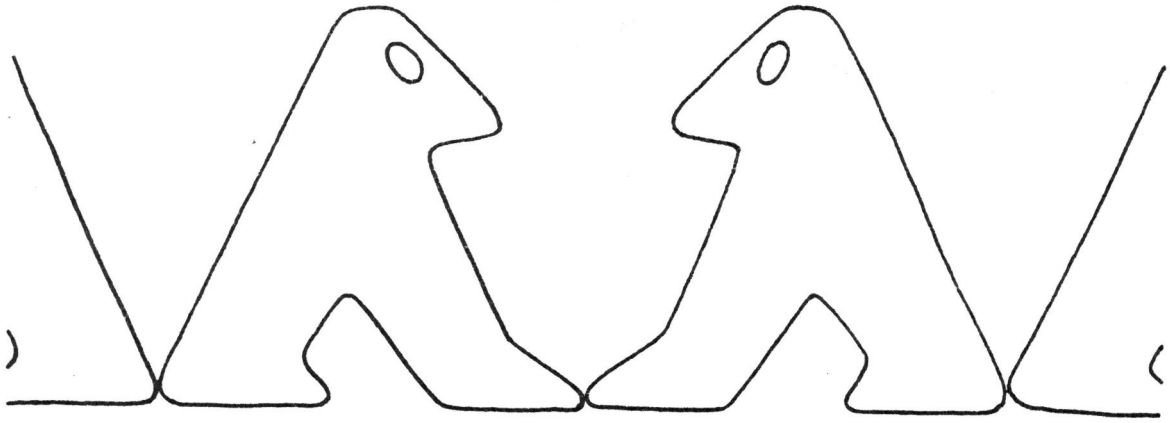
Cursus Informatica

Aart Dokter
Revius Lyceum
Leo Geurts
Centrum voor
Wiskunde en Informatica



1. Inleiding

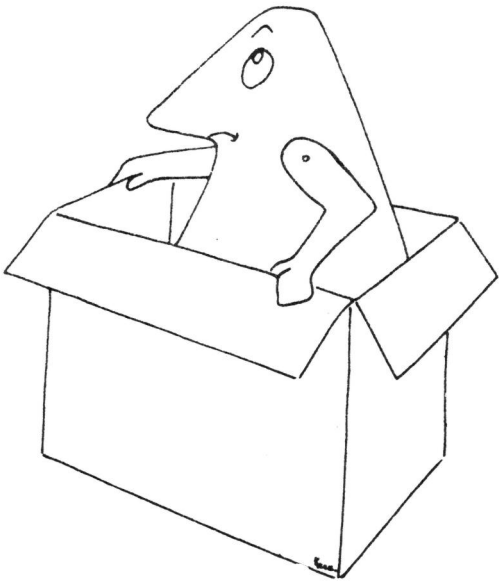
Toen je voor de eerste keer wiskunde kreeg, werd er niet verteld wat dat nu precies was. Je zou het immers toch niet precies begrepen hebben als men je had verteld dat er *met onbekenden werd gerekend.* En bovendien wist je ongeveer wel in welke richting je het vak wiskunde moest plaatsen.

Nu begin je weer met een nieuw vak: informatica. Net als aan het begin van het vak wiskunde zou je het nu misschien ook niet begrijpen als er een nauwkeurige omschrijving zou worden gegeven van wat informatica inhoudt. Toch vraag je je misschien wel af, wat informatica met een computer heeft te maken, want daarmee gaan we nu immers aan het werk.

Informatica heeft natuurlijk te maken met informatie. Informatie ontvangen we op veel manieren. We praten met elkaar, weontvangen brieven, we krijgen cijfers op ons rapport, we kijken naar de televisie, enzovoort. Gesproken en geschreven woorden, getallen en lichtsignalen zijn allemaal middelen waarmee informatie wordt overgedragen. Een computer is nu juist een apparaat dat heel goed op verschillende manieren met informatie kan omgaan. Dat maakt de computer uniek. Andere apparaten kunnen maar één ding: probeer met een wasmachine maar eens koffie te zetten. Een computer daarentegen kan heel veel verschillende dingen. Deze dingen worden bepaald door de verschillende *programma's.* Deze cursus informatica is erop gericht om verschillende eigenschappen van de computer te leren gebruiken; verder zullen we ook proberen om de computer nieuwe eigenschappen te geven: we gaan hem *programmeren.* Je kunt het een beetje vergelijken met een breimachine. Eerst leren we, hoe we hem laten breien. Vervolgens kijken we naar het breipatroon (programma), waarin nauwkeurig staat aangegeven wat de machine moet breien. Door nu dit patroon wat te wijzigen kunnen we de machine steeds iets anders laten breien.

2. Ons computersysteem

2.1. Inleiding

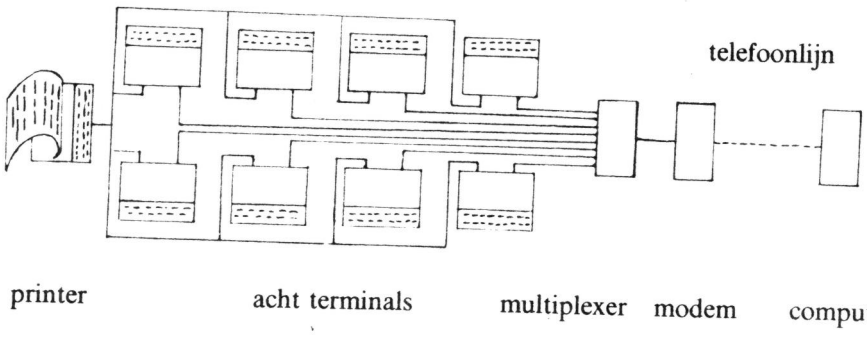


Een computer zit in een doosje. Soms is dat doosje zo klein dat je het bijna niet kunt zien. Denk maar aan een polshorloge waar bovendien nog een rekenmachine inzit. Vroeger nam een computer veel meer ruimte in beslag. Voor de eerste computers rond 1945 was een hele gymzaal nodig om ze te plaatsen, terwijl je nu met je zakrekenmachine nog sneller berekeningen kunt maken dan met die eerste computers. Trouwens, het grootste gedeelte van die zakrekenmachine bestaat uit het toetsenbordje en een schermpje. Binnenin zit de computer: een *chip* van een paar vierkante centimeter bevat de *microprocessor*. Je kunt een microcomputer (personal computer) dan ook heel goed vergelijken met zo'n zakrekenmachine, alleen het scherm en het toetsenbord zijn wat forser uitgevallen en de microprocessor kan ook wat meer. Bovendien bezitten de wat grotere computers de mogelijkheid om gegevens in hun *geheugen* op te bergen.

Er bestaan ook computers waaraan meer beeldschermen en toetsenborden gekoppeld zijn. Men kan dan gelijktijdig op verschillende plaatsen met dezelfde computer werken. Zo'n werkplaats (beeldscherm en toetsenbord) noemt men dan een *terminal*. Als je dus iemand voor een toetsenbord bij een beeldscherm ziet zitten, dan kan het zijn dat hij met zijn eigen (micro)computer aan het werk is, maar het kan ook zijn dat hij verbinding heeft met een krachtige computer, die op een andere plaats in het gebouw, of zelfs buiten het gebouw staat opgesteld. Met zo'n grotere computer gaan wij nu werken.

2.2. Apparatuur

In lokaal 32 staan 8 grote tafels, die elk aan vier leerlingen plaats bieden. Op iedere tafel staat een terminal, dus een beeldscherm en een toetsenbord. Deze 8 terminals zijn verbonden met een grote computer in Utrecht. In onderstaande tekening staat schematisch aangegeven hoe de onderlinge samenhang van de verschillende apparaten is.



Deze terminals zijn allemaal verbonden met een doos: een *multiplexer*. Dit apparaat perst de 8

signalen samen tot één signaal. Dit signaal wordt vervolgens door het *modem* geschikt gemaakt om via de telefoonlijn naar Utrecht te worden verstuurd, waar een andere multiplexer het weer in de 8 signalen splitst en aan de computer doorgeeft. De signalen die de computer naar de acht terminals stuurt komen via dezelfde weg terug. Zo is dus ieder beeldscherm via een ingewikkelde weg verbonden met de computer. Verder is iedere terminal verbonden met de printer, die je vanaf elke plaats zijn werk kunt laten doen. Zodra er contact is met de computer verschijnt er op het scherm:

```
Onderwijs Computercentrum
naam:
```

Dit is de uitnodiging om met de computer aan het werk te gaan.

2.3. Toetsenbord en beeldscherm

Het zou veel gemakkelijker zijn als een computer oren had, waarmee hij kon horen wat wij van hem willen. Zover is het nog niet. Voorlopig verloopt de communicatie met de computer nog via toetsenbord en beeldscherm. Wij "zeggen" iets tegen de computer door het in te toetsen. De computer neemt datgene wat wij intoetsen niet slechts in zich op, hij zet wat hij heeft "gehoord" bovendien op het scherm. Hierdoor kan het soms gebeuren dat het even duurt voordat een ingetoetste letter op het scherm verschijnt: de computer hoort het wel, maar heeft nog geen tijd gehad om het te laten merken. Zodra de computer de opdracht gekregen en op het scherm gezet heeft, gaat hij die opdracht uitvoeren. Het is mogelijk dat dat een tijdje duurt. In zo'n geval is het beter even geduld te hebben dan je ongeduld op het toetsenbord af te reageren.

Gelukkig lijkt het toetsenbord veel op dat van een schrijfmachine, want alle witte toetsen komen daar ook op voor. Als je een toets met een letter erop indrukt, dan verschijnt de *kleine* letter. Houd je echter tijdens het intoetsen van een letter de [SHIFT]-toets ingedrukt, dan wordt de *hoofd*letter zichtbaar.

Wanneer je veel hoofdletters achter elkaar wilt hebben, kun je ook [CAPSLOCK] een keer intoetsen. Er verschijnt dan **CAPS** links boven aan het scherm. Zonder dat je de [SHIFT]-toets gebruikt komen nu alle letters als hoofdletters te voorschijn. Druk je nog een keer op [CAPSLOCK] dan zullen de daarna ingetoetste letters weer als kleine letters verschijnen, en verdwijnt de aanduiding **CAPS** boven aan het scherm weer.

Op sommige toetsen staan twee symbolen. Als je zo'n toets aanslaat verschijnt het onderste symbool. Wil je het bovenste symbool hebben, dan gebruik je weer de [SHIFT]-toets. Zo zal [SHIFT]-**8** dus * op het scherm zetten. De spatie is een symbool dat je niet kunt zien. Je gebruikt hiervoor de balk onder aan het toetsenbord. Het intoetsen van [TAB] geeft een aantal spaties tegelijk. Verder zie je dat er twee mogelijkheden zijn om getallen in te toetsen, waarbij het geen verschil maakt of je nu de cijfers op het linker of het rechter gedeelte van het toetsenbord gebruikt.

Wanneer je op een nieuwe regel wilt beginnen, moet je de [RETURN]-toets gebruiken ([ENTER] heeft dezelfde werking). Maar de [RETURN]-toets heeft voor de computer nog een speciale betekenis. Je geeft met de [RETURN] namelijk aan, dat jouw opdracht aan de computer beëindigd is. Daarom is dit een uiterst belangrijke toets. Als je het werken met de computer wilt beëindigen, moet je de **d**-toets aanslaan, terwijl je de [CTRL]-toets ingedrukt houdt. We geven dit aan met [CTRL]-**d** (de letters *ctrl* komen van *control*). Rechts bovenaan op het scherm staan getalletjes die aangeven waar het volgende symbool op het scherm wordt geplaatst. Zo betekent **0-13-017** dat het volgende symbool op de zeventiende positie van de dertiende regel komt.

Soms geeft de computer heel veel informatie op het scherm. Dat gaat dan zo snel, dat je niet in staat bent om het te lezen. In zo'n geval kun je het scherm ook stilzetten met de [NO SCROLL]-toets. Als je hierna de [NO SCROLL]-toets weer aanslaat, rolt de rest van de informatie over het scherm.

Het gebruik van de [BACKSPACE]-toets doet de vorige aanslag teniet. Gebruik je twee keer [BACKSPACE], dan zullen de laatste twee aanslagen worden uitgewist. Je hoeft dus niet al te bang te zijn om een verkeerde toets aan te slaan.

2.4. Je eigen werkgebied

In de inleiding is gezegd, dat een computer verschillende dingen kan doen. Zo kan hij net doen alsof hij verdeeld is in heel veel afzonderlijke computers, die allemaal dezelfde eigenschappen hebben als de grote. Hierdoor is het mogelijk dat het lijkt alsof iedere leerling *zijn eigen computer* heeft. Het maakt daarbij geen verschil achter welke terminal je plaats neemt: ze zijn immers allemaal met de grote computer verbonden en dus ook met *die van jou*. Om nu uit te leggen hoe jij met je *deelcomputer* aan het werk kunt, gebruiken we de volgende beeldspraak.

Vergelijk onze computer met een kantoorflat. Het Reviuslyceum huurt, naast andere scholen, een etage van dit flatgebouw. We hebben onze etage zo ingedeeld, dat iedere klas zijn eigen afdeling heeft. Op een afdeling heeft elke leerling de beschikking over 2 kamers: één kamer voor hem alleen en één kamer waarin hij met 2 à 3 leerlingen samenwerkt. Zo'n kamer is dus je eigen werkruimte met het bordje op de deur *Geen toegang voor onbevoegden*. Nu gaan we na hoe jij, in tegenstelling tot de onbevoegden, wel toegang krijgt tot je werkruimte. We hebben in paragraaf 2.2 gezien dat het verbinding maken met de computer tot gevolg heeft dat er verschijnt:

```
Onderwijs Computercentrum
naam:
```

Je staat nu voor het flatgebouw en je moet aangeven naar welke etage je wilt:

```
Onderwijs Computercentrum
naam: revius
(Klas)naam:
```

Je toetst dus de naam `revius` in. Let erop dat je de informatie die je hiermee aan de computer geeft, afsluit met een druk op de [RETURN]-toets.

Na de begroeting kun je zeggen naar welke afdeling je wilt op de `revius`-etage. Peter uit h35 zal dus intoetsen:

```
(Klas)naam: h35
(leerling)naam:
```

De kamer van Peter heet `peter` (en die van Yvonne: `yvonne`), zodat hij nu moet intoetsen:

```
(leerling)naam: peter
Wachtwoord:
```

Als dit voor Peter al voldoende zou zijn om op zijn kamer te komen, dan zou Yvonne daar ook gemakkelijk kunnen komen: ze hoefde dan ook alleen maar

`revius` [RETURN] `h35` [RETURN] `peter` [RETURN]

in te toetsen. Maar er zit een slot op Peters deur: het *wachtwoord-slot*. We zullen later zien hoe Peter dat zelf kan maken als hij in zijn werkruimte is. Tot zolang staat zijn kamerdeur nog wijd open. Door het intoetsen van [RETURN] gaat hij in zijn kamer aan zijn buro zitten om zo aan de slag te kunnen. Er staat dus nu op Peters scherm:

```
Onderwijs Computercentrum
naam: revius
(Klas)naam: h35
(Leerling)raam: peter
Wachtwoord:
λ
```

Aan het λ-teken kun je zien dat je aan het werk kunt. Zo hebben dus alle leerlingen een eigen werkruimte. Bovendien is de klas verdeeld in 8 groepen en heeft iedere groep een eigen werkgebied. Als Peter, Yvonne, Jan en Mirjam in groep 5 zitten, dan krijgt ieder van hen op de volgende wijze toegang tot hun gemeenschappelijke werkgebied.

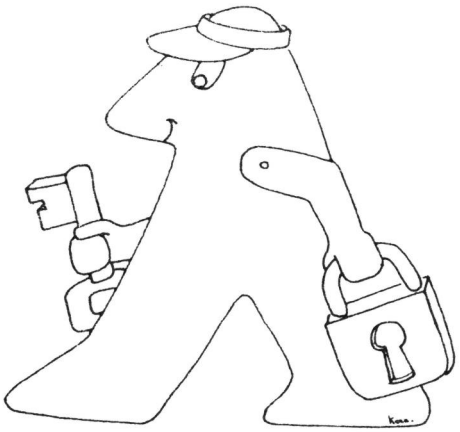
2.6. Het wachtwoord

Allereerst gaat Peter een slot op zijn deur maken. Zittend achter zijn buro roept hij de softwerker met de naam λ `wachtwoord`. Eigenlijk hoort het λ-teken er niet bij, maar dat wordt er toch bijgezet om te onderstrepen dat de naam van een softwerker alleen maar na het (automatisch verschijnende) λ-teken gebruikt mag worden. Peter toetst nu in:

```
λ wachtwoord peter
```

Het softwerkertje vraagt vervolgens om nadere instruktie:

```
oude wachtwoord:
```



Peter heeft nog geen wachtwoord (slot) dus toetst hij alleen [RETURN] in. Vervolgens komt de vraag:

```
nieuwe wachtwoord:
```

Peter heeft het woord `snor7` in z'n gedachte en toetst dat in. Gelukkig verschijnt het niet op het scherm, zodat zijn medeleerlingen niet stiekem zijn wachtwoord kunnen lezen en het daarna gebruiken om in Peters werkruimte te komen. Als Peter nu iets anders intoetst, zonder dat hij dat merkt, dan heeft hij een wachtwoord (slot) gemaakt, dat hij ook zelf niet kent, met alle gevolgen vandien. Daarom verschijnt er:

```
nogmaals nieuwe wachtwoord:
```

Nadat Peter nu weer `snor7` heeft ingetoetst, zal het slot zijn gemaakt. Er staat dan op het scherm:

```
λ wachtwoord peter
oude wachtwoord:
nieuwe wachtwoord:
nogmaals nieuwe wachtwoord:
λ
```

Peter hoeft natuurlijk niet iedere keer een nieuw wachtwoord te maken. Hij kan nu in het vervolg zo in zijn werkruimte komen:

```
Onderwijs Computercentrum
naam: revius
(Klas)naam: h35
(Leerling)naam: peter
Wachtwoord:
λ
```

Maar Peter besluit nu om maar te stoppen met het werken achter de terminal. Als hij opstapt zonder verdere maatregelen te nemen, dan kan iemand anders zo in zijn gebied verder werken. Daarom verlaat Peter zijn gebied met:

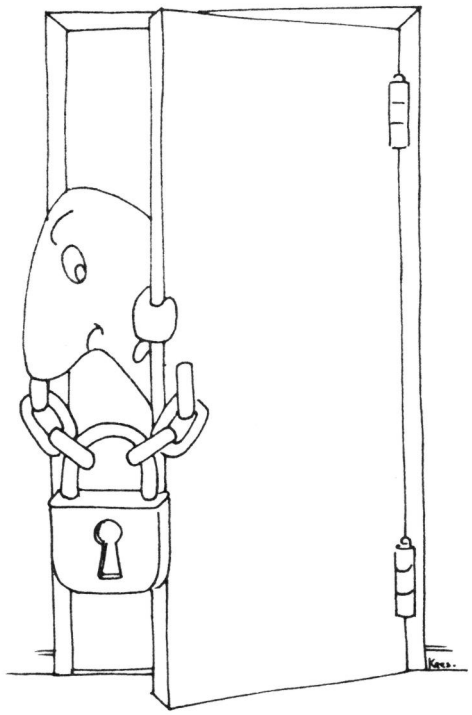
λ [CTRL]`d`

([CTRL] vasthouden en tegelijkertijd `d` intoetsen). Dit levert op:

```
(Klas)naam:
```

Inderdaad heeft Peter nu zijn kamer verlaten, maar hij staat nog wel op de `revius`-etage. Iedere leerling kan nu weer op zijn eigen kamer komen, door de naam van zijn klas, zijn eigen naam en tenslotte zijn wachtwoord in te toetsen. Wordt er na `(Klas)naam:` nogmaals [CTRL]`d` ingetoetst, dan verschijnt er weer:

```
Onderwijs Computercentrum
naam:
```



☞ 2.1. Ga naar je eigen werkgebied. Bedenk een wachtwoord en bevestig dat als slot op je deur. Verlaat vervolgens je gebied en controleer of je nu weer binnen kunt komen. Hierna verlaat je je kamer opnieuw, zodat een andere leerling uit je groepje het bovenstaande ook kan uitvoeren.

☞ 2.2. De deur van de gemeenschappelijke kamer, waar je als groepje werkt, moet ook nog van een slot worden voorzien. Spreek samen stilletjes af welk wachtwoord je kiest en installeer het daarna.

2.7. Communicatie via de terminals

Omdat alle terminals met dezelfde computer verbonden zijn, is het mogelijk om een bericht op de ene terminal in te toetsen om het op het beeldscherm van een andere terminal te laten verschijnen. Hiervoor gebruiken we de softwerker λ `post`. Peter wil aan Heidi vragen of zij vanavond met hem huiswerk wil maken. Hij doet dat als volgt:

```
λ post heidi
wil jij vanavond huiswerk
met mij maken?
.
```

(De boodschap moet eindigen met een punt vooraan de regel.) Op Heidi's scherm verschijnt nu:

```
Er is post
```

Als Heidi dan intoetst:

```
λ post
```

komt de boodschap op haar scherm, met de vraag of ze dit bericht wil bewaren. Zij kan nu `j` of `n` intoetsen (*ja* of *nee*). Als ze `j` antwoordt wordt de boodschap ergens opgeborgen; de eerstvolgende keer dat ze λ `post` zegt, komt hij weer tevoorschijn.

Het is ook mogelijk om post naar een andere klas te sturen. Als Heidi in klas h36 zou zitten, dan zou Peter moeten intoetsen:

```
λ post h36 heidi
```

Als de terminals van Heidi en Peter op wat grotere afstand van elkaar staan (b.v. in verschillende gebouwen), dan kan Peter even kijken of Heidi verbinding heeft met de computer. De softwerker

```
λ wie
```

zet de namen op het scherm van iedereen die nu verbinding heeft met de computer.

Je kunt verder met:

```
λ wie ben ik
```

je geheugen wat opfrissen.

Sommige berichten zijn voor alle terminalgebruikers van belang. Zo kan je leraar ervoor zorgen dat er op ieder scherm b.v. de datum verschijnt van het volgende proefwerk. Als je nog wat meer algemene informatie wilt hebben, dan kun je intoetsen:

```
λ nieuws
```

Heeft het computercentrum iets te melden, dan zal dat nu op je scherm verschijnen. De datum en de tijd kun je laten verschijnen met:

```
λ datum
```

☞ 2.3. Je wilt natuurlijk het bovenstaande uitproberen. Dat is prima, maar verdoe er niet te veel tijd mee: er staan nog zoveel softwerkers te popelen om voor jou hun opdracht uit te voeren.

2.8. Gegevens opbergen: file

Een computer is een belangrijk apparaat, omdat hij zoveel verschillende dingen heel snel kan uitvoeren. Verder wordt de computer ook vaak gebruikt omdat hij heel veel informatie kan opslaan. We zeggen dan dat hij die informatie in zijn *geheugen* bewaart; denk b.v. aan het bewaren van de post in de vorige paragraaf. Aan de hand van de gekozen beeldspraak stellen we ons voor hoe het opslaan van gegevens in zijn werk gaat.

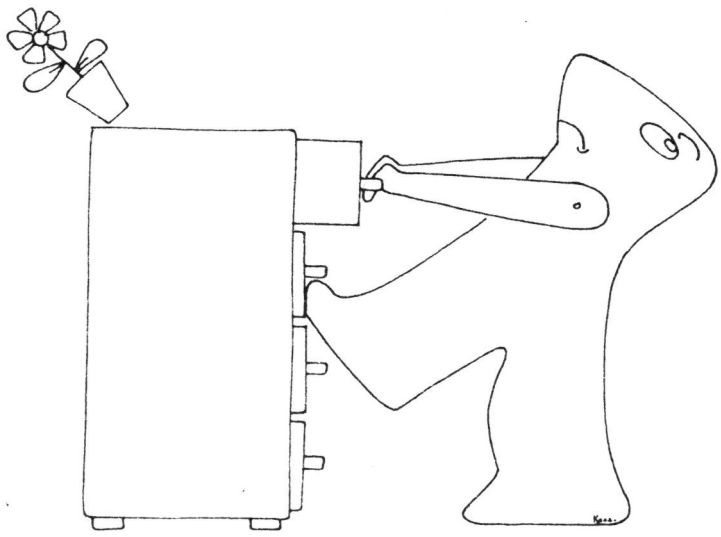
De wanden van je werkruimte zijn bedekt met ladenkasten. Op elke lade is een naamkaartje geplakt. Verder ligt in iedere lade een aantal velletjes papier. Als sommige velletjes beschreven zijn dan is het naamkaartje ingevuld. In computertermen wordt de lade een *file* genoemd; op het kaartje staat de naam van de file, terwijl de inhoud van de file op de blaadjes staat geschreven. Je zit nu op je kamer achter het buro en je kunt op afroep softwerkers hun speciale opdracht geven. Als voorbeeld nemen we λ `wie` uit de vorige paragraaf. Als je hem roept met:

```
λ wie
```

zal hij het resultaat van zijn werk, de namen van hen die op dit moment verbinding hebben, op het scherm zetten. In plaats van dit resultaat op het scherm te laten zetten, kun je het ook in een file laten opbergen. Dat gaat als volgt:

```
λ wie > lijst
```

De softwerker λ `wie` trekt dan een lege lade open, schrijft wat hij op het scherm wilde zetten nu op de lege velletjes, sluit de la en vult tenslotte het woord `lijst` op het naamkaartje in. Het resultaat van het werk van λ `wie` staat nu dus in de file `lijst`. Zo kun je met `>` en daarna de naam van een file, van veel softwerkers de resultaten van hun werk in een file bewaren (door `>` wordt een richting gesuggereerd). Wanneer je b.v.



```
λ datum > tijd
```

intoetst, wordt de file `tijd` gemaakt. De inhoud van deze file bestaat slechts uit de aanduiding van de tijd waarop hij gemaakt is.

Je kunt ook het resultaat van het werk van een softwerker in een bestaande file *bijschrijven*:

```
λ datum >> lijst
```

voegt de tijdsaanduiding toe achter de reeds bestaande inhoud van de file `lijst`.

Als je wat meer files hebt gemaakt, wordt het steeds moeilijker om alle namen te onthouden en bovendien weet je ook niet meer precies wat er in een file zit. Ook hier kan een softwerker ons helpen. We roepen hem erbij met:

```
λ toon
```

Nu zullen de namen van al je files op het scherm verschijnen:

```
lijst tijd
```

Eveneens kan λ `toon` ons helpen als we de inhoud van een file willen zien:

```
λ toon tijd
```

Er komt nu op het scherm:

```
don 21 feb 1985 - 09:05:48
```

Je zult nu begrijpen dat

```
λ toon lijst > lijst1
```

niets op het scherm zet, maar een copie van `lijst` maakt en die in `lijst1` zet.

Als je wilt weten hoe groot een file is, kun je intoetsen:

```
λ regels tijd
```

Er verschijnt nu:

```
1 tijd
```

hetgeen wil zeggen, dat de file `tijd` uit 1 regel bestaat.

Je kunt ook het aantal woorden bepalen:

```
λ woorden tijd
```

Dit levert op:

```
6 tijd
```

☞ 2.4. Maak de files `lijst`, `lijst1` en `tijd` zoals hierboven is beschreven.

☞ 2.5. Maak een file `overzicht` waarin de namen van de bestaande files zijn opgenomen. Controleer of je het goed hebt gedaan.

☞ 2.6. Maak een file `totaal` waarin de gezamenlijke inhoud van de files `lijst` en `tijd` is opgenomen. Controleer daarna of dat inderdaad is gebeurd.

☞ 2.7. Kies een file en voeg een regel toe, waarop vermeld wordt uit hoeveel regels de file bestond.

2.9. Gemeenschappelijk gebied

Iedereen kan in zijn eigen werkgebied met files manipuleren, maar je kunt niet aan de files van je buurman komen. Toch zou het wel handig zijn als je files onderling zou kunnen uitwisselen. Dit blijkt inderdaad mogelijk te zijn. Op de `revius`-etage is hiervoor speciale ruimte gereserveerd: het *gemeenschappelijke gebied*. Er zijn maar twee softwerkers, die in dit gebied kunnen werken: λ `haal` en λ `breng`. Hoe werken zij?

Peter heeft de file **boom2** in zijn eigen gebied staan. Hij toetst nu in:

```
λ breng boom2
```

wat tot resultaat heeft, dat de file **boom2** nu zowel op Peters gebied als op het gemeenschappelijk gebied staat. Peter stuurt Kees nu een berichtje, waarin hij vertelt wat hij zojuist heeft gedaan. Vervolgens toets Kees in:

```
λ haal boom2
```

Nu heeft Kees **boom2** ook in zijn gebied staan. Zo kun je met λ **haal** een file die op het gemeenschappelijke gebied staat ook in je eigen gebied zetten, terwijl λ **breng** een file in omgekeerde richting verplaatst. (In werkelijkheid worden de files zelf niet verplaatst maar worden er kopieën gebracht en gehaald.)

☞ 2.8. Op het gemeenschappelijke gebied staat de file **mt**. Zet hem op je eigen terrein en bekijk de inhoud.

☞ 2.9. Stuur een file naar een andere groep van je klas.

2.10. Nog meer handelingen met files

Als je de inhoud van een file op papier wilt afdrukken (dit is geen beeldspraak), dan kun je intoetsen:

```
λ print pietje
```

De inhoud van de file **pietje** wordt nu door de printer op computerpapier gezet. Iedere avond komt er een softwerker, de schoonmaker, op je kamer. Als een lade de afgelopen drie weken niet open is geweest, gooit dit softwerkertje de inhoud in de prullenbak en verwijdert tevens het naamkaartje van de file. Dit kan soms uiterst vervelend zijn. Maar je kunt er wel wat tegen doen. Je kunt de lade nl. zo afsluiten, dat de schoonmaker niets kan weggooien. Je toetst daarvoor één keer in:

```
λ beveilig pietje
```

De file **pietje** kan dan niet meer door de schoonmaker worden weggegooid.

Overigens ben je zelf ook in staat om een file te verwijderen:

```
λ verwijder boom
```

vernietigt de file **boom**.

Soms is het prettig om een file een andere naam te geven. Je kunt dit doen met:

```
λ herdoop boom2 struik
```

De file **boom2** heeft nu de naam **struik** gekregen.

☞ 2.10. Kies als groepje een file en laat hem door de printer afdrukken.

☞ 2.11. Ga na of je een file kunt verwijderen die je eerst hebt beveiligd.

☞ 2.12. Geef de file **mt** een nieuwe naam. Kies een korte naam, die goed bij de inhoud past. Kun je ook op aan andere manier precies hetzelfde bereiken als met λ **herdoop**?

☞ 2.13.

a. Vind je het een veilig idee als er echt belangrijke gegevens van jou in de computer (een file) staan opgeborgen? Of geef je toch de voorkeur aan een echte kluis?

b. Wat vind je van het communiceren via de terminal?

2.11. Overzicht

λ `beveilig filenaam`	beveiligt tegen dagelijkse schoonmaak
λ `breng filenaam`	zet een copie op het gemeenschappelijk gebied
λ `datum`	zet de tijdsaanduiding op het scherm
λ `haal filenaam`	copiëert een file van gemeenschappelijk naar eigen gebied
λ `herdoop filenaam1 filenaam2`	geeft een file een andere naam
λ `nieuws`	geeft het eventuele nieuws van het computercentrum
λ `post`	laat het binnengekomen bericht zien
λ `post leerlingnaam`	verzendt post aan andere leerling
λ `print filenaam`	zet een file op computerpapier
λ `regels filenaam`	telt het aantal regels van een file
λ `toon`	zet alle filenamen op het scherm
λ `toon filenaam`	laat de inhoud van een file zien
λ `verwijder filenaam`	gooit een file weg
λ `wachtwoord leerlingnaam`	beveiligt je werkruimte
λ `wie`	geeft de namen van hen die nu verbinding hebben
λ `woorden filenaam`	telt het aantal woorden van een file
`> filenaam`	zet produktie van softwerker in file `filenaam`
`>> filenaam`	zet produktie aan eind van file `filenaam`
file	geordend stukje geheugen
geheugen	plaats waar informatie wordt opgeslagen
software	verzameling instrukties (programma) voor de computer
softwerker	personificatie van een stukje software
[BACKSPACE]	wist de vorige toetsaanslag uit
[BREAK]	breekt de activiteit van een softwerkertje af
[CAPSLOCK]	wisselt van kleine naar hoofdletters (en andersom)
[CTRL]-**d**	verbreekt de verbinding
[NO SCROLL]	zet het scherm stil (en weer in beweging)
[RETURN]	beëindigt opdracht aan softwerker, geeft nieuwe regel
[SHIFT]	maakt een hoofdletter
[TAB]	geeft een aantal spaties

3. Tekstverwerking

3.1. Inleiding

Als je het woord *computer* hoort, denk je misschien eerder aan een machine waarmee je iets kunt berekenen, dan aan een apparaat waarmee je taalkundige dingen kunt doen. *To compute* betekent tenslotte *rekenen*. Dat een computer getallen in zijn geheugen kan opslaan, en van twee getallen kan zien welke het grootste is, dat is vanzelfsprekend. Toch is de stap van rekenen naar taal voor een computer minder groot dan je zou verwachten.

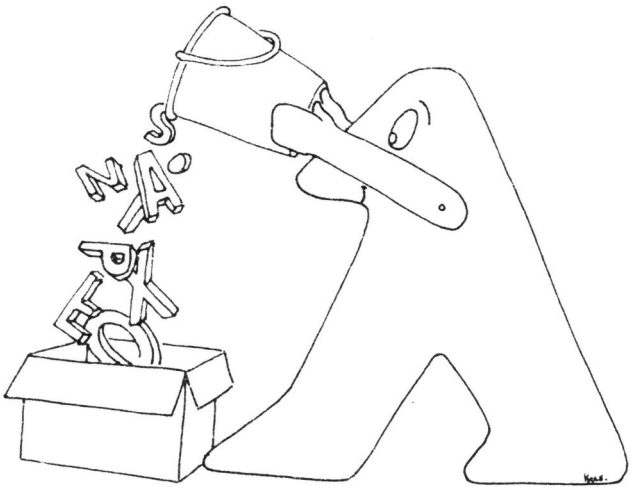
Onze taal is opgebouwd uit zinnen en woorden, die worden gevormd door de 26 letters van ons alfabet en nog wat symbolen zoals de punt, de komma, de spatie, enzovoort. We kunnen de letters a t/m z laten corresponderen met resp. de getallen 1 t/m 26, en de andere symbolen met 27, 28, enzovoort. Je begrijpt dat een computer dan ook letters, woorden en zinnen in zijn geheugen kan vastleggen. Hij moet er dan natuurlijk wel even bij onthouden dat dan b.v. 16 niet het *getal* 16, maar de *letter* p aangeeft.

☞ 3.1. Probeer uit te leggen hoe een computer dan in staat is om een rij woorden op alfabetische volgorde te zetten.

Het vervolg van dit hoofdstuk is bedoeld om kennis te maken met de computer als machine die met teksten opereert: de *tekstverwerker* of *editor*.

3.2. Teksten in het computergeheugen plaatsen

Je zit nu aan een toetsenbord bij een beeldscherm met de bedoeling om zelfgemaakte tekst in te toetsen. Om uit te leggen hoe dat gaat gebruiken we dezelfde beeldspraak als in het vorige hoofdstuk. Allereerst ga je naar je eigen werkruimte door je naam en je wachtwoord in te toetsen:



```
Onderwijs Computercentrum
Naam: revius
(Klas)naam: h35
(leerling)naam: peter
Wachtwoord:
λ
```

Aan het λ-teken zie je nu dat de softwerkertjes weer klaar staan om voor je aan het werk te gaan. We hebben nu het softwerkertje nodig dat zich speciaal op het werken met teksten heeft toegelegd. Zijn naam luidt:

```
λ maak
```

Wanneer je een tekst wilt gaan maken, moet je van te voren aan λ `maak` vertellen hoe je de file wilt noemen waarin de tekst bewaard moet worden. Als die file b.v. `briefje` moet heten, dan toets je in:

```
λ maak briefje
Hallo,
Dit is mijn eerste
kennismaking met de
tekstverwerking.
Peter.
```

Om aan te geven dat λ **maak** nu moet ophouden met zijn werk, toets je [F1] in terwijl je de [SHIFT]-toets ingedrukt houdt (we geven dit weer aan met [SHIFT]-[F1]). Nu is de ingetoetste tekst in de file **briefje** in het geheugen opgeborgen. Verder verschijnt er weer:

λ

Als je tijdens het intoetsen van de tekst een toetsaanslag ongedaan wilt maken, kun je daarvoor de toets [DEL] gebruiken. Als je de [DEL] tweemaal intoetst, worden de laatste twee aanslagen ongedaan gemaakt etc. (als je niet met λ **maak** werkt gebruik je voor het ongedaan maken van een toetsaanslag de [BACKSPACE], zoals in het vorige hoofdstuk is aangegeven).

☞ 3.2. Je gaat nu als journalist een kort verslag maken van het gebeuren dat in de nevenstaande plaatjes staat afgebeeld. Daarna berg je het verslag op in een file, waarvoor je zelf een naam verzint.

3.3. Tekstdelen in de focus plaatsen

Mogelijk vind je het verslag dat je in de vorige paragraaf hebt gemaakt, niet goed genoeg om te publiceren. Er moeten bij nader inzien wel enige veranderingen worden aangebracht. In paragraaf 3.4 komt aan de orde hoe een bestaande tekst gewijzigd kan worden. Maar voordat je een wijziging wilt aanbrengen moet je natuurlijk precies de plaats in de tekst aanwijzen waar de verandering moet komen. Stel dat je in de file **golf** de volgende tekst hebt geplaatst:

```
Mickey, een teleurgestelde kampioen
(van onze sportverslaggever).

Op de plaatselijke golfkampioenschappen heeft
Mickey Mouse na een spannende finale de eerste
plaats voor zich opgeeist.

Het in groten getale aanwezige publiek bleek bij
de uitreiking van de beker zeer ingenomen met
de nieuwe kampioen. De persfotografen zagen in
het gedrang nauwelijks kans om de hoogtepunten
vast te leggen.

Direkt na afloop van de prijsuitreiking snelde de
bekerwinnaar naar huis om zijn kostbare bezit te
tonen. Zijn teleurstelling was echter groot toen
zijn kampioensbewijs tot bloemenvaas werd gedegradeerd.
```

☞ 3.3. Bedenk een manier, waarop je de computer kunt "vertellen" dat je b.v. het woord **kampioen** wilt wijzigen.

☞ 3.4. Op het gemeenschappelijk werkgebied staat de file **golf**. Copieer deze file naar je eigen werkruimte.

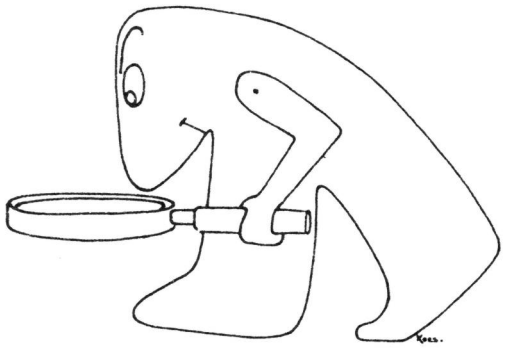
☞ 3.5. Voer wat hieronder besproken wordt direkt uit op je toetsenbord. De voorbeelden zijn zo gekozen dat je ze in de aangegeven volgorde moet uitvoeren.

Het softwerkertje λ **maak** mag nu weer aan de slag om te laten zien dat hij meer kan dan alleen een tekst in een file opnemen:

```
λ maak golf
```

In opdracht 3.3 heb je een manier bedacht om de computer (dus eigenlijk λ **maak**) te vertellen, dat je in bepaalde gedeelten van de tekst nader geïnteresseerd bent. We gaan nu na hoe dat bij onze computer geregeld is.

Het gaat hierbij allemaal om de *focus*. Dat is het oplichtende deel op het scherm, dat nu op het laatste woord van de file **golf** staat. Maar we kunnen de focus ook op een andere plaats in de tekst zetten, we kunnen hem groter en kleiner maken en zo is het mogelijk om alle plaatsen in de file met een focus te bereiken. En als de focus op een bepaalde plaats in de tekst staat, dan weet λ **maak** waar de verandering moet komen. Nu dus de vraag:



Hoe verplaatsen we de focus?

Het antwoord op deze vraag wordt in de onderstaande gedeelten a t/m j gegeven.

a) De focus groter maken: [GROTER]

We moeten allereerst de tekstdelen in oplopende grootte onderscheiden:

letter - woord - regel - alinea - gehele tekst

Het groter maken van de focus bereik je nu door de toets [GROTER] te gebruiken (let op het verschil met [SHIFT]-[F1]). Als de focus b.v. op een woord staat, zal het intoetsen van [GROTER] de focus op het tekstgedeelte zetten dat één stap groter is, dus op de hele regel.

Voorbeeld a.

[GROTER] [GROTER] [GROTER] heeft als resultaat, dat de focus, die eerst op een enkel woord stond, nu op de gehele tekst staat.

Je ziet dat de laatste regel van een alinea een lege regel is.

b) De focus op het eerste deel zetten: [EERSTE]

Ook hier moeten we dezelfde tekstdelen onderscheiden als in a). De delen van een woord zijn de letters, de delen van een regel zijn de woorden etc.. Gebruiken we de toets [EERSTE] dan zal de focus op het eerste deel worden gezet. Staat de focus b.v. op een alinea, dan zal [EERSTE] tot gevolg hebben dat de focus op de eerste regel van die alinea wordt geplaatst.

Voorbeeld b.

[EERSTE] [EERSTE] [EERSTE] [EERSTE] zal de focus nu op de eerste letter van de file zetten.

c) De focus op het laatste deel zetten : [LAATSTE]

Een aanslag van de toets [LAATSTE] plaatst de focus op het laatste deel. Als de focus b.v. op een woord staat, zal door [LAATSTE] de laatste letter in de focus komen.

Voorbeeld c.1.

[GROTER] [GROTER] [LAATSTE] verplaatst de focus naar het woord **kampioen** op de eerste regel.

Voorbeeld c.2.

[GROTER] [GROTER] [GROTER] [LAATSTE] [EERSTE] [EERSTE] zet het woordje **direkt** in de focus.

d) De focus langer maken: [LANGER]

Deze verplaatsing van de focus lijkt wel wat op die in a). Maar let op het verschil: als de focus b.v. op een woord staat, dan zal [LANGER], zo mogelijk, het volgende woord erbij voegen. Staat de focus b.v. op de *laatste* alinea, dan zal [LANGER] ook de voorlaatste alinea inschuiven.

Voorbeeld d.1.

[LANGER] [LANGER] plaatst **direkt na afloop** in de focus.

Voorbeeld d.2.

[GROTER] [LANGER] [LANGER] zal de focus op de eerste drie regels van de laatste alinea zetten.

Voorbeeld d.3.

[LANGER] [LANGER] [LANGER] zal de laatste drie alinea's in de focus schuiven.

e) De focus op het vorige deel plaatsen: [VORIGE]

Als de focus op een regel staat, zal een toetsaanslag [VORIGE] de focus op de vorige regel plaatsen.

Voorbeeld e.1.

[EERSTE] [VORIGE] zal de focus op de eerste alinea zetten.

Voorbeeld e.2.

[LANGER] [LAATSTE] [LAATSTE] [VORIGE] [VORIGE] zal de tweede regel van de tweede alinea in de focus zetten.

f) De focus op het volgende deel plaatsen: [VOLGENDE]

Door het gebruik van [VOLGENDE] plaats je het volgende deel in de focus.

Voorbeeld f.

[GROTER] [VOLGENDE] [EERSTE] plaatst de eerste regel van de derde alinea in de focus.

g) De focus zoekt zelf voorwaarts een tekstdeel: [VOORUIT]

Als je [VOORUIT] intoetst, verschijnt er onderaan het scherm:

```
Search :
```

Het woord of het zinsdeel dat je in de focus wilt plaatsen, kun je nu intoetsen. Gebruik je opnieuw [VOORUIT] dan zoekt de focus vanaf de plaats waar hij nu staat, verder in de tekst of hij het zojuist ingetoetste kan vinden. Zo ja, dan gaat hij erop staan, zoniet, dan verschijnt er:

```
Search unsuccessful
```

Voorbeeld g.1.

[VOORUIT] `kostbare` [VOORUIT] zal de focus op het woord `kostbare` plaatsen.

Voorbeeld g.2.

[VOORUIT] `gedrang` [VOORUIT] zal onderaan op het scherm zetten:

```
Search unsuccessful: gedrang
```

h) De focus zoekt achterwaarts een tekstdeel: [TERUG]

Het enige verschil met [VOORUIT] is, dat de focus bij gebruik van [TERUG] het daarna ingetoetste gedeelte opzoekt, vanaf de plaats waar hij staat tot aan het begin van de file. Er wordt dan ook vermeld:

```
Backward Search:
```

Voorbeeld h.1.

[TERUG] `gedrang` [TERUG] zal nu de focus wel op het woord `gedrang` zetten.

Voorbeeld h.2.

[TERUG] `kostbare` [TERUG] geeft daarentegen op het scherm te zien:

```
Backward Search unsuccessful : kostbare
```

i) Een focus-verplaatsing ongedaan maken: [UNDO]

Het resultaat van de [UNDO]-toets lijkt veel op dat van [BACKSPACE]. Bij [BACKSPACE] wordt het intoetsen van een letter (symbool) ongedaan gemaakt. [UNDO] zorgt ervoor dat de toestand die er was voordat je de laatste toets gebruikte wordt hersteld, ook als de laatst gebruikte toets een [F]-toets was. Als je [UNDO] nogmaals gebruikt, verschijnt de voorlaatste toestand, etc.

Voorbeeld i.1.

[GROTER] [LAATSTE] [UNDO] zet de vierde regel van de derde alinea in de focus.

Voorbeeld i.2.

[GROTER] [EERSTE] [LAATSTE] [UNDO] [UNDO] [UNDO] laat de toestand ongewijzigd.

j) Het resultaat van [UNDO] ongedaan maken: [REDO]

Het kan gebeuren dat je teveel resultaten van toetsaanslagen met [UNDO] ongedaan hebt gemaakt. Het gebruik van [REDO] heft dat van [UNDO] weer op: de door [UNDO] uitgewiste toestand verschijnt opnieuw.

Voorbeeld j.

[GROTER] [VOLGENDE] [LAATSTE] [LAATSTE] [LANGER] [UNDO] [REDO] plaatst het woord `gedegradeerd` in de focus.

☞ 3.6.

a. Noteer de toetsaanslagen die je achtereenvolgens nodig zult hebben om de focus te verplaatsen:
   1. van `gedegradeerd` naar `persfotografen`;
   2. van `persfotografen` naar `zijn kostbare bezit`.

b. Controleer op de terminal, wat je in 3.6.a hebt genoteerd.

c. Plaats de focus op `gedegradeerd` en voer de focusverplaatsing zoals beschreven in 3.6.a uit met de zoektoets.

d. Beëindig het werken met λ `maak` aan de file `golf`.

3.4. Teksten wijzigen

In de vorige paragrafen is het belangrijkste - het wijzigen van teksten - nog steeds niet is behandeld. Welnu, na alle informatie over de focus, kan nu een antwoord worden gegeven op de vraag:

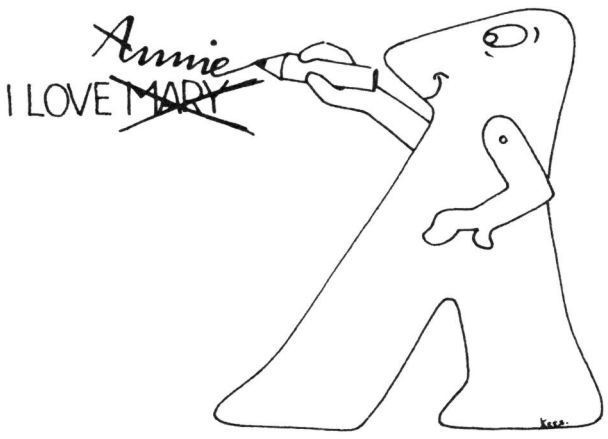
Hoe brengen we veranderingen in een tekst aan?

Het antwoord wordt gegeven in de volgende onderdelen: a t/m f.

☞ 3.7. Voer wat hieronder in de voorbeelden a t/m f besproken wordt weer direkt uit op je toetsenbord.

Opnieuw laten we λ `maak` op de file `golf` werken:

```
λ maak golf
Mickey ... ... ...
 ...
 ...
 ... ... ... gedegradeerd.
```



a) Tekst toevoegen

Als we de focus op een tekstdeel plaatsen en we toetsen daarna een letter in, dan wordt die letter direkt *voor* de focus in de tekst opgenomen.

Voorbeeld a.

We verplaatsen de focus naar het woord `tot` op de regel

```
kampioensbewijs tot .......
```

(je mag de focus ook op de eerste `t` van `tot` zetten). Nu toetsen we in:

```
door zijn vriendin
```

(let er op dat het laatste symbool een spatie is). Er staat nu:

```
kampioensbewijs door zijn vriendin tot .......
```

Als je tussen twee regels tekst wilt toevoegen, dan zet je de focus op de eerste letter van de tweede regel, daarna toets je [RETURN] in en vervolgens plaats je de focus met [VORIGE] op de gewenste positie. Wil je boven de eerste regel tekst toevoegen, dan zet je de focus op de eerste letter van de eerste regel en toets je vervolgens [RETURN] en [VORIGE].

b) Tekst weggooien: [DEL]

Ook kunnen we eenvoudig delen uit de tekst verwijderen. We plaatsen de focus op het deel (letter, woord etc.) dat eruit moet en slaan vervolgens de [DEL]-toets aan (to delete = schrappen). De inhoud van de focus is nu uit de tekst geschrapt. Indien er niets in de focus staat, zoals dat bij het intoetsen van een tekst het geval is, dan zal het gebruik van de [DEL]-toets de letter verwijderen die direkt *voor* de focus staat (vergelijk 3.2).

Voorbeeld b.

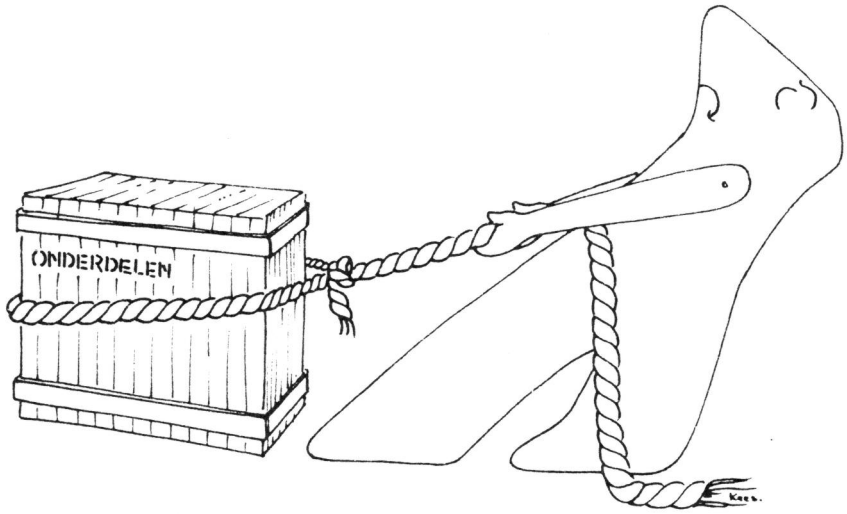
> We plaatsen de focus naar de eerste regel van de derde alinea en zetten `in groten getale aanwezige` in de focus. Vervolgens toetsen de [DEL] in en zien dan dat de betreffende regel is veranderd in: `Het publiek bleek bij.`

c) Regels aan elkaar plakken: [PLAK]

De [RETURN]-toets zorgt o.a. ervoor dat de letters die je na [RETURN] intoetst, op de volgende regel verschijnen. Het verwijderen van de [RETURN] heeft dus tot resultaat dat twee opeenvolgende regels aan elkaar geplakt worden. Dit gaat met de toets [PLAK]. Als je deze toets gebruikt, zal de [RETURN] verdwijnen aan het eind van de regel die vanaf het begin van de focus het eerst voorkomt.

Voorbeeld c.

> Plaats de focus op (een deel van) de eerste regel. [PLAK] heeft nu als resultaat dat de tweede regel er direkt achter komt. Als je dit met [UNDO] weer ongedaan maakt en je plaatst de hele eerste alinea in de focus, dan zal [PLAK] ook nu hetzelfde resultaat geven.



d) Tekstdelen verplaatsen: [DEL], [uit BUFFER]

In onderdeel b) kwam ter sprake hoe een tekstgedeelte geschrapt moet worden. We komen daar nu nog even op terug. Bij het gebruik van [DEL] wordt de inhoud van de focus uit de file verwijderd. Maar die inhoud blijft wel ergens bewaard, nl. in de *buffer*. Door nu de focus ergens anders te zetten, kunnen we het zojuist geschrapte tekstdeel op die plaats in de tekst neerzetten: het intoetsen van [uit BUFFER] plaatst de inhoud van de buffer direkt *voor* de focus in de tekst.

Voorbeeld d.

> We plaatsen `(van onze sportverslaggever).` in de focus. Nu verwijderen we deze inhoud met [DEL]. Vervolgens zetten we de focus vooraan de eerste lege regel onder de tekst. We toetsen [uit BUFFER] in en zien dat het bewuste tekstdeel nu onder de gehele tekst staat.

e) Tekstdelen copiëren: [in BUFFER], [uit BUFFER]

De handelingen die worden verricht om een copie van een tekstgedeelte op een andere plaats te zetten, lijken erg veel op die welke je gebruikt om een tekstgedeelte te verplaatsen. We zetten de focus op een tekstdeel en toetsen vervolgens [in BUFFER] in. Onderaan het scherm staat nu aangegeven dat dit tekstgedeelte in de buffer zit. Vanaf nu zijn de handelingen gelijk aan die bij het verplaatsen: de focus wordt naar de gewenste plaats gestuurd, waarna [uit BUFFER] de inhoud van de buffer vóór de focus plaatst.

Voorbeeld e.1.

> We plaatsen de laatste regel van de file in de focus. Nu toetsen we [in BUFFER] in, zodat deze regel in de buffer staat. We zetten de focus nu onder de eerste regel van de file, vooraan de regel, en gebruiken [uit BUFFER]. Het resultaat is dat we nu zowel op de tweede als op de laatste regel zien dat het artikel door de sportverslaggever is geschreven.

Er is nog een toepassing van het copiëren te vermelden. We plaatsen een tekstregel in de buffer. Vervolgens beëindigen we het werken met λ **maak** aan deze file en laten daarna λ **maak** op een andere file los. Het aardige is nu dat de buffer nog steeds dezelfde inhoud heeft. We maken daar nu gebruik van door in de andere file deze inhoud op de gewenste plaats neer te zetten.

Voorbeeld e.2.

> De totale inhoud van de file **golf** zetten we in de focus en daarna — met [in BUFFER] — in de buffer. Nu beëindigen we met λ **maak** het werken aan de file **golf**. Op gemeenschappelijk gebied staat de file **sportpagina**. Deze file zetten we nu eerst in onze eigen werkruimte. Daarna laten we λ **maak** op **sportpagina** werken. De focus wordt nu verplaatst tot onder het artikel van **FC. Houtenbeen 04**. Nog steeds staat de totale inhoud van de file **golf** in de buffer. Intoetsen van [uit BUFFER] zal nu dan ook tot gevolg hebben dat het golfverslag tussen het voetballen en het tennissen wordt opgenomen.

f) Tekstdelen globaal vervangen: [VERVANG]

Soms bevat een artikel een aantal keren dezelfde foutieve naam. We zijn nu wel in staat om die naam op iedere plaats te wijzigen door eerst de focus er naar toe te sturen, maar er is een nog simpeler methode. Deze wordt in het onderstaande voorbeeld besproken.

Voorbeeld f.

> De verslaggever heeft een fout gemaakt. Niet Mickey Mep maar Molly Mep blijkt de tennisser te zijn. Deze fout gaan we nu herstellen. We beginnen met het verplaatsen van de focus en zetten hem op het eerste woord van de file. Nu toetsen we [VERVANG] in en dan verschijnt er:
>
> ```
> Replace-search:
> ```
>
> Als je nu een woord intoetst, weet λ **maak** dat dit woord op verschillende plaatsen in de tekst vervangen moet worden. We toetsen nu in:
>
> ```
> Mickey
> ```
>
> Vervolgens gebruiken we weer de [VERVANG]-toets, waarna de focus direkt de eerste **Mickey** vangt. Nu verschijnt de vraag:
>
> ```
> Replace by:
> ```
>
> Het intoetsen van:
>
> ```
> Molly
> ```
>
> wil nu dus zeggen, dat **Mickey** vervangen moet worden door **Molly**. Maar niet overal; vandaar dat de vraag verschijnt:
>
> ```
> Change?
> ```
>
> Nu zijn er twee mogelijkheden. Als we intoetsen:

```
y
```

(van *yes*) dan zal `Mickey` inderdaad door `Molly` worden vervangen en gaat de focus op de volgende `Mickey` staan. Als we intoetsen:

```
n
```

(van *no*) dan zal de focus direkt de volgende `Mickey` opzoeken, zonder dat de huidige wordt veranderd. In beide gevallen verschijnt er opnieuw:

```
Change?
```

Dit gaat zo door totdat alle `Mickeys` voor vervanging zijn voorgedragen.

We beëindigen hiermee het werken aan de file `sportpagina`.

☞ 3.8. Tijdens een vergadering wordt er door de secretaris een verslag gemaakt: de *notulen*. Op de volgende vergadering worden deze notulen besproken. Meestal worden de notulen ongewijzigd goedgekeurd (gearresteerd), maar soms heeft men aanmerkingen op het werk van de secretaris. Een voorbeeld van zo'n verslag staat in de file `notulen`. Copieer deze file vanaf het gemeenschappelijke terrein naar je eigen werkruimte. De opmerkingen uit de vergadering onder de punten a t/m f (overeenkomstig de bespreking eerder in deze paragraaf) geven de secretaris nogal wat werk. Help hem door meteen de verandering in de notulen aan te brengen.

a. Meneer De Vorst vindt dat er na `woensdag` de datum `20-2-1985` moet worden toegevoegd. Iedereen is het daar mee eens, dus de secretaris zal het uitvoeren.

b. Mevrouw De Winter betreurt het dat het begin van de tweede alinea tot aan `Vervolgens ...` in de notulen staan vermeld. Na een heftige discussie wordt besloten deze sfeertekening te laten vervallen. Het woord `Vervolgens` zal daarna dan wel door `Eerst` moeten worden vervangen.

c. Gytemy stelt dat de zin `Daarmee ...` eigenlijk direkt achter de vorige geplaatst moet worden. Dit gaat zelfs een lerarenvergadering te ver en zal dan ook niet worden uitgevoerd.

d. Het leek voorts verschillende docenten beter om de woorden `ingevet` en `geslepen` in de laatste alinea te verwisselen. De licht blozende secretaris heeft dit onmiddellijk toegezegd.

e. De heer Y.S. Hockey meende dat het beter was om de brief van de leerlingen in zijn geheel in de notulen op te nemen en wel direkt na het gedeelte: `... reactie van de leerlingen`. Bijna iedereen kon zich in deze gedachte vinden. Om hier de secretaris een handje te helpen, moet je weten dat de file `brief` op het gemeenschappelijk gebied staat.

f. De jonge leerkracht Cor Rekt, docent in de Ylandse taal- en letterkunde, vroeg zich af of de tekstverwerker van de secretaris de combinatie van de `i` met de `j` niet kon produceren. Hij zag erg graag dat `y` in de meeste gevallen vervangen zou worden door `ij`. Ook aan deze wens zal de secretaris voldoen.

☞ 3.9. Bespreek de voor- en nadelen van de tekstverwerking zoals je die in dit hoofdstuk hebt geleerd.

3.5. Overzicht

λ **maak**	naam van de editor
buffer	plaats waar een copie van (een deel van de) file wordt bewaard
editor	computerprogramma voor het werken met teksten
focus	verlichte deel van het scherm
tekstverwerker	computerprogramma voor het werken met teksten
[SHIFT]-[F1]	verlaat de editor en bergt de file op
[GROTER]	vergroot de focus
[LANGER]	verlengt de focus
[EERSTE]	zet de focus op het eerste deel
[LAATSTE]	zet de focus op het laatste deel
[VORIGE]	zet de focus op het vorige deel
[VOLGENDE]	zet de focus op het volgende deel
[in BUFFER]	zet de inhoud van de focus in de buffer
[DEL]	schrapt de inhoud van de focus en plaatst die in de buffer
[uit BUFFER]	zet de inhoud van de buffer in de focus
[PLAK]	verwijdert de [RETURN]
[VERVANG]	vervangt vaker hetzelfde tekstdeel
[VOORUIT]	zoekt voorwaarts een tekstdeel
[TERUG]	zoekt achterwaarts een tekstdeel
[UNDO]	doet de werking van een toets teniet
[REDO]	doet de werking van de [UNDO]-toets teniet

4. Tekstopmaak

4.1. Inleiding

In het vorige hoofdstuk ben je als journalist bezig geweest. Je hebt een verslag geschreven, dat je vervolgens ook nog moest wijzigen om een zo goed mogelijke inhoud te krijgen. Vroeger kon jouw artikel dan in dezelfde vorm in de krant worden afgedrukt. Als je dan ook heel oude dagbladen bekijkt, merk je dat er bijna uitsluitend tekst instaat en er geen plaatjes of foto's zijn opgenomen. Tegenwoordig vinden we dat veel te saai en daarom wordt er bij onze kranten veel aandacht besteed aan de vorm waarin het artikel wordt opgenomen: hoe maken we ruimte voor plaatjes, hoe zetten we een kopje boven een tekst, enz. In dit hoofdstuk gaan wij ons ook bezighouden met de vorm, waarin het artikel uiteindelijk zal worden afgedrukt. De volgende opdracht is bedoeld als inleidende oefening hiertoe.

☞ 4.1. Op het gemeenschappelijk gebied staat de file `proeftekst`. Zet deze op je eigen gebied, geef hem een nieuwe naam en bekijk de inhoud. Met λ `maak` ga je nu de vorm van deze file als volgt veranderen:

a. zin 1 t/m 6 blijven onveranderd.

b. zin 7 t/m 11 komen achter elkaar over de hele breedte van het scherm.

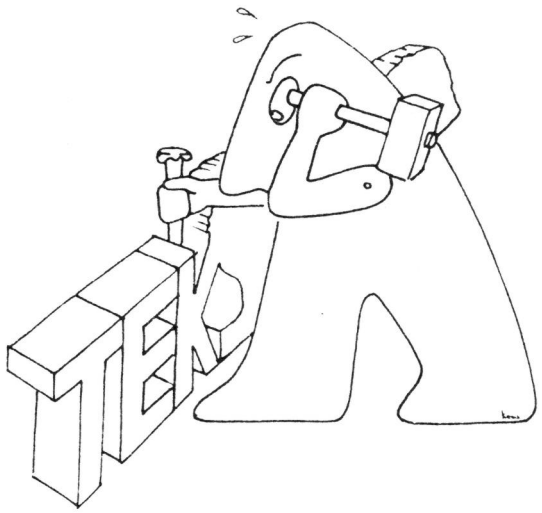
c. zin 12 t/m 14 worden achter elkaar rechts op het scherm geschreven, waarbij de linker kantlijn 60 spaties is opgeschoven.

d. zin 15 t/m 18 worden achter elkaar over de hele breedte afgedrukt.

e. zin 19 t/m 22 blijven onveranderd.

Om deze vorm te realiseren kun je [PLAK] gebruiken. Verder kun je woorden afbreken en spaties invoegen, om de tekst bij b, c en d een rechte rechter kantlijn te geven.

4.2. De tekstopmaker: λ vorm



Je hebt gemerkt, dat het een heel karwei is om de tekst met λ `maak` in de gewenste vorm te krijgen. Je zult nu merken dat de softwerker λ `vorm` je hierbij een veel betere dienst kan bewijzen.

☞ 4.2. Copieer de file `proef` opnieuw naar je werkgebied en bekijk zijn simpele inhoud nog even. Toets vervolgens in:

```
λ vorm proef
```

(Gebruik eventueel de [NO SCROLL]-toets om de tekst even stil te zetten op het scherm.)

Het enige verschil, dat je tussen λ `toon` en λ `vorm` hebt opgemerkt, is dat λ `vorm` een aantal lege regels geeft.
Toch is het verschil tussen beide softwerkers veel groter dan het nu lijkt. We kunnen λ `vorm` nl. extra instrukties meegeven. Hoe dat gaat en welke instrukties je kunt geven, wordt in de volgende onderdelen a t/m o behandeld.

☞ 4.3. Voer wat in de voorbeelden onder a t/m o wordt besproken, direkt uit op je toetsenbord.

a) Regels volschrijven: .rv

In opdracht 4.1. moest je de file `proef` zo vervormen, dat vanaf de zevende regel de tekst op schermbreedte werd afgedrukt. Dat proberen we nu weer.

Voorbeeld a.

Gebruik λ `maak` om tussen de 6e en 7e zin van de file `proef` de vorm-instruktie:

```
.rv
```

op te nemen (zet de focus op de laatste letter van de 6e zin, toets [RETURN], dan gaat de focus naar de zo ontstane lege regel en toets vervolgens `.rv`). Nu verlaat je λ `maak`. Vervolgens bekijk je de inhoud van de file door λ `toon proef` in te toetsen. Dat geeft:

```
Dit is de eerste volzin van de proeftekst.
Dit is de tweede volzin van de proeftekst.
...
Dit is de zesde  volzin van de proeftekst.
.rv
Dit is de zevende volzin van de proeftekst.
Dit is de achtste volzin van de proeftekst.
Dit is de negende volzin van de proeftekst.
...
...
...
Dit is de tweeentwintigste volzin van de proeftekst.
```

Als we nu λ `vorm proef` intoetsen, krijgen we iets anders te zien:

```
Dit is de eerste volzin van de proeftekst.
Dit is de tweede volzin van de proeftekst.
...
Dit is de zesde  volzin van de proeftekst.
Dit is de zevende volzin van de proeftekst.  Dit is de achtste
volzin van de proeftekst.  Dit is de negende volzin van de
proeftekst. ... ... ...  Dit is de tweeentwintigste volzin van de
proeftekst.
```

Nu heb je een belangrijk verschil gemerkt tussen λ `toon` en λ `vorm`. λ `toon` laat alles zien wat er in de file staat, maar λ `vorm` gaat wat fijner te werk. Hij laat de inhoud van de file ook wel zien (vergelijk zin 1 t/m 6), echter wel met een uitzondering. Als er een regel begint met een "." dan *weet* λ `vorm`, dat hij deze regel niet moet afdrukken. Hier krijgt hij namelijk een instruktie over de manier waarop hij de volgende regels in de file moet afdrukken. Zo zal hij dus vanaf de 7e zin alle regels op de volle breedte van het scherm afdrukken. Merk op dat er nu automatisch, waar nodig, woorden worden afgebroken en spaties worden toegevoegd.

b) Regels niet vol schrijven: .nv

Wanneer de regels in de oorspronkelijke lengte moeten worden afgedrukt, behoeft de volle regelbreedte (schermbreedte) dus niet te worden benut. De vorm-instruktie:

```
.nv
```

staat hiervoor garant.

Voorbeeld b.

Neem met λ `maak` de instruktie `.nv` op, tussen de 18e en de 19e zin van de file `proef`. Met λ `toon proef` zien we dan onder meer:

```
...
Dit is de achttiende volzin van de proeftekst.
.nv
Dit is de negentiende volzin van de proeftekst.
...
```

Controleer het nu met λ `vorm`.

c) Tekst laten inspringen: .in

Er bestaat ook een vorm-instruktie, waarbij de tekst ingesprongen wordt afgedrukt. Zo zal er na de instruktie:

```
.in 15
```

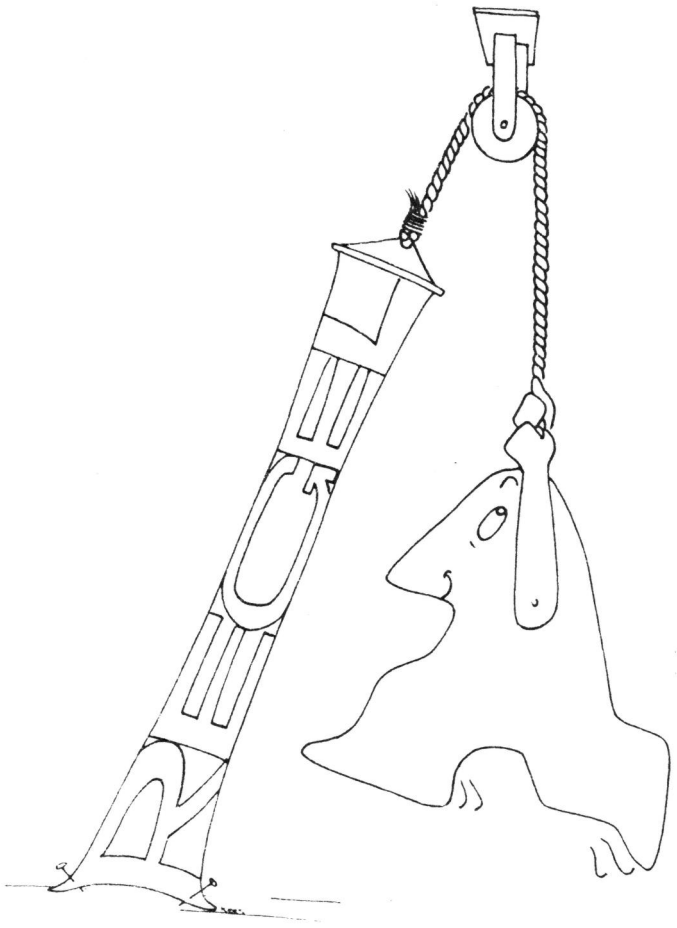
bij het afdrukken van de rest van de tekst, de kantlijn 15 posities naar rechts worden verschoven.

Voorbeeld c.

Voeg met λ `maak` na de 11e volzin de vorm-instruktie `.in 60` toe en zet direkt na de 14e zin de instruktie `.in 0`. Er verschijnt dan met λ `toon proef`:

```
...
Dit is de elfde volzin van de proeftekst.
.in 60
Dit is de twaalfde volzin van de proeftekst.
Dit is de dertiende volzin van de proeftekst.
Dit is de veertiende volzin van de proeftekst.
.in 0
Dit is de vijftiende volzin van de proeftekst
...
```

Ga je nu λ `vorm proef` uitvoeren, dan zul je zien dat de kantlijn voor de elfde t/m de veertiende zin 60 plaatsen is opgeschoven. Met de instruktie `.in 0` kun je `.in 60` uiteraard weer ongedaan maken.



☞ 4.4. Vergelijk het resultaat van λ `vorm proef` met de vorm van de file, die je in opdracht 4.1 met λ `maak` hebt verkregen. Als je alles goed hebt uitgevoerd, zullen deze resultaten goed overeenstemmen.

d) Verander de regellengte: .rl

Als deze instruktie niet wordt opgenomen dan zal de regellengte 79 posities zijn: de breedte van het scherm. Maar we zijn in staat om de regels een andere lengte te geven. Vanaf het moment dat de vorm-instruktie `.rl 52` in de tekst wordt opgenomen, zullen alle regels maximaal 52 posities innemen. N.B. De instruktie `.rl` werkt alleen maar na `.rv` en niet na `.nv`.

Voorbeeld d.

Breng met λ `maak` de volgende vorm-instrukties aan:

1. Tussen de 6e en 7e volzin: `.rl 60`
2. Tussen de 11e en de 12e volzin: `.rl 79`

We zien nu na λ `toon proef`:

```
...
Dit is de zesde volzin van de proeftekst.
.rv
.rl 60
Dit is de zevende volzin van de proeftekst.
...
Dit is de elfde volzin van de proeftekst.
.rl 79
.in 60
Dit is de twaalfde volzin van de proeftekst.
...
```

Met λ `vorm proef` zul je nu zien dat de 7e t/m de 11e volzin op regellengte van 60 posities worden afgedrukt.

e) Rechts wel/niet uitlijnen: .wu , .nu

Een bijkomend gevolg van sommige vorm-instrukties zoals van `.rv` is, dat er een rechte rechter kantlijn wordt gemaakt. Meestal is dat wel prettig, maar soms wil je dat niet. Ook hier kun je dan een instruktie gebruiken. Als je de tekst niet wilt uitlijnen, want zo wordt het creëren van een rechte kantlijn genoemd, kun je de vorm-instruktie `.nu` opnemen. Wil je daarna weer een rechte kantlijn dan helpt `.wu`.

Voorbeeld e.

Zet met λ `maak` de instruktie `.nu` voor de zevende volzin en plaats .wu voor de twaalfde. Je ziet dan met λ `toon proef`:

```
...
.rl 60
.nu
Dit is de zevende volzin van de proeftekst.
...
.in 60
.wu
Dit is de twaalfde volzin van de proeftekst.
```

Ga met λ `vorm` na, dat er tussen de 7e en de 12e zin geen rechter kantlijn is.

f) Woorden niet/wel afbreken: .na , .wa

Net zoals bij het uitlijnen, worden de woorden, waar nodig, afgebroken. Ook hier kun je dat tegengaan met een vorm-instruktie: `.na`. Nadat je de instruktie `.wa` hebt opgenomen, zal er weer afgebroken worden.

Voorbeeld f.

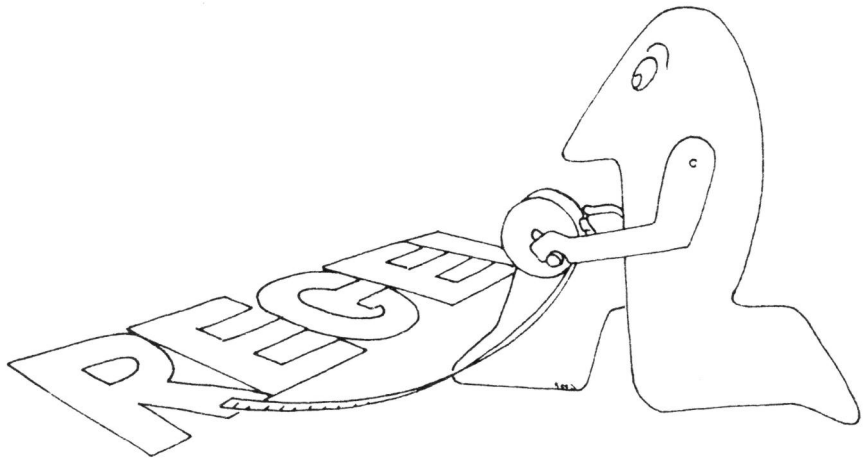
Zet met λ `maak` de instruktie `.na` voor de 14e volzin. Je ziet dan:

```
λ toon proef
...
Dit is de dertiende volzin van de proeftekst.
.na
Dit is de veertiende volzin van de proeftekst.
...
```

Met λ `vorm proef` zal nu blijken, dat het woord `veertiende` niet meer wordt afgebroken.

g) Regelafstand bepalen: .ra

De vorm-instruktie `.ra 3` zorgt ervoor dat de tekst verder wordt afgedrukt met 2 regels tussenruimte. Zo zal dus `.ra 1` weer voor de oorspronkelijke toestand zorgen.



Voorbeeld g.

Neem met λ `maak` de volgende vorm-instrukties op:

1. `.ra 2` na de 2e zin
2. `.ra 1` na de 6e zin

Met λ `toon proef` moet de inhoud van de file er dus als volgt uitzien:

```
Dit is de tweede volzin van de proeftekst.
.ra 2
Dit is de derde volzin van de proeftekst.
...
Dit is de zesde volzin van de proeftekst.
.ra 1
.rv
```

Met λ `vorm` merk je nu, dat er tussen de 2e en de 6e regel telkens een lege regel is opgenomen.

h) Regels overslaan: .ro

In tegenstelling tot `.ra`, wordt bij de vorm-instruktie `.ro` eenmalig een aantal lege regels gegeven. Zo zal `.ro 5` voor de volgende tekstregel 5 regels blank laten.

Voorbeeld h.

Gebruik opnieuw λ `maak` om na de 1e zin de instruktie `.ro 4` op te nemen. In de file `proef` staat nu dus o.a.

```
Dit is de eerste volzin van de proeftekst.
.ro 4
Dit is de tweede volzin van de proeftekst.
```

Tussen de eerste en de tweede regel slaat λ `vorm` nu dus vier regels over.

i) Tijdelijk inspringen: .ti

De instruktie `.in 15` zorgde ervoor dat voor de verdere tekst de kantlijn 15 plaatsen naar rechts werd verschoven. De instruktie `.ti` geeft echter alleen vooraan de eerstvolgende regel 15 spaties.

Voorbeeld i.

Voeg voor de 15e volzin de instruktie `.ti 10` toe. Er staat dan o.m. in de file:

```
.in 0
.ti 10
Dit is de vijftiende volzin van de proeftekst.
```

Met λ `vorm` wordt de tekst nu afgedrukt, waarbij alleen de 15e regel tien posities is ingesprongen.

j) Begin op een nieuwe regel: .nr

De instruktie `.nr` zorgt ervoor dat de volgende zin op een nieuwe regel begint.

Voorbeeld j.

Plaats de vorm-instruktie `.nr` tussen de 16e en de 17e zin. Er staat dan dus o.a. in de file:

```
Dit is de zestiende  volzin van de proeftekst.
.nr
Dit is de zeventiende volzin van de proeftekst.
```

Je merkt dat λ `vorm` de 17e zin nu vooraan de regel afdrukt.

k) Bepaal de paginalengte: .pl

Je hebt steeds bij het gebruik van λ `vorm` gezien dat er een aantal lege regels wordt afgedrukt. Dat komt omdat λ `vorm` eigenlijk pagina's afdrukt. We kunnen dat duidelijk zien als we de lengte van de pagina's veranderen.

☞ 4.5.

a. Bepaal het aantal regels van de uitvoer van λ `vorm proef`, door deze uitvoer eerst in een file `hulp` te zetten (zie 2.8).

b. Verwijder na de 22e zin alle lege regels van `hulp` en bepaal opnieuw het aantal regels van de file.

Als je deze opdrachten goed hebt uitgevoerd, dan heb je gezien dat λ `vorm` 66 regels uitvoer geeft. Dus dat is het aantal regels van een pagina. Nadat je de lege regels hebt verwijderd, hield je er 41 over.

Voorbeeld k.

We gaan nu proberen om de paginalengte op 50 regels te stellen, want dat moet voor deze tekst voldoende zijn. Je doet dat door de vorm-instruktie `.pl 50` vooraan de file op te nemen. Er staat dan:

```
.pl 50
Dit is de eerste volzin van de proeftekst.
...
```

Je merkt dat λ `vorm` nu maar een paar lege regels geeft: de tekst en een aantal blanke regels waarmee de eerste pagina van 50 regels wordt volgemaakt. (De paginalengte moet minimaal 24 regels zijn).

l) Begin op een nieuwe pagina: .np

Als de tekst op meer dan een pagina wordt afgedrukt, kan het zijn dat de volgende pagina op een minder voor de hand liggende plaats begint, b.v. op de tweede regel van een alinea. In dat geval zou je de totale alinea op de nieuwe pagina willen afdrukken. Hiervoor kun je de vorm-instruktie `.np` gebruiken.

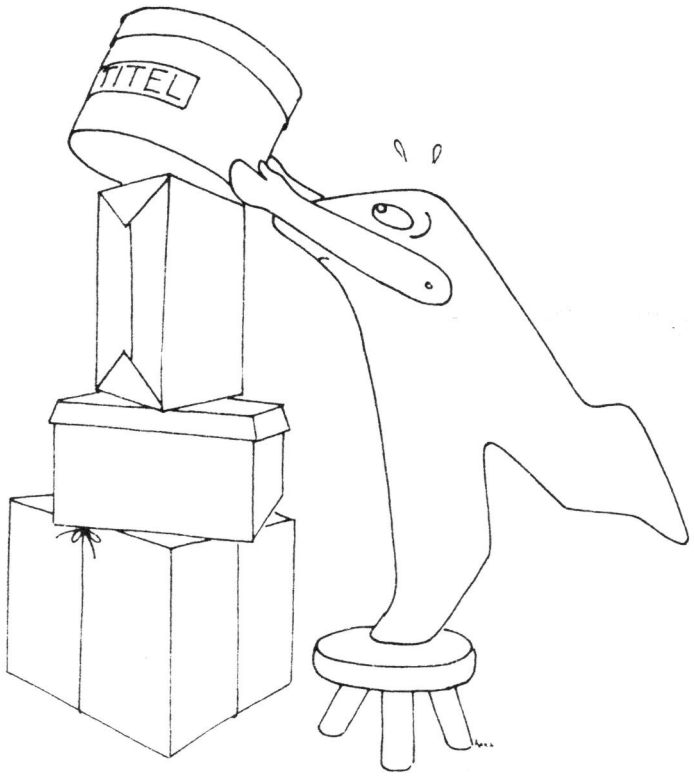
Voorbeeld l.

Verander de instruktie `.pl 50` in `.pl 37` en bekijk het resultaat met λ `vorm`. Nu wordt het begin van de 15e volzin op de eerste pagina afgedrukt en de rest op de volgende. Voeg daarom voor de 15e zin de vorm-instruktie `.np` toe; dan staat er dus o.a. in de file:

```
.ti 10
.np
Dit is de vijftiende volzin van de proeftekst.
```

De instruktie λ `vorm proef` heeft tot resultaat dat de 2e pagina met de 15e volzin begint.

m) Titel plaatsen aan de kop van de pagina: .pt



Je hebt net wel aangenomen, dat de 2e pagina begint met de 15e volzin, maar erg duidelijk was het niet. Als we de mogelijkheid gebruiken om iedere pagina van een titel te voorzien, dan is o.a. deze moeilijkheid opgelost.

Voorbeeld m.

Laten we boven de eerste pagina zetten:

```
vorm                    -1-                    proeftekst
```

Je ziet dat deze titel uit drie delen bestaat. Je kunt dit als volgt in een vorm-instruktie opnemen:

```
.pt "vorm"-1-"proeftekst"
```

Het vervelende is echter dat nu alle pagina's als pagina `-1-` worden genummerd. Daarom gebruiken we de instruktie:

```
.pt "vorm"-%-"proeftekst"
```

omdat het %-teken het nummer van de pagina aangeeft. Neem deze vorm-instruktie in de file op. Er staat dan o.a.:

```
.pl 37
.pt "vorm"-%-"proeftekst"
```

Kontroleer nu het resultaat met λ `vorm`.

☞ 4.6. Verander de instruktie zodanig, dat de paginatitels alleen bestaan uit de nummering in het midden bovenaan.

n) Regels centreren: .ce

Met de vorm-instruktie `.ce 3` worden de volgende drie tekstregels op het midden van de pagina afgedrukt.

Voorbeeld n.

Plaats de instruktie `.ce 6` voor de eerste zin zodat er dan staat:

```
.pt ""-%-""
.ce 6
Dit is de eerste volzin van de proeftekst.
```

(Je hebt nu meteen antwoord op opdracht 4.5.) Zin 1 t/m 6 worden nu midden op de pagina afgedrukt.

o) Nieuwe instrukties definieren: .de

Zo langzamerhand zijn er al heel wat instrukties in de file opgenomen. Als er dus een manier is om dit aantal te beperken, dan is die welkom. Wel nu, hiervoor is een mogelijkheid aanwezig. Stel dat je een aantal keren dezelfde kombinatie van instrukties moet gebruiken, dan kun je deze kombinatie samenvoegen tot een nieuwe instruktie.

Voorbeeld o.

Aan het begin van een nieuwe alinea zetten we een lege regel en vervolgens springen we eenmalig 10 posities in. Deze twee instrukties voeg je nu als volgt samen tot een nieuwe:

```
.de al
.ro 1
.ti 10
..
```

Let erop dat de kombinatie wordt afgesloten met twee punten op een regel apart. Neem deze vier regels op bovenaan in de file. Hierna kun je nieuwe instruktie `.al` gebruiken, waar je maar wilt. Er staat nu b.v. in de file:

```
...
...
.de al
.ro 1
.ti 10
..
Dit is de eerste volzin van de proeftekst.
...
...
.nu
.al
Dit is de zevende volzin van de proeftekst.
...
...
.nv
.al
Dit is de negentiende volzin van de proeftekst.
...
...
```

Hiermee is het werken aan de file **proef** beeindigd.

☞ 4.7. Copieer de file **gedicht** uit het gemeenschappelijke gebied naar je eigen werkruimte. Voeg de nodige vorm-instrukties toe, zodat λ **vorm gedicht** er uit gaat zien, zoals hieronder staat beschreven:

a. In totaal bestaat **gedicht** uit 8 coupletten met elk 4 regels. Plaats op iedere pagina 2 coupletten.

b. Alle coupletten zijn 25 posities breed.

c. De linker kantlijn van het eerste couplet op een pagina staat midden op het scherm.

d. De rechter kantlijn van het tweede couplet op een pagina staat midden op het scherm.

e. Iedere pagina heeft als titel:

```
Hoekstra                    -N-                    Complexen
```

waarbij **N** het nummer van de pagina is.

4.3. Overzicht

λ `vorm`	de tekstopmaker
`.nv`	regels niet vol schrijven †
`.rv`	regels volschrijven
`.wu`	rechts wel uitlijnen †
`.nu`	rechts niet uitlijnen
`.wa`	woorden wel afbreken †
`.na`	woorden niet afbreken
`.pl N`‡	de paginalengte wordt **N** (aanvankelijk 66)
`.pt "l"m"r"`	pagina titel (**%** wordt paginanummer)
`.rl N`	de regellengte wordt **N** (aanvankelijk 79)
`.ra N`	de regelafstand wordt **N** (aanvankelijk 1)
`.in N`	spring in **N** posities naar rechts
`.ti N`	spring tijdelijk in **N** posities naar rechts
`.ro N`	sla **N** regels over
`.ce N`	centreer **N** regels
`.np`	begin op een nieuwe pagina
`.nr`	begin op een nieuwe regel
`.de xx`	definieer nieuwe instruktie `.xx`
`..`	einde definitie

† Aanvankelijk geldt dit.

‡ Vul voor **N** een getal in.

5. Gegevensbestanden

5.1. Inleiding

We hebben het geheugen van de computer tot nu toe vooral gebruikt om pagina's tekst te bewaren. Het is ook mogelijk het geheugen te gebruiken als een soort kaartenbak, waarin de computer gegevens voor ons kan vastleggen en waarin hij voor ons kan zoeken.

In dit hoofdstuk leren we werken met een systeem voor het opbergen van adressen en telefoonnummers in een kaartenbak. We zullen mogelijkheden zien om nieuwe kaarten toe te voegen, bestaande kaarten te corrigeren of helemaal te verwijderen, te zoeken naar kaarten waarop een bepaald stukje tekst voorkomt, en lijsten te laten maken van bepaalde adressen.

5.2. Het adres-systeem

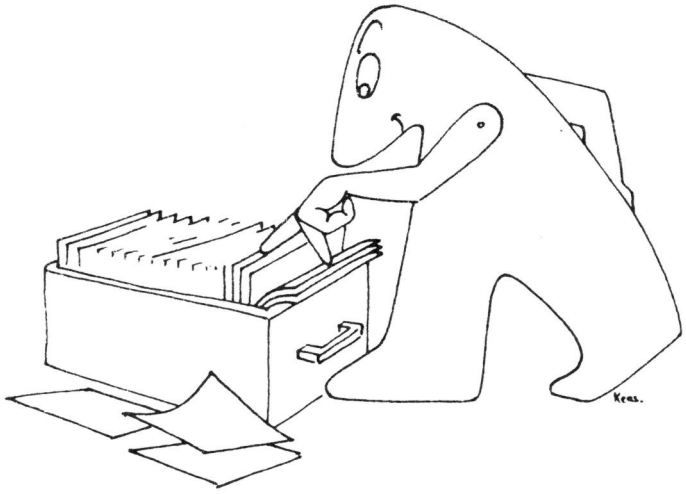
Om met het kaartenbak-systeem te werken, moeten we het softwerkertje λ `adres` oproepen, dat zo antwoordt:

```
λ adres
>>>
```

Aan λ `adres` kunnen we nu het commando **MENU** (met hoofdletters) geven:

```
>>> MENU
```

☞ 5.1 Voer de boven beschreven handelingen zelf op het toetsenbord uit. Volg ook de rest van dit hoofdstuk door de getoonde commando's zelf in te toetsen.



5.3. Werken met de kaartenbak

Het `adres`-softwerkertje voert het commando **MENU** uit door een menu van mogelijkheden op het scherm te zetten:

```
------------------------------------------------------------
Wat wil je doen: n. nieuw adres toevoegen
                 s. schrappen van bestaand adres
                 c. corrigeren van bestaand adres
                 t. tonen van een of meer adressen
                 z. zoeken naar adres met bepaald tekstje
                 o. ophouden
Geef de letter van je keuze: ?
```

We kunnen nu een van de zes getoonde mogelijkheden kiezen door de bijbehorende beginletter in te toetsen.

5.3.1. Het zoeken naar adressen waarin een bepaald tekstje voorkomt

Als we nu willen zoeken naar personen in de kaartenbak die het woord `straat` in hun adres hebben staan, dan kunnen we daarvoor een opdracht geven. We kiezen dan uit het menu de letter `z` (van zoeken):

```
Geef de letter van je keuze: z
```

De computer stelt dan nog een paar vragen, en geeft tenslotte het antwoord:

```
Je kunt zoeken op:
a. achternaam
g. gemeente
o. opmerkingen
p. postcode
s. straat & nummer
t. telefoon
v. voornaam
Geef letter van je keuze: s
Gezocht stuk tekst: straat
```

```
--------------------
|   Teresa Baars   |
|  Eykenstraat 23  |
| 1411 TK  Naarden |
--------------------
Telefoon: (02159)74110
Opmerkingen: hobby: borduren en kickboksen
```

In dit geval was er maar één adres dat de gezochte tekst bevatte. Als het er meer zijn verloopt het een beetje anders:

```
Je kunt zoeken op:
a. achternaam
g. gemeente
o. opmerkingen
p. postcode
s. straat & nummer
t. telefoon
v. voornaam
```

```
Geef letter van je keuze: o
Gezocht stuk tekst: yoga
Er zijn 2 personen met "yoga" in hun opmerkingen.
Wil je ze allemaal zien (j/n) ?
```

We kunnen nu, door j te antwoorden, alle adreskaarten te zien krijgen met **yoga**, maar we kunnen ook n zeggen:

```
Wil je ze allemaal zien (j/n) n
Moet ik deze collectie apart bewaren om verder in te zoeken (j/n) j
Naam voor deze collectie: yogaclub
```

We hebben nu de collectie van personen met **yoga** in hun opmerkingen de naam **yogaclub** gegeven. Als we nu eigenlijk geïnteresseerd waren in personen met **yoga** in hun opmerkingen en tegelijk **pen** in hun straat & huisnummer, dan kunnen we die vinden door nu weer te laten zoeken, maar dan alleen in de collectie **yogaclub**. Na het menu kiezen we dan weer **z**:

```
Geef de letter van je keuze: z
Je kunt zoeken op:
a. achternaam
g. gemeente
o. opmerkingen
p. postcode
s. straat & nummer
t. telefoon
v. voornaam
Geef letter van je keuze: s
Gezocht stuk tekst: pen
Alleen een deelcollectie (j/n) j
Naam van de collectie: yogaclub
```

```
----------------------
|     Isabel Vos      |
|    Iepensteeg 19    |
| 3791 VJ  Achterveld |
----------------------
 Telefoon: (03425)9473
Opmerkingen: hobby: archeologie en yoga
```

☞ 5.2 Hoe zou het komen dat het **adres**-softwerkertje ons nu wel vraagt of we alleen in een deelcollectie willen zoeken, terwijl dat de eerste keer niet gebeurde?

5.3.2. Het tonen van de hele kaartenbak of een deelcollectie

Als we alle kaarten uit de hele kaartenbak willen zien, kiezen we **t** (van tonen) uit het menu:

```
Geef de letter van je keuze: t
Een enkel adres (j/n) n
Alleen een deelcollectie (j/n) j
                    --------------------
                    |    Karel Valk     |
                    |   Elzendreef 29   |
                    | 2272 CG  Voorburg |
                    --------------------
                    Telefoon: (070)687913
              Opmerkingen: hobby: yoga en boogschieten

                    ----------------------
                    |     Isabel Vos     |
                    |    Iepensteeg 19   |
                    | 3791 VJ  Achterveld |
                    ----------------------
                     Telefoon: (03425)9473
               Opmerkingen: hobby: archeologie en yoga
```

☞ 5.3 Laat nu alle adressen tonen.

5.3.3. Een nieuw adres toevoegen

Laten we nu eens een nieuw adres aan de kaartenbak toevoegen:

```
Geef de letter van je keuze: n
```

Nu zal de computer om een aantal gegevens vragen:

```
Geef de letter van je keuze: n
achternaam: Spiering
voornaam: Odile
straat & nummer: Appelgaard 2
gemeente: Doorn
postcode: 3941 LZ
telefoon: (03430)15656
opmerkingen: geen
```

Hierna krijgen we vanzelf weer het menu te zien:

```
------------------------------------------------------------------
Wat wil je doen: n. nieuw adres toevoegen
                 s. schrappen van bestaand adres
                 c. corrigeren van bestaand adres
                 t. tonen van een of meer adressen
                 z. zoeken naar adres met bepaald tekstje
                 o. ophouden
Geef de letter van je keuze: ?
```

☞ 5.4 Voeg zelf een nieuwe kaart toe met je eigen naam en adres.

5.3.4. Het corrigeren van een bestaand adres

Als we de opmerkingen bij Odile Spiering willen veranderen, kunnen we van het menu de letter c (van corrigeren) kiezen:

```
Geef de letter van je keuze: c
```

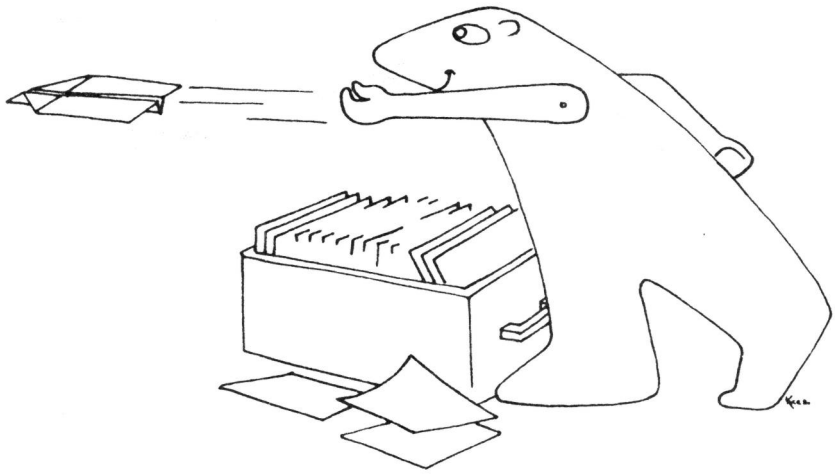
De computer wil nu weten wiens adres we willen verbeteren, en geeft ons daarna de gelegenheid een

of meer van de delen van het adres opnieuw op te geven:

```
Achternaam: Spiering
Geef alleen verbetering waar nodig, anders alleen RETURN:
straat & nummer: Appelgaard 2
verbeterd:
gemeente: Doorn
verbeterd:
postcode: 3941 LZ
verbeterd:
telefoon: (03430)15656
verbeterd:
opmerkingen: geen
verbeterd: hobby zeezeilen en drop
achternaam: Spiering
verbeterd:
voornaam: Odile
verbeterd:
```

☞ 5.5 Verbeter ook fouten die je eventueel gemaakt hebt bij het opgeven van je eigen adres.

5.3.5. Het weggooien van een kaart



Als we de kaart van `Odile Spiering` willen verwijderen uit de kaartenbak, dan kiezen we de `s` (van schrappen) uit het menu:

```
Geef de letter van je keuze: s
Achternaam: Spiering
Odile Spiering geschrapt.
```

5.3.6. Ophouden met werken met de kaartenbak

Om het werken met de kaartenbak te beëindigen, kiezen we de letter `o` (van ophouden) uit het menu:

```
Geef de letter van je keuze: o
>>> ?
```

We zouden nu weer het commando `MENU` kunnen geven; als we dat niet willen, geven we het

commando **QUIT** (Engels voor „ophouden"):

```
>>> QUIT
λ
```

Nu kunnen we weer softwerkertjes oproepen die met een λ beginnen. Als we nu de volgende keer weer het **adres**-softwerkertje oproepen, zullen de adressen nog steeds in de kaartenbak zitten, en kunnen we doorgaan met het toevoegen van nieuwe adressen, enzovoort.

☞ 5.6 Als je in je adressenbestand o.a. de leden van je bloemschik-clubje bijhoudt, en je wilt af en toe een briefje schrijven naar die leden, wat moet je dan doen om alleen die adressen te laten tonen? En als je af en toe ook een lijst van de adressen van je familieleden wilt laten maken?
Is het nu mogelijk een lijst te laten maken van die familieleden die ook nog lid zijn van het bloemschik-clubje?

6. Programma's

6.1. Inleiding

Tot nu toe hebben we alleen reeds bestaande softwerkertjes gebruikt. In dit hoofdstuk zullen we mogelijkheden zien om zelf de werking van een softwerkertje te veranderen.

6.2. Het dobbel-systeem

Het softwerkertje λ `dobbel` is bedoeld voor het werken met dobbelstenen. Als we λ `dobbel` oproepen, antwoordt dat zo:

```
λ dobbel
>>> ?
```

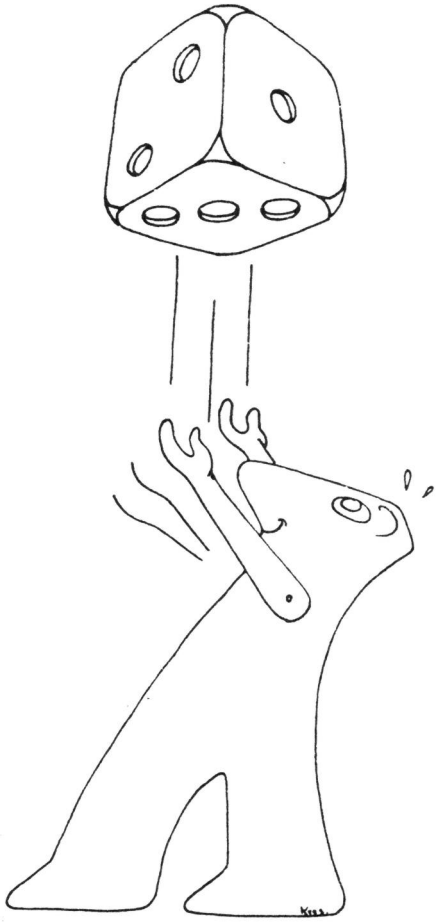
(Dit lijkt precies op λ `adres` uit het vorige hoofdstuk.) Met `>>> ?` wordt aangegeven dat we nu een commando kunnen geven aan het softwerkertje λ `dobbel`.

We kunnen nu het commando `GOOI` geven om 10 keer met een dobbelsteen te laten gooien en de uitkomst steeds op het scherm te laten zien:

```
>>> GOOI
5 2 1 3 4 1 6 3 2 2
>>> ?
```

Als we nog eens het commando `GOOI` geven, zullen we 10 andere uitkomsten te zien krijgen, bijvoorbeeld:

```
>>> GOOI
4 3 3 1 5 1 2 6 5 2
>>> ?
```



6.3. Het programma van een commando

We kunnen ook te weten komen hoe het `dobbel`-softwerkertje het `GOOI`-commando precies uitvoert. We moeten dan eerst een dubbele punt typen, en dan de naam van het commando. We krijgen dan het lijstje commando's te zien die λ `dobbel` in werkelijkheid uitvoert als we het commando `GOOI` geven:

```
>>> :GOOI

HOW'TO GOOI:
    FOR keer IN {1..10}:
        WERP
        SCHRIJF
WERP:
    CHOOSE ogen FROM {1..6}
SCHRIJF:
    WRITE ogen
```

Dit is het *programma*, zeg maar het werkbriefje, dat bij het commando `GOOI` hoort. Op het eerste gezicht is er weinig van te begrijpen. We herkennen eigenlijk alleen de `10` die aangeeft hoeveel worpen er gedaan moeten worden, en de `6` die aangeeft hoeveel kanten een dobbelsteen heeft. We zullen het programma eens regel voor regel bekijken.

a. `HOW'TO GOOI:`

Betekent: de naam van het commando is `GOOI`, en hieronder wordt uitgelegd hoe het commando `GOOI` moet worden uitgevoerd.

b. `FOR keer IN {1..10}:`

Betekent: doe de ingesprongen commando's hieronder (dus `WERP` en `SCHRIJF`) 10 keer.

c. `WERP`

Hoe het commando `WERP` moet worden uitgevoerd staat 2 regels lager.

d. `SCHRIJF`

Hoe het commando `SCHRIJF` moet worden uitgevoerd staat onderaan.
Tot zover het hoofdprogramma. Hierna wordt nog uitgelegd wat er onder `WERP` en `SCHRIJF` verstaan moet worden.

e. `WERP:`

Dit betekent: het uitvoeren van het commando `WERP` komt neer op het uitvoeren van het ingesprongen commando hieronder.

f. `CHOOSE ogen FROM {1..6}`

Betekent: kies een willekeurig getal uit de verzameling gehele getallen `1` t/m `6`, en bewaar het gekozen getal onder de naam `ogen`.

g. `SCHRIJF`

Betekent: het commando `SCHRIJF` bestaat uit het volgende.

h. `WRITE ogen`

Betekent: schrijf het getal dat onder de naam `ogen` bewaard wordt op het scherm.

In dit programma komen twee soorten commando's voor:
a. Commando's met een vaste betekenis: `WRITE`, `CHOOSE` en `FOR`. Deze behoren tot de vaste commando's van de programmeertaal B. We zullen later nog andere vaste commando's leren kennen.
b. Nieuwe, erbij gemaakte commando's: `WERP` en `SCHRIJF`. Zulke namen kunnen vrij gekozen worden. Het commando `DOBBEL` is zelf trouwens ook een nieuw erbij gedefinieerd commando.
We zien dat sommige regels op een dubbele punt eindigen. De volgende regel of regels horen daar dan bij en staan dan ingesprongen (met een [TAB]).

☞ 6.1 Wat zou er in het programma veranderd moeten worden om niet 10 maar 25 worpen met de dobbelsteen te zien te krijgen?

6.4. Het veranderen van het programma van een commando

We kunnen de werking van het `GOOI`-commando veranderen door het programma ervan te veranderen. We hebben gezien dat we het programma te zien kunnen krijgen door `:GOOI` te typen. We kunnen daarna de tekst van het programma veranderen op ongeveer dezelfde manier als met λ `maak`, dus door gebruik te maken van de focus..

☞ 6.2 Roep op de terminal het softwerkertje λ `dobbel` op. Geef daarna het commando `GOOI`. Als je het `GOOI`-commando geprobeerd hebt, kun je de definitie ervan gaan veranderen door `:GOOI` te typen. Verplaats daarna de focus naar de `10` en maak daar `25` van. Gebruik daarvoor de speciale toetsen die beschreven zijn in hoofdstuk 3 (zie samenvatting in 3.5.). Verlaat tenslotte de definitie met [SHIFT][F1]. Geef daarna opnieuw het commando `GOOI`, zodat je de gevolgen van de verandering kunt zien.

Het is mogelijk de definitie van `GOOI` zo te veranderen, dat we na het geven van het `GOOI`-commando, kunnen opgeven hoeveel worpen met de dobbelsteen we willen zien. De definitie moet er dan zo uitzien:

```
HOW'TO GOOI:
    WRITE "Aantal worpen: "
    READ w EG 0
    FOR keer IN {1..w}:
        WERP
        SCHRIJF
WERP:
    CHOOSE ogen FROM {1..6}
SCHRIJF:
    WRITE ogen
```

De eerste twee regels van deze definitie zijn nieuw, de derde is veranderd:

a. **WRITE "Aantal worpen: "**

Betekent: schrijf de tekst **Aantal worpen:** op het scherm. (Dat het om de letterlijke tekst gaat, en niet om een naam waaronder een getal is opgeborgen, is te zien aan de aanhalingstekens.)

b. **READ w EG 0**

Betekent: schrijf een vraagteken op het scherm, wacht totdat de gebruiker een getal heeft ingetoetst (gevolgd door [RETURN]), en bewaar dat getal onder de naam **w**.

c. **FOR keer IN {1..w}:**

Betekent: doe de ingesprongen commando's die volgen (hier dus **WERP** en **SCHRIJF**) even vaak als het getal dat wordt bewaard onder de naam **w**.

We zien dat deze nieuwe versie van het **GOOI**-commando eerst aan de gebruiker vraagt hoeveel keer er gedobbeld moet worden. Dat getal wordt dan opgeborgen onder de naam **w**, en in het **FOR**-commando wordt er dan voor gezorgd dat de ingesprongen regels daarna precies **w** keer worden uitgevoerd. Je zou **w** kunnen vergelijken met een heel kleine file, waarvan **w** de naam is.

☞ 6.3 Breng de veranderingen in de definitie van **GOOI** ook via het toetsenbord aan, en gebruik de nieuwe versie van het commando nu om 32 worpen met de dobbelsteen te laten zien. Bij het toevoegen van het **WRITE**-commando zul je merken dat je na het intoetsen van een **W** of een **w** op het scherm krijgt:

```
W?RITE ?
```

Omdat je inderdaad **WRITE** had willen typen, kun je met [TAB] aangeven dat de suggestie **W?RITE ?** goed was. Er verschijnt dan:

```
WRITE ?
```

en je kunt op de plaats van het vraagteken verder typen. (Als je het niet met zo'n suggestie eens bent, dan typ je gewoon verder wat je had willen typen, i.p.v. op [TAB] te drukken.)

6.5. Nog een verandering: het totaal wordt ook getoond

Hier is een nieuwe versie van het **GOOI**-commando die er ook voor zorgt dat na afloop het totaal van alle worpen wordt getoond:

```
HOW'TO GOOI:
    WRITE "Aantal worpen: "
    READ w EG 0
    PUT 0 IN totaal
    FOR keer IN {1..w}:
        WERP
        SCHRIJF
        TEL
    TOTAAL
WERP:
```

```
    CHOOSE ogen FROM {1..6}
SCHRIJF:
    WRITE ogen
TEL:
    PUT totaal+ogen IN totaal
TOTAAL:
    WRITE / "Totaal", totaal, "gegooid met", w, "dobbelstenen"
```

Nieuw zijn deze regels:

a. `PUT 0 IN totaal`

Betekent: berg het getal 0 op onder de naam `totaal`.

b. `PUT totaal+ogen IN totaal`

Betekent: bereken de som van de getallen die bewaard worden onder de namen **totaal** en **ogen**, en berg de uitkomst op onder de naam **totaal**.

c. `WRITE / "Totaal", totaal, "gegooid met", w, "dobbelstenen"`

Betekent: ga over op een nieuwe regel van het scherm (dat is de betekenis van **/**), schrijf de tekst **Totaal**, schrijf het getal dat bewaard wordt onder de naam **totaal**, schrijf de tekst **gegooid met**, schrijf het getal dat bewaard wordt onder de naam **w**, en schrijf de tekst **dobbelstenen**.

In deze versie wordt dus het totaal aantal gegooide ogen bijgehouden, doordat bij elke worp het aantal ogen wordt opgeteld bij het oude getal dat onder de naam **totaal** werd bewaard. Omdat in het begin het getal 0 onder de naam **totaal** is opgeborgen zal er aan het eind precies de goede uitkomst onder de naam **totaal** staan.

☞ 6.4 Breng de veranderingen ook via het toetsenbord aan, en gebruik het resultaat om de computer 52 keer te laten dobbelen en na afloop het totaal van alle 52 worpen te laten zien. Reken uit hoe groot de uitkomst van een worp gemiddeld was in deze 52 worpen.

☞ 6.5 Wat moet er nog veranderd worden om te laten gooien met een regelmatig twaalfvlak i.p.v. een kubus? Zorg ervoor dat het resultaat er ongeveer zo uitziet:

```
GOOI
Aantal worpen met een regelmatig twaalfvlak: 18
3 10 2 4 3 8 11 5 9 4 4 12 1 5 6 2 10 9
Het totaal van de 18 worpen was 108
```

6.6. Werpen met meer dobbelstenen

Hier is nog een versie van **GOOI** waaraan we kunnen opgeven met hoeveel dobbelstenen elke worp gedaan moet worden:

```
HOW'TO GOOI:
    WRITE "Aantal worpen: "
    READ w EG 0
    WRITE "Aantal dobbelstenen per worp: "
    READ s EG 0
    PUT 0 IN totaal
    FOR keer IN {1..w}:
        WERP
        SCHRIJF
        TEL
    TOTAAL
WERP:
    PUT 0 IN worp
    FOR steen IN {1..s}:
        CHOOSE ogen FROM {1..6}
```

```
        PUT worp+ogen IN worp
SCHRIJF:
    WRITE worp
TEL:
    PUT totaal+worp IN totaal
TOTAAL:
    WRITE / "Totaal", totaal, "gegooid met", w*s, "dobbelstenen"
```

Het enige echt nieuwe in deze versie is het gebruik van de formule `w*s` in de laatste regel. De uitkomst van die formule wordt ber' end door de getallen die bewaard worden onder de namen `w` en `s` met elkaar te vermenigvuldigen. Omdat in `w` het aantal worpen staat dat gedaan moest worden, en in `s` het aantal stenen dat bij elke worp gebruikt moest worden, levert de uitkomst van de formule `w*s` dus precies het totaal aantal dobbelstenen dat heeft meegedaan om het totaal te produceren dat onder de naam `totaal` bewaard wordt. In formules kunnen behalve `*` ook `+`, `-`, en `/` voorkomen.

☞ 6.6 Breng de veranderingen ook via het toetsenbord aan, en gebruik het resultaat om 10 worpen te laten doen met 3 dobbelstenen.

☞ 6.7 Is de laatste versie van het `GOOI`-commando ook te gebruiken om elke worp weer gewoon met 1 dobbelsteen te laten uitvoeren?

☞ 6.8 Wat moet er aan het programma worden toegevoegd om ook nog de gemiddelde uitkomst per worp op een aparte regel te zien te krijgen?

Als we het stukje programma vanaf `TOTAAL:` er zo laten uitzien, dan wordt aan het eind nog gekeken of het totaal aantal gegooide ogen aan de lage kant of aan de hoge kant was, of misschien precies 3.5:

```
TOTAAL:
    WRITE / "Totaal", totaal, "gegooid met", w*s, "dobbelstenen"
    SELECT:
        totaal < 3.5*w*s:
            WRITE / "Dat is aan de lage kant."
        totaal = 3.5*w*s:
            WRITE / "Dat is gemiddeld precies 3.5!"
        ELSE:
            WRITE / "Dat is aan de hoge kant."
```

Hierin wordt een `SELECT`-commando gebruikt. De betekenis daarvan is: kijk of aan de eerste ingesprongen voorwaarde voldaan is (hier: `totaal < 3.5*w*s`), en zo ja, voer de daarop volgende ingesprongen commando's uit (hier maar een); zo nee, probeer dan de tweede voorwaarde, enz. totdat er een voorwaarde blijkt te kloppen; is er geen kloppende voorwaarde bij, doe dan de commando's na `ELSE`.

6.7. Geheimschrift

We gaan hier het programma bekijken voor het commando `CODEER`, dat bedoeld is om van een regel tekst geheimschrift te maken. Dat gebeurt door voor elke letter van de oorspronkelijke regel een andere letter op het scherm te schrijven, namelijk de letter die 13 plaatsen verder in het alfabet voorkomt. Dus een `a` wordt een `n`, een `b` wordt een `o` enz., en een `m` wordt een `z`. Dan is er een moeilijkheid, omdat de `z` de laatste letter van het alfabet is, maar het gaat gewoon bij het begin van

het alfabet door; dus een n wordt een a, een o wordt een b, enz.

☞ 6.9 Roep het softwerkertje λ code op, en geef daarna het commando CODEER. Er wordt dan een regel tekst gevraagd. Toets dan bijvoorbeeld je naam in om te zien hoe die er in dit geheimschrift uitziet.

Het programma van het CODEER-commando ziet er zo uit:

```
HOW'TO CODEER:
    WRITE "Oorspronkelijke regel tekst:" /
    READ tekst RAW
    FOR symbool IN tekst:
        SELECT:
            symbool in {"a".."z"}:
                KLEIN
            symbool in {"A".."Z"}:
                GROOT
            ELSE:
                WRITE symbool
KLEIN:
    PUT #{"a"..symbool} IN rangnr
    TEL'AF
    WRITE rangnr th'of {"a".."z"}
GROOT:
    PUT #{"A"..symbool} IN rangnr
    TEL'AF
    WRITE rangnr th'of {"A".."Z"}
TEL'AF:
    PUT rangnr+13 IN rangnr
    IF rangnr > 26:
        PUT rangnr-26 IN rangnr
```

Bij sommige regels is wat uitleg nodig:

a. **READ tekst RAW**

Betekent: schrijf een vraagteken op het scherm, wacht totdat de gebruiker een regel tekst heeft ingetoetst (gevolgd door [RETURN]), en bewaar die regel onder de naam **tekst**.

b. **FOR symbool IN tekst:**

Betekent: doe de ingesprongen regels die volgen even vaak als het aantal symbolen (inclusief spaties, cijfers enz.) opgeborgen onder de naam **tekst**, en wel met de eerste keer het eerste symbool van **tekst** onder de naam **symbool**, de tweede keer met het tweede symbool van **tekst** onder de naam **symbool**, enz.

c. **symbool in {"a".."z"}:**

Betekent: doe het ingesprongen deel dat nu volgt alleen als het symbool dat bewaard wordt onder de naam **symbool** voorkomt in de rij **a** t/m **z**.

d. **PUT #{"a"..symbool} IN rangnr** Betekent: ga na hoe lang het rijtje letters is dat begint met **a** en eindigt met de letter die is opgeborgen onder de naam **symbool**, en bewaar het getal dat daar uitkomt onder de naam **rangnr**.

e. **WRITE rangnr th'of {"a".."z"}**

Betekent: schrijf de zoveelste letter van de rij **a** t/m **z** als wordt aangegeven door het getal onder de naam **rangnr**.

f. **IF rangnr > 26:**

Betekent: voer het ingesprongen commando dat volgt alleen uit als het getal dat bewaard wordt onder de naam `rangnr` groter is dan `26`.

We zien dus dat van elk symbool van de oorspronkelijke regel tekst eerst wordt gekeken of het een kleine letter is, of een hoofdletter, of iets anders. Is het een kleine letter (of een hoofdletter), dan wordt vastgesteld om de hoeveelste van het alfabet het gaat (`rangnr`), wordt daar `13` bij opgeteld en, als de uitkomst te groot is, weer `26` van afgetrokken; het resterende rangnummer geeft dan aan welke letter van het alfabet op het scherm geschreven moet worden.
Is het oorspronkelijke symbool geen letter, maar bijvoorbeeld een spatie of een komma, dan wordt die onveranderd op het scherm geschreven.

☞ 6.10 Als je een regel tekst met dit commando gecodeerd hebt, en je hebt het geheimschrift dat er uitkwam bewaard, maar de oorspronkelijke tekst vergeten, hoe zou je die oorspronkelijk tekst dan weer te weten kunnen komen?

☞ 6.11 Je kunt dit programma natuurlijk gemakkelijk zo veranderen dat er andere geheimschriften worden gemaakt: je hoeft alleen maar de `13` in een ander getal te veranderen.

☞ 6.12 Je kunt het programma ook zo maken dat het eerst vraagt over hoeveel plaatsen er „geschoven" moet worden, i.p.v. dat er altijd over 13 plaatsen geschoven wordt.

☞ 6.13 Wat zou er veranderd moeten worden om niet te schuiven, maar te „spiegelen", d.w.z. om van een `a` een `z` te maken, en van een `b` een `y`, van een `c` een `x`, enz.? Oftewel: van letter nummer 1 moet letter nummer 26 gemaakt worden, van nummer 2 moet nummer 25 gemaakt worden, van nummer 3 moet nummer 24 gemaakt worden, enz.
Een tip: er hoeft maar heel weinig te veranderen; als het rangnummer van de oorspronkelijke letter `r` is, dan is het rangnummer van de letter die op het scherm moet komen: `27-r`.

6.8. Samenvatting

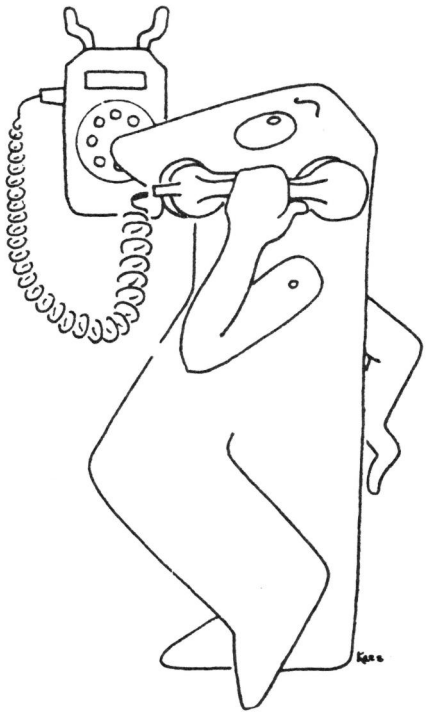
Commando's

`WRITE` *formule*	reken de *formule* uit en schrijf de uitkomst op het scherm
`PUT` *formule* `IN` *naam*	bewaar uitkomst van de *formule* onder *naam*
`READ` *naam* `EG 0`	schrijf `?` en bewaar het ingetypte getal onder *naam*
`READ` *naam* `RAW`	schrijf `?` en bewaar de ingetypte tekst onder *naam*
`CHOOSE` *naam* `FROM` *serie*	kies zomaar een element van de *serie* en bewaar dat onder *naam*
`FOR` *naam* `IN` *serie*`:`	voer ingesprongen commando's een keer uit voor elk element van de *serie*, met steeds het volgende element van de *serie* onder *naam* opgeborgen
`IF` *voorwaarde*`:`	voer ingesprongen commando's alleen uit als de *voorwaarde* klopt
`SELECT:`	voer ingesprongen commando's uit die volgen op de eerste ingesprongen voorwaarde die klopt, anders de ingesprongen commando's na `ELSE`
`HOW'TO` *NAAM*`:`	hier volgt programma voor commando *NAAM*

Formules

getal	
naam	levert datgene wat het laatst onder *naam* is opgeborgen
formule `*` *formule*	levert het produkt van de getallen die uit de twee *formules* komen (net zo met `+`, `-` en `/`)
"serie symbolen"	levert de tekst die bestaat uit de *serie symbolen*
`{"`*letter*`"..."`*letter*`"}`	levert de alfabetische serie letters van de ene *letter* t/m de andere
getalsformule `th'of` *serie*	levert het element van de *serie* waarvan de uitkomst van de *getalsformule* het rangnummer is
`#`*serie*	levert het getal dat de lengte van de *serie* aangeeft

7. In gesprek met de computer



7.1. Inleiding

We hebben de computer inmiddels gezien als een machine, die voor verschillende zaken gebruikt kan worden. In hoofdstuk 2 funktioneerde hij als hulpmiddel voor de redakteur en in hoofdstuk 5 bleek dat de administrateur deze machine ook kon gebruiken. In dit hoofdstuk lijkt de computer steeds minder op een machine, doordat hij steeds meer menselijke eigenschappen lijkt te krijgen. We willen nagaan of dat inderdaad zo is.

7.2. Een gesprek met de computer

Vanuit je eigen werkgebied kun je de softwerker

```
λ prater
```

oproepen. Er verschijnt dan op je scherm:

```
TELL ME ABOUT YOUR PROBLEM.
TYPE q TO EXIT.
```

Je wordt nu blijkbaar uitgenodigd om iets (willekeurigs) in te toetsen. Dat doe je dan:

```
ik vind jou niet sympathiek
```

Zijn reaktie:

```
LET'S CHANGE THE SUBJECT
```

Jij:

```
I am too busy today
```

Prater:

```
WHY ARE YOU TOO BUSY TODAY ?
```

Jij:

```
My sister is ill
```

Prater:

```
DO YOU HAVE ANY SISTERS ?
```

Jij:

```
Vier
```

Prater:

```
WOULD YOU LIKE TO MEET MY SISTER? SHE IS VERY SEXY
```

Jij:

```
Tell me about her
```

Prater:

```
NO, YOU TELL ME MORE ABOUT HER
```

Jij:

```
No, I am not a doctor
```

Prater:

```
HOW DO YOU FEEL ABOUT DOCTORS ?
```

Jij:

```
It makes no difference. My sister needs a doctor, for she is very in pain.
```

Prater:

```
ARE YOU OFTEN IN PAIN ?
```

Jij:

```
Je zoekt het maar uit
```

Prater:

```
TELL ME ABOUT YOUR PAINS
```

Jij:

Prater:

```
PLEASE GO ON
```

Jij:

```
q
```

Prater:

```
I HOPE MY TREATMENT HAS MADE YOU FEEL BETTER.
```

Tot zover het gesprek! Iedere keer als je iets intypt, krijg je weer antwoord.

☞ 7.1 Roep λ `prater` op en ga na of het bovenstaande gesprek inderdaad gevoerd kan worden.

7.3. De antwoord-file.

Het lijkt erop dat λ `prater` bij het formuleren van zijn reaktie bepaalde regels hanteert. Dat is inderdaad het geval. Als jij iets intoetst, gaat λ `prater` met jouw zin naar een speciale antwoord-file. In deze file, waarin dus allemaal reakties staan opgenomen, kiest hij volgens bepaalde regels een reaktie en zet die op het scherm. We gaan eerst eens kijken wat er in deze file is opgenomen:

```
TELL ME ABOUT YOUR PROBLEM. TYPE q TO EXIT.
I HOPE MY TREATMENT HAS MADE YOU FEEL BETTER.
all men are %
* many women are also %
* men are not sheep.  each one is an individual

all women are %
* there are men who are % too
* my sister is not
* no, some are not

do you %
* what difference does it make whether or not i % ?
* maybe i % and maybe i don't
* yes
* yes
```

```
i am %
* why are you % ?
* does your mother know you are % ?
* how do you know you are % ?
* i don't think you are %
* i am % sometimes too

my % hurts
* if my diagnosis is correct, we will have to amputate your %
* i am a shrink, not a real doctor
* take two aspirins and go to bed

tell me %
* no, you tell me %
* quiet. from now on, i will ask the questions, not you
* no. i refuse to tell you %

you are %
* maybe i am %
* perhaps i only pretend i am %
* how do you know i am % ?

+
xxxxxxx 5
* let's change the subject
* go on
* is there anything else ?
* what makes you happy ?

advice 3
* why do you need advice ?
* why not ask your family for advice ?

certainly 2
* how can you be so sure
* i wouldn't be so sure if i were you

doctor 5
* how do you feel about doctors ?
* what do you think of my treatment?

hollands 5
* i don't speak dutch, just english

pain 6
* are you often in pain ?
* tell me about your pains
* what causes you pain ?

sister 5
* do you have any sisters ?
* would you like to meet my sister? she is very sexy
```

Weliswaar staat hier slechts een klein gedeelte van de file, maar toch kun je een goede indruk krijgen van de wijze waarop deze 'antwoord-file' is opgebouwd. Voor het kiezen van zijn reaktie neemt

λ `prater` de volgende regels in acht:

1. Als eerste zin neemt λ `prater` de bovenste regel uit de file.
2. Als er alleen maar "`q`" wordt ingetypt, spreekt λ `prater` als slotzin de tweede regel uit de file.
3. Eerst probeert λ `prater` of de ingetypte zin overeenkomt met een zin uit het eerste gedeelte van de file, waarbij het `%`-teken een willekeurig zinsdeel vervangt. Als dat lukt wordt de reaktie *een* van de zinnen eronder, die met een `*` begint. Voor het `%`-teken wordt het vervangen zinsdeel genomen.
4. Als 3 niet lukt, probeert λ `prater` of er in de ingetypte zin een woord uit het tweede gedeelte van de file staat. Zo ja, dan geeft hij als reaktie *een* van de zinnen eronder, die met een `*` beginnen.
5. Als 3 en 4 niet lukken geeft λ `prater` als reaktie een van de `*`-zinnen onder het laatst herkende woord aan.
6. Als 5 niet lukt geeft hij een reaktie die onder net woord `xxxxxxx` staat.
7. Als je alleen maar een RETURN intoetst, komt er als reaktie:

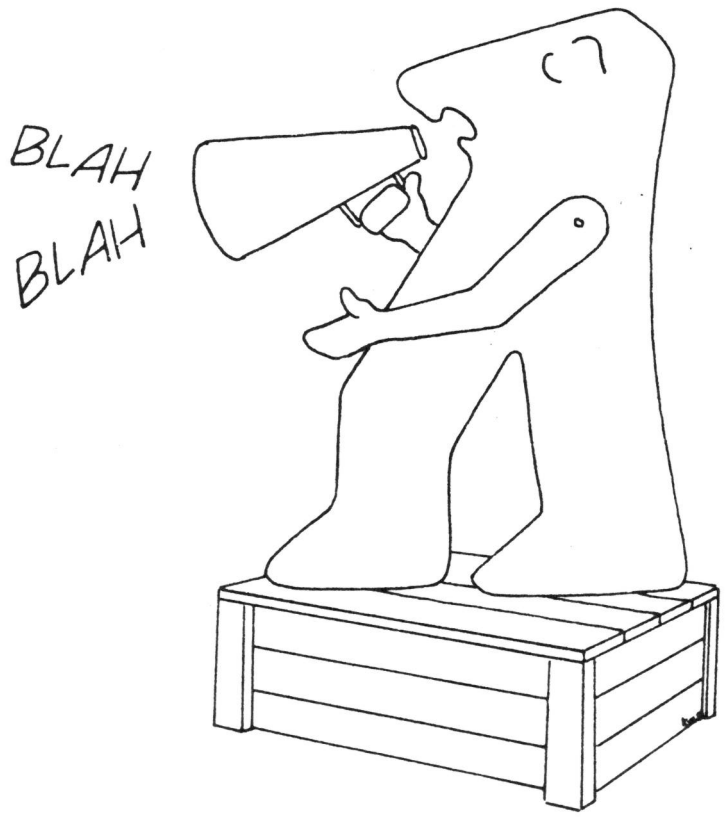
   `PLEASE GO ON`

8. Als er meer woorden in een getypte zin worden herkend, kiest λ `prater` zijn reaktie onder het woord met de hoogste prioriteit (het grootste getal). Als er meer woorden zijn met de hoogste prioriteit, wordt het eerste genomen.
9. Als λ `prater` bij de voortgang van het "gesprek" weer eenzelfde zinsdeel of eenzelfde woord herkent, geeft hij de opvolgende zin uit de met een `*` aangegeven reaktie.

☞ 7.2 Ga na of het "gesprek" uit paragraaf 7.2 volgens de bovenstaande regels verloopt.

7.4. Een andere "prater"

Zoals je hebt gezien, hangt de reaktie van λ `prater` in sterke mate af van de inhoud van de antwoord-file. Als je in plaats van deze file een andere maakt, zou je dus een eigen "prater" kunnen maken.



In de volgende regels staat precies aangegeven hoe een prater-file eruit moet zien.

1. De file begint met 2 regels.
2. De rest van de file is in 2 gedeelten verdeeld, gescheiden door een **+**.
3. De file eindigt met een **+**.
4. Het eerste deel van een file bestaat uit gedeelten van zinnen en een **%**; zo'n zinsdeel begint nooit met een **%**.
5. Onder de zinsdelen uit het eerste gedeelte van de file staat een aantal (een of meer) zinnen, die allemaal met een * beginnen. In deze zinnen kan (hoeft niet) het **%**-teken worden opgenomen.
6. Het tweede gedeelte van de file bestaat uit woorden gevolgd door een spatie en een getal (tussen 1 en 9).
7. Het eerste "woord" is **xxxxxxx 5**.
8. Onder ieder woord in het tweede gedeelte van de file staat een aantal (een of meer) zinnen, die met een * beginnen.

☞ 7.3 Maak in groepsverband een eigen file volgens bovenstaande voorschriften. Als je deze file b.v. **ABC** noemt, dan kun je daarna

```
λ prater ABC
```

intoetsen, om een gesprek te voeren.