Computational thinking (computergericht denken)

- Niet perse opdrachten veranderen maar toevoegen.
  Deze dienen zijn niet extra, maar dienen als een soort vervanging.
  * Soort opdrachten: Een "algemene opdracht". De leerling heeft het hele jaar door de vrijheid/opracht om na te denken over het inzetten van een computer als hulpmiddel.
  Bijv: § 2.2 in [GR 1 h/v] Leerlingen moeten hier voortdurend ggv(x,y), ggd(x,y) uitrekenen. Leerlingen kunnen zelf (of met hulp) tot inzicht komen, hier een code voor te schrijven.

Voorbeeld:	
1	? → a
2	? → b
3	Vind delers a
4	Vind delers b
5	Vergelijk delers a,b
6	Vind ggd(a,b)
7	Print ggd(a,b)

In dit voorbeeld zijn 3/4 zelf kun je 3/4 apart weer uitschrijven (apart programma). Geld ook voor 5 en 6.

toetsbaarheid	Problemen/Nadelen	Voordelen
alg. taal	- Wanneer "werkt" de het programma? - Wanneer is code "handig" geschreven?	- Laagdrempelig voor zowel ll als leraar.
spec. taal	- Leerling moet een taal kennen. - Docent moet zelfde taal als ll kennen - Leerling kan prog. echt maken en, waar niet gewenst gebruiken.	- Je kan controleren of de code werkt. - Leerling kan programma echt maken en waar gewenst gebruiken

Hypothese: De leerling kan zich alleen wenden tot het schrijven van code, wanneer hij de stof begrijpt.

Type I fout: Gemiste kans

Type II fout: Leerdoelen wiskunde komen in het geding.