

1	1	1	1	0	1	1	1	0
0	0	1	0	0	1	0	1	1
0	0	1	1	0	0	0	0	1
1	1	1	0	0	1	0	0	0
0	1	1	0	0	1	0	1	0
0	1	0	0	1	1	0	1	1
1	0	1	1	0	1	0	0	1
0	1	0	0	0	1	1	1	0
1	1	1	0	1	0	1	1	0
1	1	0	0	1	1	0	1	1
0	1	1	1	0	1	0	0	1
1	1	0	0	0	1	0	1	0
0	1	0	0	1	1	0	1	0
0	1	0	1	0	0	0	1	1
1	1	0	0	1	1	0	1	0
0	0	1	1	1	1	1	0	1
1	0	1	0	0	1	0	0	1
0	1	0	1	1	1	1	0	1
0	0	1	0	0	1	0	0	0
0	0	1	0	0	1	1	1	0
0	0	1	1	1	1	1	0	0
0	0	1	1	1	0	1	0	0
1	1	0	0	1	0	1	0	0
0	0	0	1	1	0	0	1	1
1	1	1	0	0	1	0	1	1
0	0	0	0	0	0	1	1	1
0	1	0	1	0	0	0	0	0
1	0	0	1	0	0	0	0	0
0	0	0	0	0	0	0	0	0
1	0	1	1	1	1	0	0	0
1	0	1	0	0	0	0	0	1
1	1	1	0	0	0	0	1	0
0	1	1	0	0	0	1	0	1
1	0	0	0	0	1	1	0	0
1	1	1	1	0	0	1	1	1
0	0	0	1	0	0	1	0	1
1	1	1	1	0	1	1	1	1
0	0	1	0	1	0	1	1	0
0	1	0	1	1	1	0	1	0
0	0	0	1	1	0	0	0	1
1	1	0	0	0	1	0	0	0
0	1	1	0	0	1	1	1	1
0	0	0	1	1	1	0	1	1
1	0	1	0	1	0	1	0	1
0	0	0	0	1	1	0	0	0
0	1	0	1	1	0	0	1	1
1	0	0	1	0	1	0	1	1
0	1	0	1	0	1	1	0	1
1	1	1	1	0	1	0	0	1
1	1	0	0	1	0	1	1	1
1	1	1	1	1	1	1	1	0
1	1	0	1	0	0	0	1	0
0	0	0	1	1	1	0	0	0

Foutje?

Dat verbeteren we toch!

Foutenverbeterende codes



Dit boekje is een bewerking van de masterclass over foutenverbeterende codes met de titel "Foutje? Dat verbeteren we toch!" gegeven door Prof.dr. J.H van Lint.

## **Inhoud**

<b>1</b>	<b>Inleiding</b>	<b>1</b>
<b>2</b>	<b>Een "eenvoudig" coderingssysteem</b>	<b>4</b>
<b>3</b>	<b>De code van de Mariner 6</b>	<b>9</b>
<b>4</b>	<b>De code van CD's</b>	<b>13</b>
	<b>4.1 Modulo rekenen</b>	<b>13</b>
	<b>4.2 Een coderingssysteem modulo 13</b>	<b>15</b>
<b>5</b>	<b>En verder ... ?</b>	<b>20</b>
<b>6</b>	<b>Antwoorden</b>	<b>21</b>



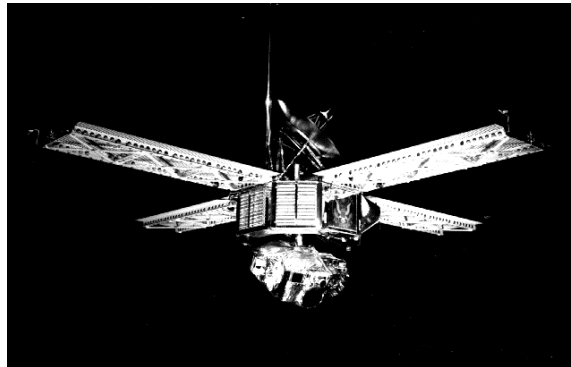
## 1 Inleiding

Informatie wordt vaak vastgelegd via rijtjes **symbolen**. De verzameling symbolen noem je meestal het **alfabet**, de rijtjes worden **woorden** genoemd

Voorbeelden

woorden	alfabet	naam
• Enkele woorden	{ a, b, c, ..... x, y, z }	“ons” alfabet
• $\pi\alpha\nu\tau\alpha$ ρει	{ $\alpha$ , $\beta$ , $\gamma$ .... }	Grieks alfabet:
• 2000	{ 0, 1, ...., 9 }	tientallig stelsel
• 10011	{ 0, 1 }	tweetallig stelsel
• ... - - - ...	{ ., - }	Morse (woord betekent SOS)

In dit artikel worden twee voorbeelden van het vastleggen van informatie door rijtjes 0'en en 1'en besproken. In paragraaf 3 gaat het om foto's van de planeet Mars die door een zender in de satelliet Mariner 6 naar de aarde geseind zijn. Bij die foto's bestond het alfabet uit een 0 en een 1.



De Mariner 6

In paragraaf 4 gaat het om muziek die vastgelegd is op CD. Daar bestaat het alfabet uit 256 “letters” die elk uit acht enen en/of nullen zijn opgebouwd. Een letter noem je dan ook wel een byte. Zo is bijvoorbeeld 01101011 één letter uit dat alfabet. De muziek wordt vastgelegd op de CD op een 5 km lang spiraalvorming spoor van zgn. putten en dammen die bij het uitlezen door de laser omgezet worden in een lange reeks bytes.

In beide voorbeelden worden bij de “ontvanger” van het signaal symbolen door allerlei fouten verkeerd gelezen. Bij het signaal van de satelliet komt dat door ruis die o.a. veroorzaakt wordt door atmosferische storingen. Bij de CD kunnen bijvoorbeeld stofdeeltjes, krassen en vingerafdrukken tot fouten leiden.

In dit artikel gaat het niet om de technische achtergrond bij deze processen. Het gaat om de systemen die zijn bedacht om fouten op te sporen en te verbeteren. Het centrale idee daarachter is niet moeilijk en lijkt op de verbetering van geschreven taal. Als je in een tekst het woord “*gesvhiedenid*” leest zie je direct dat twee maal de verkeerde toets is aangeslagen. Je vindt de twee fouten en verbetert ze: v = c en (de achterste) d = s. Dat je dit kunt komt omdat je maar één woord van twaalf letters kent dat lijkt op wat er stond.

In de **coderingstheorie** kies je eerst een geschikt alfabet. Met dat alfabet maak je dan een “nieuwe” taal waarin alle woorden aan bepaalde voorschriften moeten voldoen. De voorschriften worden zo gekozen dat twee woorden in de taal even veel letters hebben en op veel plaatsen van elkaar verschillen. De taal wordt de **code** genoemd. De woorden in de taal heten **codewoorden**. Als je een woord ontvangt dat geen codewoord is dan zoek je het codewoord dat het meest op een codewoord lijkt.

### Opdracht 1

Een “taal” bestaat uit acht codewoorden  $C_1$  tot en met  $C_8$  die bestaan uit zes symbolen uit het binaire alfabet. De codewoorden zijn:

$C_1$	=	0	0	0	0	0	0
$C_2$	=	1	0	0	1	0	1
$C_3$	=	0	1	0	1	1	0
$C_4$	=	0	0	1	1	1	1
$C_5$	=	1	1	0	0	1	1
$C_6$	=	1	0	1	0	1	0
$C_7$	=	0	1	1	0	0	1
$C_8$	=	1	1	1	1	0	0

$C_1$  verschilt op precies drie plaatsen met  $C_2$ . Je zegt dan dat de **afstand** tussen  $C_1$  en  $C_2$  drie is. Ook de afstand van  $C_1$  tot  $C_3$ , van  $C_1$  tot  $C_6$  en van  $C_1$  tot  $C_7$  is 3. De afstand tussen  $C_1$  en de andere drie codewoorden is 4.  $C_1$  verschilt dus op tenminste drie plaatsen van de andere woorden.

- Bepaal alle andere onderlinge afstanden
- Stel je voor dat je 011110 ontvangt. Je weet zeker dat dat niet verzonden is. Het is immers geen codewoord. Welk woord is er waarschijnlijk verzonden?
- Stel je voor dat er door “ruis” in het codewoord 111100 twee symbolen verkeerd ontvangen worden. Wordt dat woord nog goed gedecodeerd?
- Welk aantal fouten in een ontvangen woord kun je zeker herstellen?

Een andere taal bestaat uit slechts vier codewoorden:

$C_1$	=	0	1	0	1	0	1	0	1
$C_2$	=	1	0	1	0	1	0	1	0
$C_3$	=	1	1	1	1	1	1	1	1
$C_4$	=	0	0	0	0	0	0	0	0

e) Welk aantal fouten in een codewoord kun je nu zeker herstellen?

In opdracht 1 heb je gezien dat je precies één fout in een codewoord kunt herstellen als alle codewoorden op tenminste drie plaatsen van elkaar verschillen. Een woord met twee fouten kun je dan niet met 100% zekerheid herstellen omdat je van het principe uitgaat dat het kleinste aantal fouten het meest waarschijnlijk is. Je weet dan alleen maar dat het ontvangen woord geen codewoord is. De code noem je **1-fout-herstellend**.

Als je ook woorden met twee fouten wilt herstellen dan moet je een code gebruiken waarvoor elk tweetal verschillende codewoorden op tenminste vijf plaatsen verschilt.

### **Opdracht 2**

Verzin een (eenvoudige) 2-fout-herstellende code, d.w.z. verzin een verzameling codewoorden en maak duidelijk dat twee of minder fouten in een codewoord correct hersteld worden.

## 2 Een "eenvoudig" coderingssysteem

In opdracht 1 heb je een voorbeeld van een 1-fout-herstellende code gezien met acht woorden. De keuze van de codewoorden is natuurlijk niet willekeurig. Hoe maak je zo'n code?

Stel dat informatie die gegeven is als een lange rij 0'en en 1'en via een bepaald medium door een zender naar een ontvanger gestuurd moet worden. Het medium heeft de nare eigenschap dat het af en toe een 0 in een 1 of een 1 in een 0 verandert. Als je daaraan niets doet dan zal de ontvanger een rij krijgen waarin op allerlei plaatsen een verkeerd symbool staat.

Toch kun je hiertegen iets doen. Je kunt op een slimme manier extra symbolen aan de rij toevoegen (zogenaamde redundante symbolen). Deze symbolen hebben niks met de boodschap te maken maar ze zorgen ervoor dat de ontvanger eventuele fouten ziet en kan corrigeren. Daarna kan de ontvanger de extra symbolen weggooien.

Een concrete manier waarop dat kan gaat als volgt. Als voorbeeld gaan we uit van de rij 110101101111. Deze rij bestaat uit 12 symbolen uit het alfabet  $\{0, 1\}$ .

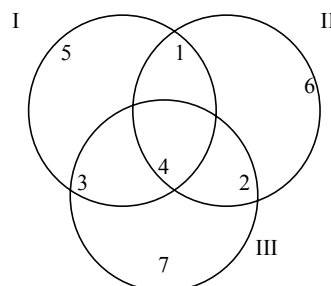
1	2	3	4	5	6	7	8	9	10	11	12
1	1	0	1	0	1	1	0	1	1	1	1

Verdeel de rij in groepjes van vier. Achter elk groepje wordt plaats gemaakt voor drie extra symbolen

1	2	3	4	5	6	7	1	2	3	4	5	6	7	1	2	3	4	5	6	7
1	1	0	1	*	*	*	0	1	1	0	*	*	*	1	1	1	1	*	*	*

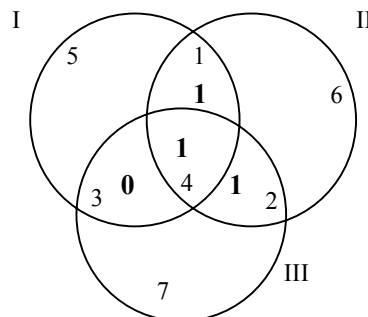
De drie extra symbolen aan het einde worden als volgt vastgelegd

- Teken drie cirkels, I, II en III die elkaar onderling snijden. Dat levert zeven gebieden op:



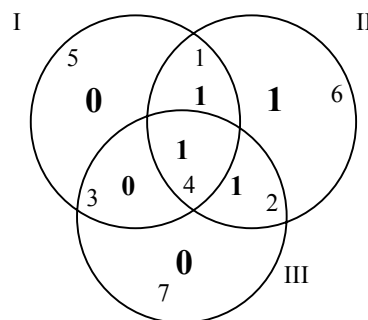


- Schrijf de vier eerste symbolen in de vier binnengebieden. Het eerste symbool in gebied 1, het tweede symbool in gebied 2 etc. Het (eerste) viertal 1101 leidt tot:



- Bepaal de drie achterste symbolen vervolgens met de regel:

**in elke cirkel een even aantal enen**



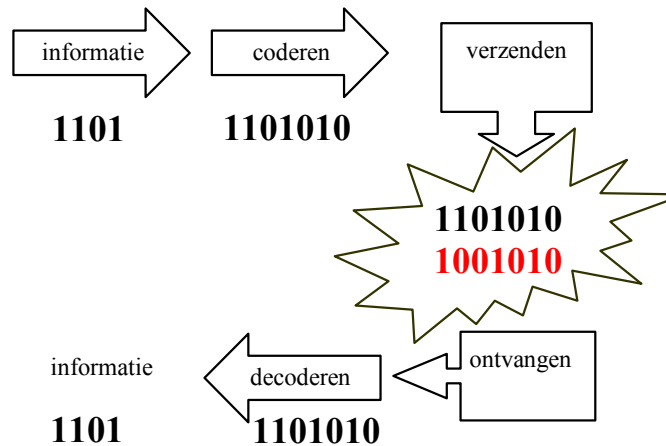
Dat levert in dit geval het woord 1101**0**1**0**.

Op deze manier wordt op eenduidige wijze aan elk groepje drie symbolen toegevoegd en maak je codewoorden die bestaan uit zeven symbolen. Als je de "cirkelregel" op de twee andere groepjes loslaat dan blijkt dat de zender de volgende rij moet versturen:

1	2	3	4	5	6	7	1	2	3	4	5	6	7	1	2	3	4	5	6	7
1	1	0	1	<b>0</b>	<b>1</b>	<b>0</b>	0	1	1	0	<b>1</b>	<b>1</b>	<b>0</b>	1	1	1	1	<b>1</b>	<b>1</b>	<b>1</b>

De ontvanger splitst de rij nullen en enen in groepjes van zeven symbolen en gebruikt de afgesproken regels om te zien of alles klopt. Als iets niet klopt dan herstelt hij de fout als dat mogelijk is en vervolgens laat hij van zo'n zevental de laatste drie symbolen weg.

In de figuur hieronder zie je het hele coderingssysteem schematisch weergegeven.

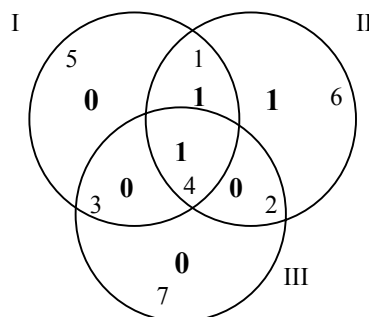


### Opdracht 3

- Ga na dat 0110110 en 1111111 codewoorden in dit systeem zijn.
- Zijn 1010010 en 0001111 codewoorden?
- Hoeveel verschillende codewoorden bezit deze code?
- Hoeveel verschillende rijtjes van 7 symbolen zijn geen codewoord?

Deze code kan geen twee fouten herstellen: 1101010 en 1111111 zijn immers twee codewoorden die onderlinge afstand drie hebben. Als 1101010 ontvangen wordt als 111110 dan zul je dat decoderen met 1111111. De code kan wel één fout herstellen.

Dat gaat gemakkelijk met de cirkels. In het volgende plaatje zie je hoe je dat doet bij het ontvangen woord 1001010. Daarbij ga je uit van het basisprincipe dat het kleinste aantal fouten het meest waarschijnlijk is.



De fout zit niet in cirkel 1, want daarin staan twee enen. In cirkel II staat een oneven aantal enen. Dus daar zit een fout in. Dat geldt ook voor cirkel III. Het symbool in gebied 2 moet dus fout zijn. Het meest waarschijnlijke codewoord is dus 1101010. De verzonden informatie is 1101.

De codewoorden in dit voorbeeld bestaan uit zeven symbolen. Achter de vier informatiesymbolen zet je drie extra symbolen. De code noemt men een  $[7, 4, 3]$ -code. Het eerste getal in deze notatie geeft de lengte van de woorden aan. Het tweede getal de lengte van de informatie-eenheid. Het achterste getal in  $[7, 4, 3]$  geeft de kleinste onderlinge afstand tussen twee verschillende codewoorden aan.

Bij dit systeem verstuur je  $7/4$  keer zo veel symbolen als eigenlijk nodig is. Dat kost  $7/4$  keer zo veel tijd en  $7/4$  keer zoveel energie. De informatiedichtheid is  $4/7$ .

#### Opdracht 4

Ga uit van het in deze paragraaf gemaakte coderingssysteem

a) Decodeer 0000111, 1010001 en 0011010.

Deze code heeft nog een andere aangename eigenschap. Hij kan twee onleesbare symbolen herstellen. Stel je voor dat de symbolen op de eerste en de derde plaats door de een of ander reden niet duidelijk zijn en dat je  $?1?1010$  ontvangt.

b) Zet de bekende symbolen op de juiste plek in de cirkels. Neem aan dat die symbolen correct zijn en bepaal vervolgens de symbolen op de plaats van de vraagtekens.

De besproken  $[7, 4, 3]$ -code verbetert alle woorden waarin één fout gemaakt is. Als er meer fouten in een woord zitten dan wordt er niet juist verbeterd.

Je kunt je afvragen of deze code überhaupt zinvol is.

Stel dat de kans dat een verkeerd symbool ontvangen wordt gelijk is aan 0,05. De kans dat een ontvangen woord dan goed gedecodeerd wordt is gelijk aan de kans dat nul symbolen fout zijn plus de kans dat precies één van de zeven symbolen fout is.

Die kans is dus gelijk aan:  $0,95^7 + 7 \cdot 0,95^6 \cdot 0,05 \approx 0,96$ . Dus gemiddeld worden vier van de 100 groepjes van vier verkeerd geïnterpreteerd.

Als je geen coderingssysteem toepast dan is de kans dat een groepje van vier juist wordt ontvangen gelijk aan  $0,95^4 \approx 0,81$ . Dan worden ongeveer 19 van de 100 groepjes van vier verkeerd geïnterpreteerd.

Dit eenvoudige systeem zorgt er dus al voor dat het foutenpercentage ongeveer vier keer zo klein wordt!

### Opdracht 5

In paragraaf 3 maak je kennis met een  $[16, 5, 8]$  - code. Een codewoord bestaat uit precies 16 symbolen uit alfabet  $\{0, 1\}$

- a) Wat is de informatiedichtheid van deze code?
- b) Hoeveel woorden zijn er in deze code?
- c) Leg uit dat je 3 of minder fouten zeker kunt herstellen?

Stel dat de kans dat een 1 ontvangen wordt als een 0 of een 0 ontvangen wordt door een 1 gelijk is aan 0,05.

- d) Laat met een berekening zien dat slechts 44% van alle verzonden codewoorden foutloos ontvangen wordt.

Gelukkig kun je fouten herstellen.

- e) Laat met een berekening zien dat 99,3% van alle ontvangen woorden goed gedecodeerd wordt.

Als je geen codering gebruikt dan is de kans dat vijf symbolen foutloos ontvangen worden gelijk aan  $(0,95)^5 \approx 0,77$ . De codering heeft dus tot gevolg dat de kans op goed ontvangen van 77% daalt naar 44% *maar na foutenverbetering* is dat 99,3%.

### 3 De code van de Mariner 6

Bij het overseinen van foto's van Mars is geen [7, 4, 3]-code maar een veel "grotere" code gebruikt.



This cratered surface of Mars was photographed by Mariner 6's wide angle TV camera on July 30, 1969 at a range of 3459 kilometers

De code maakt gebruik van **matrices**. Een **matrix** is niks anders dan een blok getallen. Hieronder zie je een matrix, met de naam  $H_1$ , met twee rijen en twee kolommen.

$$H_1 = \begin{pmatrix} +1 & +1 \\ +1 & -1 \end{pmatrix}$$

Deze kleine matrix is de bouwsteen van de matrix die bij het coderingssysteem van de Mariner 6 een rol speelt. Hij heeft de plezierige eigenschap dat het "product" van de twee rijen gelijk aan nul is. Met product wordt hier de som van de producten van de overeenkomstige rijelementen bedoeld:  $(+1) \cdot (+1) + (+1) \cdot (-1) = 0$ . Zo'n product noemt men een **inproduct**. Het inproduct van een rij met zichzelf is bij deze matrix gelijk aan 2. Immers  $(+1) \cdot (+1) + (+1) \cdot (+1) = (+1) \cdot (+1) + (-1) \cdot (-1) = 2$ .

De tegengestelde matrix  $-H_1$ , is een matrix waarin elk getal van  $H_1$  van teken wisselt:

$$-H_1 = \begin{pmatrix} -1 & -1 \\ -1 & +1 \end{pmatrix}$$

Ook het inproduct van de twee verschillende rijen in deze matrix is weer 0.

Met deze twee matrices kun je nu grotere matrices bouwen. Daarbij wordt de volgende verdubbelingsregel gebruikt:

als je een matrix  $H_k$  gemaakt hebt dan is  $H_{k+1}$  gelijk aan

$$H_{k+1} = \begin{pmatrix} H_k & H_k \\ H_k & -H_k \end{pmatrix}$$

Volgens deze regel is de matrix  $H_2$  gelijk aan

$$H_2 = \begin{pmatrix} +1 & +1 & +1 & +1 \\ +1 & -1 & +1 & -1 \\ +1 & +1 & -1 & -1 \\ +1 & -1 & -1 & +1 \end{pmatrix}$$

Als je twee verschillende rijen van  $H_2$  neemt dan is hun inproduct steeds 0. Het inproduct van een rij met zichzelf is dit keer 4.

### Opdracht 6

- Zet onder  $H_2$  de matrix  $-H_2$  en ga na dat in die matrix alle mogelijke combinaties van drie symbolen uit  $\{+1, -1\}$  in de eerste drie kolommen staan
- Ga na of de inproducten van de rijen van matrix  $H_3$  gelijk zijn aan nul en bereken het inproduct van een willekeurige rij met zichzelf.
- Als je onder  $H_3$  de matrix  $-H_3$  zet dan kun je door het schrappen van een aantal kolommen alle mogelijke combinaties van vier symbolen uit  $\{+1, -1\}$  overhouden. Welke kolommen moet je schrappen?

De verdubbelingsregel zorgt ervoor dat ook bij  $H_4$  het inproduct van twee verschillende rijen gelijk is aan 0. De matrix  $H_4$  bestaat uit 16 rijen en 16 kolommen en ziet er als volgt uit:

+1	+1	+1	+1	+1	+1	+1	+1	+1	+1	+1	+1	+1	+1	+1	+1
+1	-1	+1	-1	+1	-1	+1	-1	+1	-1	+1	-1	+1	-1	+1	-1
+1	+1	-1	-1	+1	+1	-1	-1	+1	+1	-1	-1	+1	+1	-1	-1
+1	-1	-1	+1	+1	-1	-1	+1	+1	-1	-1	+1	+1	-1	-1	+1
+1	+1	+1	+1	-1	-1	-1	-1	+1	+1	+1	+1	-1	-1	-1	-1
+1	-1	+1	-1	-1	+1	-1	+1	+1	-1	+1	-1	-1	+1	-1	+1
+1	+1	-1	-1	-1	-1	+1	+1	+1	+1	-1	-1	-1	-1	+1	+1
+1	-1	-1	+1	-1	+1	+1	-1	+1	-1	-1	+1	-1	+1	+1	-1
+1	+1	+1	+1	+1	+1	+1	+1	-1	-1	-1	-1	-1	-1	-1	-1
+1	-1	+1	-1	+1	-1	+1	-1	-1	+1	-1	+1	-1	+1	-1	+1
+1	+1	-1	-1	+1	+1	-1	-1	-1	-1	+1	+1	-1	-1	+1	+1
+1	-1	-1	+1	+1	-1	-1	+1	-1	+1	+1	-1	-1	+1	+1	-1
+1	+1	+1	+1	-1	-1	-1	-1	-1	-1	-1	-1	+1	+1	+1	+1
+1	-1	+1	-1	-1	+1	-1	+1	-1	+1	-1	+1	+1	-1	+1	-1
+1	+1	-1	-1	-1	-1	+1	+1	-1	-1	+1	+1	+1	+1	-1	-1
+1	-1	-1	+1	-1	+1	+1	-1	-1	+1	+1	-1	+1	-1	-1	+1

**Opdracht 7**

- a) Hoe groot is het inproduct van een rij uit  $H_4$  met zichzelf?
- b) Op hoeveel plaatsen verschillen twee rijen minimaal met elkaar?

In de kolommen met rangnummer 1, 2, 3, 5 en 9 bevinden zich alle mogelijke combinaties van vijf symbolen die beginnen met +1. Als je onder  $H_4$  de matrix  $-H_4$  zet dan krijg je een matrix met 32 rijen en 16 kolommen. In die matrix staan in de kolommen met rangnummer 1, 2, 3, 5 en 9 **alle** mogelijke combinaties van vijf symbolen. Je hebt dan dus 32 verschillende informatiedragers.

Hoe verstuur je nu met deze matrix foto's?

De foto wordt bijvoorbeeld verdeeld in 1000 keer 1000 kleine vierkantjes (pixels). Van elke vierkantje wordt de zwarting gemeten met een of ander optisch instrument op een schaal van 0 tot en met 31. De zwarting wordt uitgedrukt in het tweetallig stelsel. Zwarting 13 komt dan overeen met 01101. Voor één foto moeten dus 1 miljoen van die vijftallen naar de aarde gestuurd worden.

De codering gaat als volgt.

- maak van een 0 een 1 en van een 1 een -1  
De rij 01101 voor zwarting 13 wordt dan (+1, -1, -1, +1, -1)
- Zoek in de matrix  $H_4$  met daaronder  $-H_4$  de rij op die op plaatsen 1, 2, 3, 5 en 9 met deze vijf symbolen overeenstemt.  
In dit voorbeeld is dat de twaalfde rij:

+1 -1 -1 +1 +1 -1 -1 +1 -1 +1 +1 -1 -1 +1 +1 -1

- Verstuur dit codewoord naar de aarde.

De gemaakte code is een [16, 5, 8]-code met fraaie eigenschappen. Je kunt namelijk zeer eenvoudig decoderen en herstellen.

Stel dat er op de aarde ontvangen wordt:

+1 +1 -1 +1 -1 +1 +1 -1 -1 +1 +1 -1 +1 -1 -1 +1

Als dit woord een codewoord is dan is het inproduct van dit woord met alle andere codewoorden gelijk aan 0 en het inproduct met zichzelf is 16. Het inproduct van dit woord met de eerste rij in  $H_4$  is gelijk aan  $1 + 1 - 1 + 1 - 1 + 1 + 1 - 1 - 1 + 1 + 1 - 1 - 1 + 1 = 2$ . Alle inproducten van dit woord met de 16 codewoorden in de matrix  $H_4$  zijn achtereenvolgens 2, -2, 2, -2, 2, -2, 2, -2, 2, -2, 2, -2, 2, -2, 2, 14.

Het woord is dus geen codewoord. Dat weet je al na de eerste berekening. Bovendien wordt duidelijk dat het codewoord in de zestiende rij het meest op het woord lijkt. Omdat die rij op één plaats van het ontvangen woord verschilt is het inproduct 14. De meest waarschijnlijke rij is dus

**+1 -1 -1 +1 -1 +1 +1 -1 -1 +1 +1 -1 +1 -1 -1 +1**

De corresponderende zwarting is 01111.

Omdat de rijen in  $H_4$  op precies acht plaatsen van elkaar verschillen kun je drie gemaakte fouten zeker herstellen. De code is 3-fout-herstellend. Met behulp van de inproducten kun je weer "snel" decoderen. Bekijk als voorbeeld het woord.

**-1 -1 +1 +1 +1 +1 +1 +1 -1 -1 -1 -1 -1 -1 +1 -1**

De zestien inproducten zijn dit keer -2, 2, -6, -2, -6, -2, -2, 2, 10, -2, -2, 2, -2, 2, -6, -2. De negende rij in  $H_4$  is dit keer het meest waarschijnlijk. Daar zitten drie fouten in, namelijk op de eerste, tweede en vijftiende positie. De zwarting is in deze situatie 00001

Vier fouten kun je niet met 100% zekerheid herstellen. Dat wordt gokwerk.

### Opdracht 8

- Hoe kun je aan de inproducten zien dat er meer dan drie fouten gemaakt zijn?
- Decodereer met behulp van inproducten.

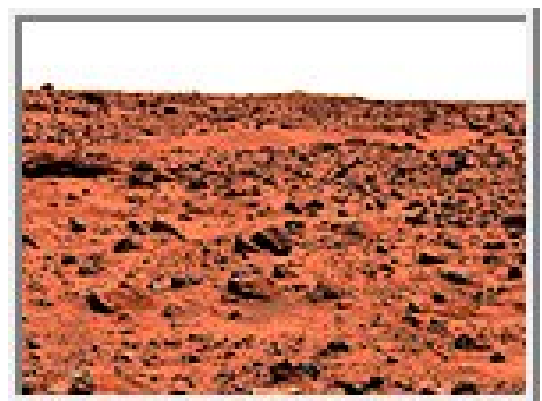
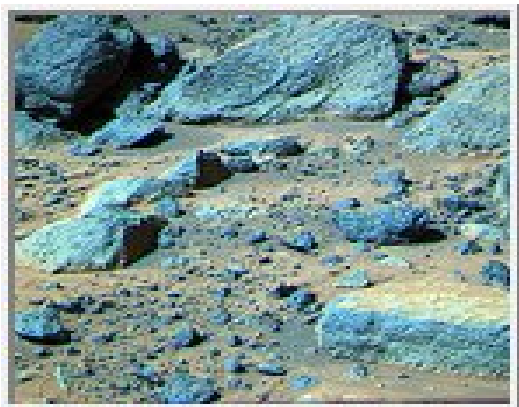
+1 -1 -1 -1 -1 -1 -1 +1 +1 -1 +1 -1 -1 +1 -1 +1  
-1 -1 +1 -1 -1 +1 -1 +1 -1 +1 -1 +1 -1 +1 -1 +1

In werkelijkheid werd er bij de Mariner 6 een nog grotere code gebruikt. De code die gebruik maakt van de matrix  $H_5$

### Opdracht 9

De op  $H_5$  gebaseerde code is een  $[a, b, c]$  - code. Bepaal "zonder veel rekenwerk" a, b en c.

Hieronder zie je foto's van het oppervlak van Mars. Die foto's zijn gemaakt met een opvolger van de Mariner 6.





## 4 De code van CD's

In de code die bij een CD-speler gebruikt wordt, wordt gebruikt gemaakt van bytes. Een byte is opgebouwd uit 8 symbolen uit het tweetallig stelsel {0, 1}. Een voorbeeld van een byte is 10110001. Er bestaan dus 256 verschillende bytes.



Met bytes kun je ook rekenen. Maar met bytes reken je niet gewoon: 10101011 keer 11100011 is niet gelijk aan 112121333211121. Want dat "getal" is geen byte meer. Je moet dus duidelijk afspreken wat de rekenregels zijn.

Omdat het rekenen met bytes bewerkelijk is, wordt in deze paragraaf een ander alfabet gekozen. De essentie van het coderingssysteem is hetzelfde als bij de CD.

### 4.1 Modulo rekenen

Als het 18 uur is dan is het over 48 uur weer 18 uur. Na 24 uur begin je immers opnieuw te tellen. Over 10 uur is het 4 uur (in de nacht):  $18 + 10$  is namelijk 28 uur en daar haal je dan 24 uur vanaf.

Eigenlijk "kijk" je steeds hoe vaak 24 uur in de som "past" en schrijf je de rest op. Dat kun je ook zo opschrijven:  $18 + 10 = 28 = 1 \cdot 24 + 4 = 4$

Deze manier van rekenen heet *modulo* rekenen. In dit voorbeeld reken je modulo 24. In feite kijk je dan alleen naar de resten die je krijgt bij deling door 24. Om het onderscheid met "gewoon" rekenen aan te geven wordt bij een gelijkheid het teken  $\equiv$  gebruikt in plaats van  $=$ . Het modulusgetal schrijf je achteraan.

$$18 + 10 \equiv 28 \equiv 4 \text{ mod } 24$$

Bij modulo 24 rekenen werk je meestal alleen met de getallen 0, 1, 2, ..., 23. Dat zijn de resten die je kunt krijgen bij deling door 24. Bij vermenigvuldigen doe je precies hetzelfde. Je rekent steeds met de resten:

$$5 \cdot 23 \equiv 115 \equiv 19 \pmod{24}$$

$$3 \cdot 8 \equiv 24 \equiv 0 \pmod{24}$$

Je kunt ook met negatieve getallen rekenen. Rest 23 kun je zien als rest -1. Het rekenen gaat dan soms wat sneller:

$$5 \cdot 23 \equiv 5 \cdot (-1) \equiv (-5) + 24 \equiv 19 \pmod{24}$$

Ook delen is mogelijk. Als je vraagt naar  $11/6 \pmod{24}$  dan zoek je een getal dat vermenigvuldigd met 6 het getal 11 geeft. Dat zie je niet zo maar. Maar omdat er slechts eindig veel resten modulo 24 zijn kun je dat met een vermenigvuldigingstabel snel uitzoeken:

$x$	0	1	2	3	4	5	6	7	8	9	10	11
$(6 \cdot x) \pmod{24}$	0	6	12	18	0	6	12	18	0	6	12	18
$x$	12	13	14	15	16	17	18	19	20	21	22	23
$(6 \cdot x) \pmod{24}$	0	6	12	18	0	6	12	18	0	6	12	18

In de tabel staat geen 11. Voor geen enkele waarde van  $x$  levert  $6 \cdot x$  het getal 11. Dus  $11/6$  bestaat modulo 24 niet. Je kunt wel 12 door 6 delen. Maar ook hier gebeurt er iets vreemds. Er zijn zes oplossingen:

$$6 \cdot 2 \equiv 6 \cdot 6 \equiv 6 \cdot 10 \equiv 6 \cdot 14 \equiv 6 \cdot 18 \equiv 6 \cdot 22 \equiv 12 \pmod{24}$$

Delen modulo 24 is geen bewerking met "mooie eigenschappen". Dat komt door het getal 24. Een getal waarbij delen geen problemen oplevert is bijvoorbeeld 13. In de onderstaande tabel zie je de tafel van 6 modulo 13.

$x$	0	1	2	3	4	5	6	7	8	9	10	11	12
$(6 \cdot x) \pmod{13}$	0	6	12	5	11	4	10	3	9	2	8	1	7

Elke rest komt precies één keer voor. Dat betekent dat je "elk getal door 6 kunt delen":

$$3/6 \equiv 7 \pmod{13} \quad \text{want} \quad 6 \cdot 7 \equiv 42 \equiv 3 \pmod{13}.$$

Ook bij andere getallen heeft de deling modulo 13 precies één uitkomst. Dat kun je als volgt inzien. Neem een willekeurig getal  $x$  ongelijk aan 0 en twee willekeurige getallen  $a$  en  $b$ . Omdat je modulo 13 rekent zijn  $x$ ,  $a$  en  $b$  geheel en kleiner dan 13.

Bekijk de producten  $x \cdot a$  en  $x \cdot b$  modulo 13. Als die twee producten gelijk zijn dan geldt:  $x \cdot a \equiv x \cdot b \pmod{13}$ . Dan is het verschil tussen  $x \cdot a$  en  $x \cdot b$  een veelvoud van 13. Dus  $x \cdot a - x \cdot b = x \cdot (a - b)$  is een veelvoud van 13.

Dan volgt echter dat  $x$  of  $(a - b)$  deelbaar is door 13. Dat kan niet  $x$  zijn want  $x$  ligt tussen 1 en 12. Verder weet je dat  $a - b$  tussen 0 en 12 ligt als je modulo 13 rekent. Dat houdt in dat de enige mogelijkheid die overblijft  $(a - b) = 0$  is. Maar dat betekent dat  $a \equiv b \pmod{13}$  en dus dat  $x \cdot a \equiv x \cdot b \pmod{13}$ . Een uitzondering moet je nog maken. Je kunt niet delen door 0. Maar dat kan ook niet als je "gewoon" rekent.

### Opdracht 10

Het bovenstaande bewijs is niet goed voor het modulusgetal 24. Dan kunnen twee producten wel gelijk zijn als  $a \neq b$ .

- Welke conclusie in het bewijs is niet waar voor 24? Waarom klopt dat wel bij 13?
- Bereken  $8 / 7$  en  $12 / 11$  modulo 13 en modulo 24

## 4.2 Een coderingssysteem modulo 13

Nu kun je een code maken met als alfabet de getallen modulo 13. Stel je voor dat de informatie bestaat uit een "lange" rij getallen tussen 0 en 12. Voor alle duidelijkheid wordt voor 1-cijferige getallen een 0 geplaatst. De informatie is bijvoorbeeld:

12 , 05 , 09 , 02 , 07 , 01 , 05 , 11 , 10 , 03 ...

Deze rij wordt opgesplitst in vijftallen waaraan steeds twee extra symbolen worden toegevoegd.

C0	C1	C2	C3	C4	C5	C6
12	05	09	02	07	*	*

C0	C1	C2	C3	C4	C5	C6
01	05	11	10	03	*	*

Op deze manier maak je dus een code met woordlengte 7. De twee extra symbolen worden niet zomaar gekozen. Ze moeten voldoen aan:

$$\begin{cases} c_0 + c_1 + c_2 + c_3 + c_4 + c_5 + c_6 & \equiv 0 \pmod{13} \\ c_1 + 2 \cdot c_2 + 3 \cdot c_3 + 4 \cdot c_4 + 5 \cdot c_5 + 6 \cdot c_6 & \equiv 0 \pmod{13} \end{cases}$$

Met deze twee vergelijkingen kun je steeds de onbekenden  $c_5$  en  $c_6$  uitrekenen. Bij het eerste vijftal krijg je:

$$\begin{cases} 12 + 5 + 9 + 2 + 7 + c_5 + c_6 & \equiv 9 + c_5 + c_6 & \equiv 0 \pmod{13} \\ 5 + 2 \cdot 9 + 3 \cdot 2 + 4 \cdot 7 + 5 \cdot c_5 + 6 \cdot c_6 & \equiv 5 + 5 \cdot c_5 + 6 \cdot c_6 & \equiv 0 \pmod{13} \end{cases}$$

Uit de eerste vergelijking volgt  $c_5 \equiv -c_6 - 9 \pmod{13}$ . Dat vul je in in de tweede vergelijking. Je krijgt dan:

$$5 + 5 \cdot (-c_6 - 9) + 6 \cdot c_6 = c_6 - 40 \equiv c_6 - 1 \equiv 0 \pmod{13}$$

Deze vergelijking heeft als oplossing  $c_6 = 1$ .

Invullen in  $c_5 \equiv -c_6 - 9 \pmod{13}$  levert  $c_5 \equiv -10 \equiv 3 \pmod{13}$ . Het eerste codewoord wordt dus (12 , 05 , 09 , 02 , 07 , **03** , **01**).

### Opdracht 11

- a) Is (01 , 02 , 03 , 04 , 05 , 02 , 09] codewoord?
- b) Bepaal de twee extra symbolen achter het vijftal (01 , 05 , 11 , 10 , 03).

### Opdracht 12

$C_1 = (12 , 05 , 09 , 02 , 07 , 03 , 01)$  is een van de vele codewoorden.

- a) Hoeveel verschillende codewoorden en verschillende woorden zijn er in dit coderingssysteem?
- b) Hoeveel woorden hebben afstand een tot  $C_1$ ? En hoeveel hebben afstand twee?

Bij het decoderen moet er gecontroleerd worden of de zeven symbolen voldoen aan de twee vergelijkingen voor codewoorden. Als beide antwoorden 0 zijn dan is er waarschijnlijk geen fout gemaakt.

Als je bijvoorbeeld (12 , 05 , 09 , 02 , 07 , 03, 01) ontvangt dan trek je die conclusie. Een ander voorbeeld.

Stel je ontvangt: (12 , 05 , 09 , 02 , 04 , 03 , 01). Invullen in de vergelijkingen levert:

$$\begin{cases} 12 + 5 + 9 + 2 + 4 + 3 + 1 & \equiv 10 \pmod{13} \\ 5 + 2 \cdot 9 + 3 \cdot 2 + 4 \cdot 4 + 5 \cdot 3 + 6 \cdot 1 & \equiv 1 \pmod{13} \end{cases}$$

Het woord is dus geen codewoord. Het meest waarschijnlijk is precies één fout. Hoe kun je die fout nu opsporen?

Dat kan op de volgende manier.

Stel dat het verzonden woord  $(c_0 , c_1 , c_2 , c_3 , c_4 , c_5 , c_6)$  is. Het ontvangen woord is  $(12 , 05 , 09 , 02 , 04 , 03 , 01) = (r_0 , r_1 , r_2 , r_3 , r_4 , r_5 , r_6)$ .

Stel nu dat **alleen** op de  $i^e$  positie een fout zit ( $i = 0$  of  $i = 1$  of ... of  $i = 6$ ). De fout is een getal  $e$  tussen 0 en 12. Je rekent immers modulo 13.

Er geldt dus voor een zekere  $i$  dat  $r_i = c_i + e$ . Als je nu  $i$  en  $e$  kunt berekenen dan ben je klaar. Je weet dan de positie van de fout en de fout zelf.

De fout  $e$  is eenvoudig te berekenen met vergelijking 1. Je moet gewoon alle getallen van het ontvangen woord optellen.

Je krijgt dan:

$$\begin{aligned} & r_0 + r_1 + r_2 + r_3 + r_4 + r_5 + r_6 \\ & \equiv c_0 + c_1 + c_2 + c_3 + c_4 + c_5 + c_6 + e \\ & \equiv 0 + e \\ & \equiv e \pmod{13} \end{aligned}$$

Dus in dit voorbeeld krijg je:

$$e \equiv r_0 + r_1 + r_2 + r_3 + r_4 + r_5 + r_6 = 12 + 5 + 9 + 2 + 4 + 3 + 1 \equiv 10 \pmod{13}$$

De fout  $e$  is dus gelijk aan 10. De plaats van de fout kun je bepalen met vergelijking 2.

Door de slimme keuze van de getallen voor de symbolen komt de fout  $e$ ,  $i$  keer in de som tevoorschijn:

$$\begin{aligned} & r_1 + 2 \cdot r_2 + 3 \cdot r_3 + 4 \cdot r_4 + 5 \cdot r_5 + 6 \cdot r_6 \\ & \equiv c_1 + 2 \cdot c_2 + 3 \cdot c_3 + 4 \cdot c_4 + 5 \cdot c_5 + 6 \cdot c_6 + i \cdot e \\ & \equiv 0 + i \cdot e \\ & \equiv i \cdot e \pmod{13} \end{aligned}$$

In dit voorbeeld:

$$\begin{aligned} i \cdot e & \equiv r_1 + 2 \cdot r_2 + 3 \cdot r_3 + 4 \cdot r_4 + 5 \cdot r_5 + 6 \cdot r_6 \\ & \equiv 5 + 2 \cdot 9 + 3 \cdot 2 + 4 \cdot 4 + 5 \cdot 3 + 6 \cdot 1 \\ & \equiv 1 \pmod{13} \end{aligned}$$

En nu wordt ook duidelijk waarom je een modulusgetal moet kiezen waarbij delen geen probleem is. In dit voorbeeld is  $i$  gelijk aan :

$$1/10 \equiv 40/10 \equiv 4 \pmod{13}.$$

Er is dus op de vijfde positie (je begint bij  $i = 0$  te tellen) een fout van  $e = 10$  gemaakt. Dat betekent dat  $c_4 \equiv r_4 - 10 \equiv 4 - 10 \equiv -6 \equiv 7 \pmod{13}$ .

Het meest waarschijnlijke codewoord is dus : (12 , 05 , 09 , 02 , **07** , 03 , 01)

In het algemeen bereken je de (meest waarschijnlijke) fout  $e$  en de foutenplaats  $i$  met het stelsel:

$$\begin{cases} r_0 + r_1 + r_2 + r_3 + r_4 + r_5 + r_6 & \equiv e \pmod{13} \\ r_1 + 2 \cdot r_2 + 3 \cdot r_3 + 4 \cdot r_4 + 5 \cdot r_5 + 6 \cdot r_6 & \equiv i \cdot e \pmod{13} \end{cases}$$

Als dit stelsel geen oplossingen of een oplossing met  $i > 6$  heeft dan zijn er twee of meer fouten gemaakt. Die kun je niet meer herstellen. Je hebt maar twee vergelijkingen en er zijn vier onbekenden; de twee posities van de fouten en de twee fouten op die posities.

### Opdracht 13

In het ontvangen woord (01 , 03 , 04 , 06 , 08 , 05 , 01) zit meer dan een fout.

- Laat dat met moduloberekeningen zien.
- Zoek een codewoord met afstand 2 tot dit ontvangen woord?  
Tip: stel dat de fouten  $e$  en  $f$  op de plaatsen  $i$  en  $j$  zitten
- Hoeveel verschillende codewoorden zijn er met afstand 2 tot dit ontvangen woord?

Deze code kan dus maar één fout herstellen. Het is een  $[7, 5, 3]$ -code die bekend staat onder de naam **Reed-Solomon** code. De informatiedichtheid is  $5/7$ . De code is dus efficiënter dan de  $[7, 4, 3]$ -cirkelcode in paragraaf 2 die ook 1-fout-herstellend is.

### Opdracht 14

Decodeer de volgende woorden als dat mogelijk is.

- (11 , 02 , 03 , 04 , 05 , 02 , 09)
- (08 , 02 , 10 , 03 , 05 , 00 , 11)
- (05 , 01 , 03 , 01 , 01 , 08 , 03)

Een  $[8, 5, 4]$ - Reed-Solomon code ontstaat als je drie symbolen aan een vijftal toevoegt. De vergelijkingen die dan gebruikt worden zijn.

$$\begin{cases} c_0 + c_1 + c_2 + c_3 + c_4 + c_5 + c_6 + c_7 & \equiv 0 \pmod{13} \\ c_1 + 2 \cdot c_2 + 3 \cdot c_3 + 4 \cdot c_4 + 5 \cdot c_5 + 6 \cdot c_6 + 7 \cdot c_7 & \equiv 0 \pmod{13} \\ c_1 + 2^2 \cdot c_2 + 3^2 \cdot c_3 + 4^2 \cdot c_4 + 5^2 \cdot c_5 + 6^2 \cdot c_6 + 7^2 \cdot c_7 & \equiv 0 \pmod{13} \end{cases}$$

Als je de kwadraten in de onderste vergelijking modulo 13 uitrekent dan krijg je  $c_1 + 4 \cdot c_2 + 9 \cdot c_3 + 3 \cdot c_4 + 12 \cdot c_5 + 10 \cdot c_6 + 10 \cdot c_7 \equiv 0 \pmod{13}$

**Opdracht 15**

- a) Welke symbolen moeten er in dit coderingssysteem bij het verzenden achter (06 , 05 , 01 , 01 , 02) gezet worden?
- b) Stel dat er precies één fout in een codewoord gemaakt is:  $r_i = c_i + e$ . Met welke drie vergelijkingen kun je  $e$  en  $i$  bepalen?
- c) Decodeer (01 , 02 , 00 , 04 , 05 , 10 , 06 , 08) en (05 , 03 , 03 , 02 , 00 , 11 , 10 , 03) als dat mogelijk is.

Bij een CD-speler wordt een coderingssysteem gebruikt dat op soortgelijke principes werkt. Alleen is de situatie veel gecompliceerder. Er wordt o.a. gebruik gemaakt van een [28 , 24 , 5] Reed-Solomon code gevolgd door een [32 , 28 , 5] Reed-Solomon code. Elk van die codes kan twee fouten verbeteren, maar door samenwerking kunnen ze meestal veel meer.

## 5 En verder ... ?

In de vorige paragrafen heb je kennis gemaakt met de wiskundige basisprincipes van een aantal coderingssystemen. Naar aanleiding daarvan kun je nog veel vragen stellen. Bijvoorbeeld:

- Ook een Nederlandse tekst kun je met een Reed-Solomon code coderen. Je kunt dan het modulusgetal 31 kiezen en een paar leestekens toevoegen

<b>spatie</b>	<b>a</b>	<b>b</b>	<b>c</b>		<b>'</b>	<b>,</b>	<b>!</b>	<b>?</b>
0	1	2	3		27	28	29	30

Hoeveel fouten kun je herstellen als je aan een boodschap van 4 symbolen twee symbolen toevoegt (informatiedichtheid  $2/3$ ). Is de Reed-Solomon code die aan 8 symbolen vier symbolen toevoegt beter?

- Wat doe je met een woord waarin je de fouten niet kunt herstellen? Je moet toch iets doen!
- Bestaan er methoden waarmee je een stelsel modulo-vergelijkingen "snel" kunt oplossen?
- Hoe werken de coderingssystemen op de CD-speler samen?
- Bestaan er nog meer "slimme" coderingssystemen?
- ....

En dan zijn er nog veel "technische" vragen

- Hoe zet je analoge signalen om in rijtjes enen en nullen en andersom?
- Hoe automatiseer je het decoderingsalgoritme?
- Hoe leest een CD-speler de enen en nullen?
- Hoe wordt een CD-schijfje gemaakt?
- Hoe verstuur je met een satelliet signalen naar de aarde?
- Wat was de missie van Mariner 6?
- ....

Kortom. Dit onderwerp biedt vele uitbreidingsmogelijkheden. Iets voor een profielwerkstuk?



## 6 Antwoorden

### Opdracht 1

a)

	$C_2$	$C_3$	$C_4$	$C_5$	$C_6$	$C_7$	$C_8$
$C_2$		4	3	3	4	4	3
$C_3$			3	3	4	4	3
$C_4$				4	3	3	4
$C_5$					3	3	4
$C_6$						4	3
$C_7$							3

b)  $C_3 = 010110$ c) Nee, stel dat bijvoorbeeld de symbolen op plaats 2 en 3 verkeerd worden ontvangen. Je ontvangt dan 100100. Het codewoord met de kleinste afstand is nu  $C_2 = 100101$ .

d) 1 fout

e) 1 fout

### Opdracht 3

b) 1010010 is geen codewoord, 0001111 wel

c) 16

d) 112

### Opdracht 4

a) 0001 ; 1010 ; 0011

b) woord is 1101010

### Opdracht 5

a)  $\frac{5}{16}$ 

b) 32

### Opdracht 6

b) inproduct rij met zichzelf = 8

c) Een mogelijkheid is de kolommen 4, 6, 7 en 8 schrappen. Er zijn ook andere mogelijkheden.

### Opdracht 7

a) 16

b) 8

### Opdracht 8

b) 01010 ; 10111

### Opdracht 9

[32, 6, 16]

**Opdracht 10**

$$\text{b) } \frac{8}{7} \equiv 3 \pmod{13} \quad \frac{8}{7} \equiv 8 \pmod{24} \quad \frac{12}{11} \equiv 7 \pmod{13} \quad \frac{12}{11} \equiv 12 \pmod{24}$$

**Opdracht 11**

- a) ja
- b) Het codewoord is dan (01, 05, 11, 10, 03, 06, 03)

**Opdracht 12**

- a) 37293 codewoorden en 62748517 woorden
- b) 84 op afstand 1; 3024 op afstand 2

**Opdracht 13**

- a)  $i = 7 > 6$
- b) Je krijgt het stelsel:

$$\begin{cases} e + f \equiv 2 \pmod{13} \\ i \cdot e + j \cdot f \equiv 1 \pmod{13} \end{cases}$$

Kies bijvoorbeeld  $i = 0$  en  $j = 1$ . Dan volgt  $f = 1$  en  $e = 1$ . Een codewoord op afstand twee is dus (00, 02, 04, 06, 08, 05, 01)

- c) 21

**Opdracht 14**

- a) (01, 02, 03, 04, 05, 02, 09)
- b) woord is codewoord
- c)  $i = 8$  dus niet te decoderen

**Opdracht 15**

- a) codewoord wordt (06, 05, 01, 01, 02, 11, 08, 05)
- b) woord 1: (01, 02, 03, 04, 05, 10, 06, 08)  
woord 2: niet te decoderen