leren programmeren, levert tenslotte de bouwstenen van het probleemoplossen en noemen we de *wiskunde* van het computeronderwijs. Hier moet men inzicht krijgen in de structuur van algoritmen. Bij het programmeeronderwijs mag het niet gaan om puntkomma's of declaraties, maar moeten de centrale begrippen van het programmeren aan de orde komen. Men moet inzicht krijgen in het effect van het sequentieel uitvoeren van instructies, het uitvoeren afhankelijk van bepaalde condities en het herhaald uitvoeren totdat aan een bepaalde conditie al dan niet voldaan is. Bij het laatste is vooral belangrijk het letten op de beëindiging van de herhaling. Ook aan de correctheid van programma's moet aandacht geschonken worden, voordat deze eventueel aan een computer aangeboden worden. Bij de vergelijking tussen programmeren en wiskunde moet vooral gedacht worden aan de methode van werken in de meetkunde. Evenals bij het programmeren heeft men in de meetkunde slechts een kleine hoeveelheid begrippen, waar een probleem mee moet worden opgelost. De leerling zal te maken krijgen met het "zien" van de oplossing. Daarna moet het bewijs nog geleverd worden. De didaktiek van het programmeren zal dan ook veel overeenkomst vertonen met de didaktiek van de meetkunde. De leraar zal voor hetzelfde probleem staan: hoe breng ik de leerling inzicht bij.
Naar onze mening moet de nadruk bij het computeronderwijs op het tweede aspect liggen. Aan computerliteracy hoeft niet veel aandacht te worden besteed, daar de leerling hiermee steeds meer en wellicht voldoende geconfronteerd wordt via de media. In dat

geval is speciaal onderwijs niet nodig. Het programmeren is een veel wezenlijker onderdeel van de informatica, maar er zal waarschijnlijk te weinig tijd vrijgemaakt kunnen worden voor goed onderwijs hierin. Men mag de leerling niet de illusie geven dat programmeren in een achternamiddag geleerd kan worden. Op onze universiteit heeft een eerste cursus een bestedingsduur van 80 uur en dat wordt al ervaren als een minimale hoeveelheid. Men heeft dan als het ware net het zwemdiploma A behaald.
Ons lijkt het belangrijkste van het informatica-onderwijs het leren gebruiken van de computer in de werkelijke maatschappelijke praktijk. Daarin zal het niet zo zijn dat de leerling echt zal moeten programmeren. Wel zal hij bij allerlei zaken de computer tegenkomen en dan de machine goed moeten kunnen gebruiken. Hierbij komen wel aspecten van het programmeren naar voren, maar niet in zo'n abstracte vorm als bij het werkelijke programmeren. De talen waarmee gewerkt moet worden zijn doelgericht. Denk bijvoorbeeld aan een tekstverwerkingstaal met commando's als verwissel, sorteer, vervang, zoek; een turtle taal met de commando's vooruit, draai, herhaal; of denk aan een tabelverwerkingsprogramma. De gekozen toepassingen moeten aansluiten bij de dagelijkse omgeving van de leerling en moeten de leerlingen uitsluitend laten kennismaken met de karakteristieken van het computergebruik.

Dat dit iets heel anders is dan leren programmeren in BASIC, ELAN of PASCAL zal duidelijk zijn. Programmeren kan natuurlijk heel zinvol zijn, maar dan denken we meer in de richting van het werken met procedures en het maken en veranderen daarvan. Dat computeronderwijs vaak neerkomt op een BASIC-cursus, wordt meestal toegeschreven aan een gebrek aan geld. Maar het zou wel eens kunnen zijn dat het programmeren een vlucht is. Dat de leraar door gebrek aan kennis of aan tijd niet in staat is een goede cursus toepassingsprogrammatuur op te zetten. Veel gemakkelijker is het om naar de winkel te stappen en de leerlingen een BASIC-boekje door te laten werken. En dan heeft hij nog gelijk ook, want het is natuurlijk onzin dat alle docenten onafhankelijk van elkaar cursussen gaan maken. Omdat het onlangs begonnen "100-scholenproject" daar juist op gebaseerd is vrezen we dat het informatica-onderwijs hier totaal niet mee gediend zal zijn. Er had eerst gepraat moeten worden over de algemene doelstellingen van het informatica-onderwijs. Nu zien we dat er alleen apparatuur op de scholen gedumpt is, zonder verdere begeleiding. De leraren moeten zelf maar zien wat ze er mee doen en wat ze met het vak burger-informatica willen bereiken.

Doelstellingen

Natuurlijk is er de afgelopen jaren al wel veel nagedacht over (de) doelstellingen van het computeronderwijs. Dat is een goede zaak, maar het mag niet tot gevolg hebben, dat nieuwe leraren informatica volstaan met de opmerking dat het geven van computeronderwijs een logisch gevolg is van maatschappelijke ontwikkelingen.

Voorbereiding op de maatschappij is één van de drie hoofddoelstellingen van het onderwijs (naast beroepsvoorbereiding en persoonlijke ontwikkeling). Dit wil echter nog niet zeggen dat alles wat in de maatschappij verandert ook automatisch het onderwijs doet veranderen.
Een goed voorbeeld van een instrument, dat de maatschappij zeer beïnvloed heeft en er niet meer uit is weg te denken, is de auto. Toch beperkt de invloed van de auto binnen het onderwijs zich tot de technische opleidingen (van LTS tot TH). Daarnaast kennen we de specifieke gebruikerscursus, namelijk de rijles. Een ander zeer bekend apparaat is de zakrekenmachine. Deze wordt overal gebruikt, tot op de eindexamens toe. Toch heeft men het niet nodig gevonden op school les te geven in de bouw van het apparaat, in de techniek, in de micro-electronica, waarmee bv. de sinus van een getal berekend wordt. Het enige onderwijs dat gegeven wordt betreft het leren gebruiken van de machine.

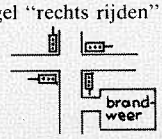
Zo moet er aangaande de computer ook goed nagedacht worden over de vraag wat voor invloed deze machine op het onderwijs moet hebben. Wij zijn van mening, dat wat betreft het computeronderwijs er evenals bij bovengenoemde voorbeelden een goede splitsing te maken is tussen gebruikerscursus en techniek. De gebruikerscursus hoort in het algemeen vormend onderwijs, de apparatuurkennis in het technisch onderwijs. Wij vinden het dan ook onjuist, dat de Programmacommissie PAO-Informatica in haar voorstel voor leerstofinhoud informatica in het voortgezet onderwijs [PAO 93] zoveel nadruk legt op deze technische zaken. De commissie schrijft wel dat voor sommige zaken (halfgeleider componenten, schakelingen en micro-architectuur) "slechts summiere behandeling wenselijk" is, maar wij denken dat onderwijs hierin in het geheel niet zinvol is. Juist een summiere behandeling kan grote verwarring stichten en tot gevolg hebben dat leerlingen afhaken en na een leuke "toepassingsfase" denken: ik snap blijkbaar nog niks van computers. Om de vergelijking met de auto door te trekken: Bij de autorijscholen gaat men wat apparatuur betreft ook niet verder dan "er is een motor, die voor of achterin zit" en "hij loopt op benzine en niet op water". Zo kunnen we het wat betreft computeronderwijs laten bij "er is een geheugen en een verwerkingseenheid". Uiteraard dient de docent informatica wel goed op de hoogte te zijn van de opbouw van de apparatuur. Ten eerste om te weten waarmee hij werkt, ten tweede om op lager niveau onderwijsondersteunende programmatuur te kunnen schrijven en laten functioneren, ten derde om vragen van geïnteresseerde leerlingen juist te kunnen beantwoorden.

Wat zouden dan wel de doelstellingen van het computeronderwijs moeten zijn? Een heel algemene doelstelling, waar iedereen het mee eens zal zijn luidt:

> De leerling dient inzicht te krijgen in wat wel en niet met een computer gedaan kan worden.

We zullen een voorbeeld geven van een probleem, waarvan de leerling moet kunnen inzien dat de computer bij de oplossing kan worden ingeschakeld. Tevens moet hij kunnen aangeven hoe dit dient te gebeuren.

Op een kruising van twee gelijke wegen vinden veel ongelukken plaats. Dit gebeurt ondanks het feit dat het systeem van de Nederlandse verkeersregels (rechts gaat voor, rechtdoor gaat voor etc.) op zichzelf goed werkt. Ongelukken gebeuren, zodra mensen risico's gaan nemen, bv. omdat men vindt dat men te lang moet wachten. Daarom wil de buurtvereniging met het oog op de veiligheid van de kinderen nu stoplichten geplaatst zien. Zoals bekend zetten stoplichten de normale verkeersregels buiten spel en wordt het verkeer door lichten geregeld. De regel "rechts rijden" blijft uiteraard wel gelden.



Opgaven:

a. Welke minimale eisen moeten gesteld worden aan het stoplichten-systeem?
b. Welke beperkingen laat je toe?
c. Introduceer voor het programmeren van de stoplichten verschillende toestanden. Welke overgangen zijn er tussen de toestanden van het systeem?
d. Schrijf een programma dat de stoplichten bedient en o.a. gebruik maakt van de aanwezige procedures
   – stoplicht (nummer, kleur) # zet een bepaald stoplicht op een bepaalde kleur #
   – wacht(tijd) # laat de toestand van het systeem onveranderd gedurende de opgegeven tijd #
e. Aan een van de toevoerwegen staat een brandweerkazerne. Welke extra maatregelen zijn nodig voor het stoplichten-systeem om een brandalarm veilig te verwerken? Richt je programma hiervoor in.

Oplossingen:

a. In eerste instantie eisen we bij een stoplichtensysteem twee dingen:
   1. er mogen geen botsingen plaatsvinden;
   2. van alle zijden moet het verkeer binnen redelijke tijd het kruispunt kunnen passeren.

   Hieruit leiden we verdere eisen af:
   3. er mag slechts van een zijde tegelijk gereden worden (twee zijden gebeurt ook wel, maar dit kan nog gevaar opleveren);
   4. er moet voldoende tijd zitten tussen twee verkeersstromen, nodig om het kruispunt vrij te kunnen maken;
   5. er moet een maximale wachttijd per zijde zijn.

b. Bij een zo eenvoudig mogelijk systeem laten we de volgende beperkingen toe:
   1. geen voorsorteer vakken met aparte stoplichten;
   2. geen splitsing auto's/fietsen en bromfietsen;
   3. geen voetgangers-stoplichten;
   4. geen afhankelijkheid van het verkeersaanbod van de diverse zijden (denk aan contacten in het wegdek, die verkeer kunnen opmerken);
   5. gelijke "groentijden" voor alle zijden;
   6. geen voorzieningen voor onderbreking van het stoplichtenprogramma.

c. De toestanden per zijde zijn rood/groen/oranje. De toestanden van het hele systeem van 4 zijden kunnen we in een tabel zetten, waarbij vertikaal de toestanden af te lezen zijn.

zijde	kleuren										
1	r	g	o	r	r	r	r	r	r	r	r
2	r	r	r	r	g	o	r	r	r	r	r
3	r	r	r	r	r	r	r	g	o	r	r
4	r	r	r	r	r	r	r	r	r	g	o
	(1)			(2)			(3)			(4)	

Merk op dat er 4 toestanden r r r r zijn. Dit is nodig omdat in de toestand r r r r onthouden moet worden wie er aan de beurt is. De toestandsovergangen zijn in de tabel te vinden door van links naar rechts de kolommen langs te gaan. Daarna repeterend.

d. Noem de zijden 1, 2, 3, resp. 4. Deze nummering is vrij te kiezen, bv. rechtsom.

```
hoofdprogramma:
  start:
  HERHAAL:
    laatrijden(zijde);
    zijde := zijde + 1;
    ALS zijde > 4 DAN zijde := 1
  EINDHERHAAL
```

hierin moeten de volgende verfijningen nog gedefinieerd worden:

```
start:
  VOOR zijde IS 1,2,3,4 DOE stoplicht( zijde, rood);
  zijde := 1

laatrijden(zijde): stoplicht( zijde, groen );
                   wacht( tijdgroen );
                   stoplicht( zijde, oranje );
                   wacht( tijdoranje );
                   stoplicht( zijde, rood );
                   wacht( vrijezone )
```

Nu moeten nog de keuzes voor de tijden gemaakt worden:

```
tijdgroen:    15 sec
tijdoranje:    3 sec
vrijezone:     5 sec
```

Deze keuzes moeten binnen start gedefinieerd worden.
De wachttijd per zijde is nu 3× (15+3+5) +1× extra vrijezone = 74 sec.

e. De eis is, dat zodra er brandalarm is het kruispunt vrijgemaakt wordt voor de brandweer. Dit kan op meerdere manieren:
   1. Alles op rood, brandweer rijdt door
   2. 3 zijden rood, zijde brandweer groen.
      In dit geval rijden de auto's wel voor de kazerne langs. Dit kan opgelost worden via een extra stoplicht direct voor de kazerne. Dit licht staat alleen op rood als er brandalarm is. Anders groen.
   3. We moeten nog kiezen of we bij brandalarm alles direct op rood zetten of dat de cyclus van de rijdende zijde gewoon afgemaakt wordt. M.a.w.: Is de overgang groen – > rood in dit geval acceptabel?

We programmeren geval 1 met normale afhandeling. De keuze voor normale afhandeling wordt gemotiveerd door de overweging dat er voldoende tijd zit tussen moment van alarm en moment van uitrukken. Introduceer de functie brandalarm, die werkt op een knop in de kazerne:

```
brandalarm:
  ALS knopingeduwd DAN ja ANDERS nee
```

Het hoofdprogramma passen we nu zodanig aan, dat een cyclus afhankelijk wordt van het brandalarm.

```
hoofdprogramma:
  start:
  HERHAAL:
    ZOLANG brandalarm DOE wacht( 5 sec );
    laatrijden( zijde );
    zijde := zijde + 1;
    ALS zijde > 4 DAN zijde := 1;
  EINDHERHAAL
```

Aan start voegen we nog toe dat er in het begin geen brandalarm is.

```
start:
  VOOR zijde IS 1,2,3,4 DOE stoplicht (zijde, rood);
  zijde := 1;
  tijdgroen := 15 sec;
  tijdoranje := 3 sec;
  vrijezone := 5 sec;
  brandalarm := nee
```

Een man gaat in verband met maagklachten naar een huisarts. Na wat praten schrijft deze bepaalde medicijnen voor. Aan het einde van het bezoek blijkt dat de man al drie jaar hartpatient is en dagelijks al andere medicijnen gebruikt. Hij had dit vergeten te zeggen, omdat hij daar al helemaal aan gewend was. De huisarts gaat nu nadenken of hij de zojuist voorgeschreven medicijnen wel kan geven. Het is immers bekend, dat bepaalde combinaties van medicijnen schadelijk en zelfs gevaarlijk zijn. Nu is het voor de huisarts niet te doen om alle effecten van alle medicijnen op elkaar uit het hoofd te kennen. Hij overweegt een computer aan te schaffen. Heeft hij daar iets aan? Wat voor programma's heeft hij nodig? Welke gegevens moeten er opgeslagen kunnen worden?

Deze vragen dient een leerling te kunnen beantwoorden. Verder dient hij het inzicht te verwerven, dat heel algemene vragen (Is moeder vanavond thuis?) niet beantwoord kunnen worden, tenzij de antwoorden expliciet in het geheugen opgeslagen zijn.

De bovengenoemde algemene doelstelling kan in vele richtingen geconcretiseerd worden. In het PAO-structuurplan wordt dit gedaan door de leerstof in vier hoofdgebieden te verdelen. Voor de volledigheid noemen we ze even en geven we de inhoud zeer beknopt aan:

A. Maatschappelijke plaatsbepaling en gevolgen:
gevolgen van opslag van gegevens; veranderende banen, werkmethoden, werkgelegenheid; invloed op mens en samenleving; kosten/baten-aspecten.

B. Toepassingen:
actief bezig zijn met de gebruiksmogelijkheden van de computer, o.a.: tekstverwerking, informatiesystemen, procesbesturing, onderwijs.

C. Probleemanalyse en programmeren:
problemen analyseren, oplossingsstrategie bedenken via systematische verfijning, ontwerpen van algoritmen, programmeren in beschermde omgeving.

D. Principes van opbouw van programmatuur en apparatuur:
processor, geheugen, micro-electronica, diverse machine-niveau's, assembly, operating systems, vertalers, netwerken.

Duidelijk zal zijn dat we bij A onze vraagtekens plaatsen. Wat betreft B en C onderschrijven we de doelstellingen. Bij D zouden we zoals gezegd de apparatuurzijde willen schrappen. Bij de programmatuurzijde willen we concreter zijn: de leerling dient in staat te zijn een algoritme van maximaal 20 regels te doorzien. Tevens dient hij zo'n algoritme te kunnen wijzigen en uitbreiden, indien dit noodzakelijk is voor de oplossing van een probleem. De leerling dient in staat te zijn te specificeren wat een programma doet of moet doen.

Ter oriëntatie laten we in het kader elders in dit artikel een probleem volgen, dat een eindexamenopgave zou kunnen zijn. Denkt u niet, dat zo'n opgave veel te eenvoudig is voor een eindexamen VWO. Met name op het gebied van nauwkeurigheid en volledigheid zullen veel fouten gemaakt worden. Met opzet is in de opgave niet gezegd in welke taal het programma geschreven moet worden. Men mag best een eigen stoplichten-taal introduceren, als men dan maar goed beschrijft wat de commando's van die taal inhouden. Men is dan meer met informatica bezig dan wanneer er in BASIC of PASCAL geprogrammeerd moet worden. Overigens, hebt u enig idee van het verschil in moeilijkheid van de opgave bij het toevoegen van de eis BASIC, de eis PASCAL of bij het vrijlaten van de te gebruiken taal? Is dat verschil ook zinvol?

Computer-onderwijs en andere vakken

Tenslotte nog iets over de relatie van computeronderwijs tot de andere vakken. Naar onze mening mag computeronderwijs niet een puur op zichzelf staand vak worden met een eenduidige relatie computer – computeronderwijs. Het computeronderwijs zal duidelijk gevolgen hebben voor de andere vakken. Enerzijds omdat het bepaalde vaardigheden uit andere vakken nodig heeft, anderzijds omdat andere vaardigheden aangeleverd zullen worden, die de leerlingen op dit moment niet hebben. Naar onze mening is het zeer goed mogelijk en nuttig om bepaalde onderdelen te combineren met andere vakken.

Zo is bij geen enkel ander vak juiste formulering van wat bedoeld wordt en volledig inzicht in wat men wil bereiken zo noodzakelijk als bij computerkunde. Hierin zal onderwijs gegeven moeten worden en een gedeeltelijk gezamenlijke aanpak met een aantal vakken lijkt ons voor de hand te liggen. Het goed leren formuleren en specificeren kan samen worden opgezet met het vak Nederlands. Het leren ontwikkelen van algoritmen past uitstekend bij wiskunde.

Voorbeelden, toepassingen en oefenstof kunnen prima gevonden worden in de biologie, natuurkunde, scheikunde, economie, maar ook in de talen. Deze vakken zullen overigens goed kunnen profiteren van de invoering van informatica op de scholen. Immers goed formuleren en problemen kunnen oplossen komt overal van pas.

Referenties:

[Herm 83a]
Hermsen, H., *Goede computers voor het onderwijs,* Nieuwe Wiskrant, mei 1983, pag. 17–19.

[Herm 83b]
Hermsen, H., *Programmeren ja of nee, en zo ja in welke taal?* Nieuwe Wiskrant, mei 1983, pag. 25–30.

[Herm 83c]
Hermsen, H., *Een computer kopen, maar welke?* Nieuwe Wiskrant, mei 1983, pag. 57–62.

[Koers 82]
Koers, B.J., *Kennismaking met de computer,* Nieuwe Wiskrant, nov. 1983, pag. 53–59.

[PAO 83]
Programmacommissie PAO-Informatica voor leraren, Structuurplan Nascholing Voortgezet Onderwijs, april 1983.

[Vonk 83]
Vonk, G.A., *Vijftien jaar inleidend informatica-onderwijs,* Nieuwe Wiskrant, mei 1983, pag. 9–12.