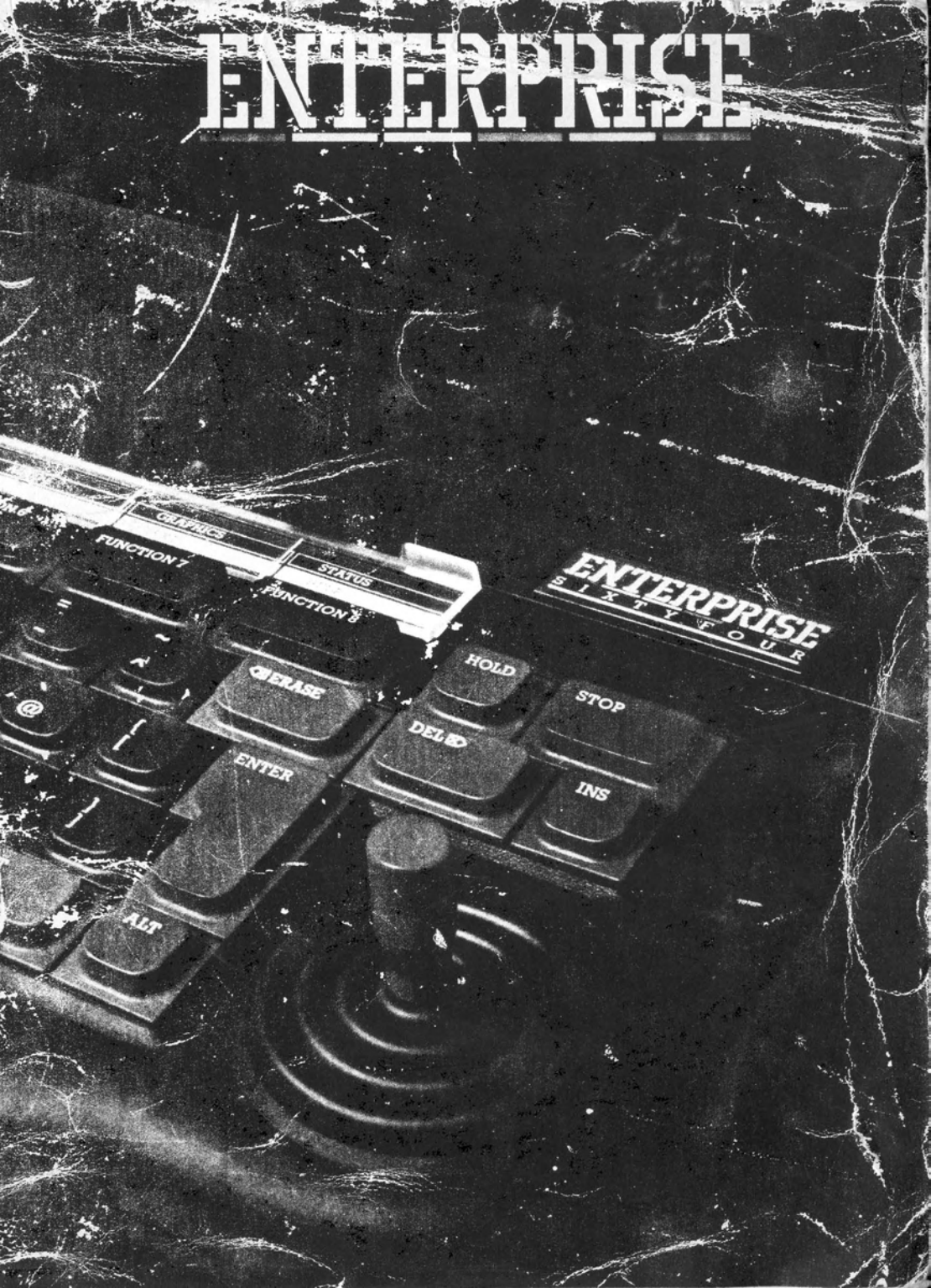

ENTERPRISE
GRAPHICS
FUNCTION 7
STATUS
FUNCTION 8
ENTERPRISE
SIXTY FOUR
HOLD
ERASE
STOP
DEL
ENTER
INS
ALT

INHOUD

KENNISMAKING 3

INLEIDING 4
DE EERSTE PROGRAMMA'S 5
PROGRAMMEREN 17
IN JUISTE VOLGORDE TE WERK GAAN 19
DE RECHTSTREEKSE MODUS EN SLEUTELWOORDEN 21
VARIABELEN 24
BEWERKINGSTEKENS EN UITDRUKKINGEN 31
DE OPMAAK VAN TEKST 36
DE OPMAAK VAN PROGRAMMA'S 40
TEKSTVERWERKING 46
DE FUNCTIETOETSEN 52
WAT BEWERKT ELKE AFZONDERLIJKE FUNCTIETOETS? 54
PROGRAMMA'S OP CASSETTE 56

WETENSWAARDIGHEDEN 61

STRINGS 62
LUSSEN 70
BESLISSINGEN 76
DE OPSLAG VAN GROTERE HOEVEELHEDEN INFORMATIE 85
FUNCTIES DEFINIËREN 93
GRAFISCHE GEGEVENSVERWERKING 104
DE TEKENSET 121
KLANK EN RITME 125
PROBLEMEN IN EEN PROGRAMMA OMZETTEN 134
DE KARAKTERISTIEKEN VAN 'MINIMAL BASIC' 145
KANALEN 151
HOE OM TE GAAN MET UITZONDERINGEN 153
HET NETWERK 156
TOEPASSING VAN MACHINECODE 161

NASLAGGEDEELTE 167

DE REGELS VAN HET BASIC 168
COMMANDO'S EN OPDRACHTEN 172
MACHINE-OPTIES (ALGEMEEN) 211
VIDEO-OPTIES 217
GELUIDS-OPTIES 223
INGEBOUWDE FUNCTIES EN VARIABELEN 225
EXOS 232
FOUTBOODSCHAPPEN 235
WOORDENLIJST 242
INDEX 258

Om te beginnen gaan we met de machine werken. Op een dergelijke manier dat u ermee vertrouwd kunt raken en inzicht verkrijgt in de mogelijkheden ervan. Technische aspecten komen later aan de orde.

U doet er het verstandigst aan de volgorde van dit gedeelte van de handleiding aan te houden, immers deze is zo samengesteld dat u gemakkelijk met uw nieuwe computer vertrouwd kunt raken. Waar de mogelijkheid bestaat om het ene onderwerp af te ronden voordat u met een nieuw begint worden verwijzingen gegeven, zodat u naar eigen inzicht en in uw eigen tempo de machine kunt leren kennen. In het tweede gedeelte van deze handleiding komen alle 'onderdelen' van het programmeren gedetailleerd aan de orde (naar u zult zien staan deze in werkelijkheid allen met elkaar in verband), alleen op een iets hoger niveau dan in het eerste gedeelte. Wanneer u eenmaal in essentie de computer kunt besturen, zal het laatste gedeelte ertoe bijdragen dat u de Enterprise nog beter leert kennen.

Voordat u een aantal programma's gaat bekijken is het raadzaam eerst een beetje met het toetsenbord te experimenteren. Wat u maar wilt kunt u intikken, het zal de computer absoluut geen kwaad doen. Wanneer de computer weigert letters weer te geven druk dan op de 'reset' (opnieuw instellen) knop aan de achterkant. De joystick (spelpookje) is uitermate geschikt om van alles en nog wat uit te proberen — verderop zult u zien hoe nuttig dit apparaatje is. Stel uzelf in de tussentijd in de gelegenheid de computer te leren kennen. Hij heeft u heel wat te bieden.

Let wel: wanneer u de computer alleen maar als tekstverwerker gebruikt (zie pp. 46-51 en de beide tabellen op pp. 44-45), hoeft u de IS-BASIC cartridge niet in te voeren. Maar voor het schrijven van BASIC programma's moet u deze cartridge in de ROM BAY steken aan de linkerkant van de machine.

Probeer onderstaande gegevens in te tikken. Als het om fouten gaat, al is het maar de kleinste fout, doen computers een beetje raar; controleer daarom na het tikken alles dat u heeft ingetikt. Onthoud dat u aan het einde van elke regel de 'enter' toets moet indrukken. Vergeet niet het regelnummer aan het begin van iedere regel, dat is eveneens van belang. Maar u hoeft zich geen zorgen te maken over de spaties achter de nummers. De Enterprise is in staat deze spaties automatisch in te voegen, zodat het programma er wat netter uitziet. Merk op dat computers een speciaal symbool voor nul gebruiken: Ø, dit om de nul van de hoofdletter O te onderscheiden.

```
100     GRAPHICS
110     PLOT 640, 360,
120     !
130     DO
140         FOR RADIUS=250 TO 1 STEP -16
150             SET INK RND(3)+1
160             PLOT ELLIPSE RADIUS, RADIUS,
170             PLOT PAINT
180         NEXT RADIUS
190         PING
200     LOOP
210     !
220     END
```

FOUTEN CORRIGEREN

Heeft u iets verkeerd ingetikt, dan is het heel eenvoudig dit te corrigeren. Bevindt u zich nog op de regel waar de fout is gemaakt, druk dan op de 'erase' (uitwissen) toets waardoor de rode knipperende 'cursor' (positiewijzer) naar links gaat en de letters uitgewist worden die hij op zijn weg tegenkomt. Wanneer u naar een vorige regel terug moet, moet u gebruik maken van de joy-stick om de cursor aan het eind van die regel te plaatsen; wis daarna alles tot en met de fout uit. Nu kunt u de correctie intikken, de regel afmaken door op 'enter' te drukken en de cursor weer naar de onderkant van het scherm -of waar dan ook- brengen (door gebruik te maken van de joystick).

Onthoud dit: telkens wanneer u een toets indrukt zal de letter of het getal daar verschijnen waar de cursor zich op dat moment bevindt.

In een later stadium, onder het kopje 'De opmaak

van programma's (pagina 40), zullen we ingaan op veelzijdiger manieren om veranderingen aan te brengen.

HET PROGRAMMA LATEN FUNCTIONEREN

Wanneer u het programma heeft ingetikt moet u het woord RUN intikken en daarna weer op 'enter' drukken. In plaats hiervan kunt u ook 'functietoets 1' indrukken die zich boven de toetsen met cijfers bevindt. Als u deze toets gebruikt hoeft u niet op 'enter' te drukken; bovendien gaat dit sneller dan wanneer u RUN intikt (op pagina 52 vindt u meer over de 'functie'toetsen). Kijk vervolgens een tijdje toe. Heeft u een fout gemaakt, dan zal de computer 'Not understood' op het beeldscherm laten verschijnen. Wanneer dit gebeurt moet u zich absoluut niet druk maken. Breng de correctie aan en probeer het nog eens.

HET INTIKKEN VAN COMMANDO'S

De commando's die u in deze handleiding terugvindt voor invoering in de computer zullen hier altijd in hoofdletters weergegeven worden.

Dit doen we hoofdzakelijk om de commando's in het oog te laten springen maar gedeeltelijk ook omdat de computer zelf commando's in BASIC met hoofdletters weergeeft. Maar u kunt worden als RUN (en andere BASIC-commando's) ook klein geschreven *intikken*; de computer zal u zonder meer begrijpen. U hoeft zich geen moeilijkheden op de hals te halen door voortdurend de 'shift'toets (verschuivingstoets) in te drukken om in plaats van kleine letters hoofdletters te krijgen.

Elk woord dat u niet begrijpt kunt u terugvinden in de Verklarende Woordenlijst, pagina 242-256.

Wilt u het programma stopzetten, druk dan 'stop' in. U krijgt nu het volgende antwoord op het scherm:

```
STOP AT LINE — — —          (— — — is een getal)
ok
—
```

WAT IS EEN PROGRAMMA?

Een programma bestaat uit een reeks instructies die de computer laat weten wat er gedaan moet worden. Dit is nu eenmaal zo, want een computer kan volstrekt niets doen zonder een programma. Programma's zijn uiterst precies en zeer gedetailleerd, maar dat geldt ook voor

borduurwerk. En evenals ieder ander tijdverdrijf dat bedrevenheid vergt, kan het programmeren serieus aangepakt of alleen maar voor de lol gedaan worden.

Indien gewenst, kan het programma opnieuw gestart worden door CONTINUE (ga verder) in te tikken en 'enter' in te drukken. Wat zelfs nog beter is, is dat u de 'shift'toets indrukt en tegelijkertijd functietoets 1 (deze toets heeft u gebruikt om het programma te laten lopen (RUN)). Het programma zal dan hervat worden vanaf het punt waar u het stopzette.

Begint u het vervelend te vinden, type dan maar de volgende regel:

```
135  SET PALETTE RND (256),
     RND (256), RND (256), RND (256)
```

WISSEN VAN HET BEELDSCHERM

Wanneer u dit programma uitgevoerd en stopgezet hebt blijft u met een plaatje zitten dat het beeldscherm opvult. Hierdoor wordt het voor u niet erg gemakkelijk om te lezen wat u intikt. Om het beeldscherm vrij te krijgen moet u TEXT intikken en de 'enter'toets indrukken. Hierdoor zal het scherm weer volledig beschikbaar komen voor de weergave van tekst. Ook hier kunt u tijd (en energie) sparen door functietoets 5 in te drukken.

HET UITLIJSTEN

Wanneer u de nieuwe regel heeft ingetikt moet u deze keer LIST (uitlijsten) intikken. *Functietoets 2 zal hetzelfde bewerkstelligen.* Hier aangekomen moet u in staat zijn te onthouden telkens 'enter' in te drukken wanneer u een instructie of een nieuwe programmaregel heeft ingetikt. 'Enter' is de toets die de computer opdraagt uit te voeren wat u heeft ingetikt. Doorgaans zal er niets gebeuren zolang u deze toets nog niet heeft ingedrukt; maar de 'enter'toets hoeft u niet in te drukken wanneer u de functietoetsen gebruikt — hier zult u waarschijnlijk wel al achter zijn gekomen.

LIST is een woord waardoor het hele programma op het scherm verschijnt (of zoveel van het programma als het beeldscherm in een keer kan bevatten). Nu kunt u zien dat de nieuwe regel in het programma is ingepast. Het programma is nu numeriek geordend, in deze volgorde zal de computer uw instructies uitvoeren wanneer het programma begint te lopen.

HOE KOMEN WE VAN EEN PROGRAMMA AF

Hebt u genoeg van een programma, tik dan gewoonweg NEW (nieuw)in, sla 'enter' aan en het programma is verdwenen.

Met deze computer kunt u heel gemakkelijk tal van geluiden en vele kleuren voortbrengen. Maar hij is niet alleen geschapen om geluiden voort te brengen en regenbogen weer te geven. Uw computer kan ook uiterst verfijnde tekeningen maken, beslissingen nemen, en in een zeer hoog tempo ingewikkelde berekeningen uitvoeren; dit allemaal in elke gewenste volgorde en zo vaak als u wilt.

OEFENINGEN

De programma's op de volgende bladzijden zijn eenvoudige voorbeelden van de mogelijkheden waarover de Enterprise beschikt.

Probeer ze alle! Deze programma's laten slechts een paar dingen zien die u met deze machine kunt doen. Dit boek zal u in staat stellen alle programma's voor uzelf te proberen — en nog veel meer. Vergeet niet TEXT in te tikken wanneer u het hele scherm ter beschikking wilt hebben. U kunt ook gebruik maken van CLEAR SCREEN (wis het scherm) om het beeldscherm schoon te vegen wanneer het vol raakt. Bedenk dat in feite geen van deze commando's ook maar een enkele programmaregel uit het geheugen van de computer verwijdert; u kunt het volledige programma altijd weer onder ogen krijgen door LIST in te tikken.

```
100   PROGRAM "Vuurtunnel"
110   !
120   !     dit programma tekent een
130   !     kleurrijke tunnel met
140   !     exploderende vuurballen
150   !
160   GRAPHICS HIRES 256
170   LET X=640:LET Y=360
180   FOR R=1 TO 255
190         SET INK R
200         LET A=X-R-220:LET A1=Y-R-50
210         LET C=X+R+220:LET C1=Y+R+50
220         PLOT A,A1;A,C1;C,C1;C,A1;A,A1
230         PRINT R
240   NEXT
250   FOR BAL=1 TO 100
260         CALL VUURBAL (256,X,Y)
270   NEXT
280   !
290   END
300   !
310   !
320   DEF VUURBAL (KLEUREN,A,B)
330         SET LINE MODE 3
340         SET INK RND(KLEUREN)
350         FOR GA=1 TO 2
360               FOR ROND=1 TO 650 STEP 30
370                  PLOT A,B ELLIPSE ROND,ROND
380               NEXT
390         NEXT
400         SET LINE MODE 0
410   END DEF
```

```
100   PROGRAM "Rechthoeken"
110   !
120   !     dit programma tekent rechthoeken
130   !     100-140 zijn commentaarregels.
140   !     Ze kunnen weggelaten worden.
150   !
160   CLEAR SCREEN
170   PRINT AT 5,11: "DIT PROGRAMMA TEKENT"
180   PRINT AT 6,11: "RECHTHOEKEN VOOR U"
190   PRINT AT 8,1: "Het programma zal u vragen
      enkele getalparen in te tikken."
200   PRINT AT 11,1: "Het eerste getal van elk paar
      mag niet"
210   PRINT AT 12,1: "groter zijn dan 1279 en het
      tweede"
220   PRINT AT 13,1: "getal mag niet groter zijn dan
      719"
230   PRINT AT 16,1: "Sla 'ENTER' aan nadat u elk
      getal heeft getikt."
240   PRINT AT 24,1: "Druk een toets voor
      vervolg ...";
250   DO
260   LOOP UNTIL INKEY$< >" "
265   !
266   !           regel 250-260 wachten tot er een
267   !           toets wordt ingedrukt.
268   !
270   DO
280         CLEAR SCREEN
290         INPUT AT 5,1, PROMPT "Getallen voor
            een van de hoeken :":X
300         IF X>1279 THEN 290
310         INPUT AT 6,33 PROMPT " ":Y
320         IF Y>719 THEN 310
330         INPUT AT 9,1, PROMPT "en voor de
            tegenoverliggende hoek: ":V
340         IF V>1279 THEN 330
350         INPUT AT 10,35 PROMPT " ":W
360         IF W>719 THEN 350
370         PRINT AT 13,6: "Hoeveel seconden moet
            de"
380         PRINT AT 14,5: "rechthoek in beeld
            blijven:"
```

```
390         INPUT AT 16,5, PROMPT "Seconden:
            ":TIJD
395         !
400         !     de regels 450-480 zijn de instructies
410         !     voor het tekenen van de rechthoek
420         !     en voor de gewenste tijdsduur dat
425         !     deze in beeld moet blijven.
426         !
430         GRAPHICS
440         PLOT X,Y;X,W;V,W;V,Y;X,Y
450         WAIT TIJD
460         TEXT
470         PRINT AT 15,18: "meer?"
480         DO
490           INPUT AT 17,17,PROMPT "j of n:
              ":ANTW$
500         LOOP WHILE ANTW$< >"j"AND
            ANTW$< >"n"
510     LOOP WHILE ANTW$="j"
520     END ! einde programma.
```

```
100   PROGRAM "Sorteren"
110   !  ---------------------------------------
120   !     Dit programma ordent 10 getallen
130   !     in numerieke volgorde.
140   !  ---------------------------------------
150   NUMERIC REEKS (1 TO 10)
160   NUMERIC VAR,NUM,GROOT
170   CLEAR SCREEN
180   PRINT AT 10,10:"Getallen ordenen"
190   FOR N=1 TO 10
200         PRINT AT 14,10:"Getal";N
210         INPUT PROMPT ":":REEKS(N)
220         PRINT AT 14,20:"                    "
225   CLEAR SCREEN
230   NEXT N
240   CLEAR SCREEN
250   PRINT AT 20,12:"Bezig ordenen..."
260   LET LSTE=10
270   FOR X=1 TO 10
280         LET GROOT=0
290         FOR Y=1 TO LSTE
300               IF REEKS(Y)>GROOT THEN LET
                  GROOT=REEKS(Y)
310               IF REEKS(Y)=GROOT THEN LET
                  NUM=Y
320         NEXT
330         LET VAR=REEKS(LSTE)
340         LET REEKS(LSTE)=GROOT
350         LET REEKS(NUM)=VAR
360         LET LSTE=LSTE-1
370   NEXT
380   CLEAR SCREEN
390   PRINT AT 2,16:"Resultaat"
400   PRINT
410   FOR X=1 TO 10
420         PRINT USING 440:X,REEKS(X)
430   NEXT
440   IMAGE:ff#e          ffffffffff#
450   END
```

```
100   PROGRAM "Cirkels"
110   !
120   !     Dit programma geeft de omtrek
130   !     of de oppervakte van cirkels.
140   !
150   LET A$="van de cirkel is: "
160   LET B$="Straal van de cirkel: "
170   NUMERIC STRAAL,OPPVL,OMTREK
180   DO
190         CLEAR SCREEN
200         PRINT AT 7,13: "CIRKELS"
210         PRINT AT 10,10:"1) OPPERVLAKTE"
220         PRINT AT 11,10:"2) OMTREK"
230         PRINT AT 12,10:"3) EINDE"
240         DO
250               INPUT AT 15,10,PROMPT "uw
                  keuze :":NUM
260         LOOP WHILE NUM<1 OR NUM>3 OR
            NUM< >INT (NUM)
270         CLEAR SCREEN
280         IF NUM=1 THEN
290               INPUT AT 10,1,PROMPT
                  B$:STRAAL
300               LET OPPVL=PI*STRAAL^2
310               PRINT AT 15,1:"De oppervlakte
                  ";A$;
320               PRINT AT 17,10:OPPVL
330               WAIT DELAY 5
340               PING
350         ELSE IF NUM=2 THEN
360               INPUT AT 10,1,PROMPT B$:STRAAL
370               LET OMTREK=2*PI*STRAAL
380               PRINT AT 15,1:"De omtrek ";A$;
390               PRINT AT 17,10:OMTREK
400               WAIT DELAY 5
410               PING
420         END IF
430   LOOP WHILE NUM< >3
440   END
```

```
100     PROGRAM "Boodschappen"
105     !
110     !     Dit programma drukt boodschappen af.
115     !
120     DO
130           CLEAR SCREEN
140           PRINT AT 10,5:"BOODSCHAPPEN:"
150           INPUT AT 12,5,PROMPT "Voer uw
              boodschap in:":BOODSCH$
160           INPUT AT 13,5,PROMPT "Hoe vaak zal ik
              dit afdrukken? ":AANTAL
170           PRINT AT 15,5:"MAAK EEN KEUZE
              TUSSEN:"
180           PRINT AT 17,5:"1) Onder elkaar"
190           PRINT AT 18,5:"2) Direct naast elkaar"
200           DO
220                 INPUT AT 21,5,PROMPT "Uw keuze
                    ? ":KEUS
230           LOOP WHILE KEUS<1 OR KEUS >3 OR
              KEUS< >INT(KEUS)
240           !
250           CLEAR SCREEN
260           SELECT CASE KEUS
270           CASE 1
280                 FOR L=1 TO AANTAL
290                   PRINT BOODSCH$
340                 NEXT
350           CASE 2
360                 FOR L=1 TO AANTAL
370                   PRINT BOODSCH$;
380                 NEXT
390           END SELECT
400           PRINT :PRINT
410           DO
420                 INPUT PROMPT "Meer (j/n) ?
                    ":ANTW$
430           LOOP UNTIL ANTW$="j"
              OR ANTW$="n"
440     LOOP WHILE ANTW$="j"
450     END
```

PROGRAMMA'S VERANDEREN

Probeer de programma's maar te veranderen als u dat wilt. Het is geen goede zaak wanneer u de schrijfwijze van de programma instructies wijzigt. Maar daar waar woorden tussen aanhalingstekens staan kunt u die veranderen zonder het programma zelf in de war te schoppen. Getallen kunt u eveneens veranderen. Door dit te doen kunt u erachter komen wat het programma bewerkstelligt — een gewijzigd getal zal vaak van invloed zijn op het aantal keren dat iets gedaan wordt of op de positie van tekens op het beeldscherm. Sommige getallen binnen een programma zijn codes, dat wil zeggen dat ze voor iets anders staan, een kleur bijvoorbeeld. Hierover zult u al het noodakelijke aan de weet komen verderop in deze handleiding (zie pagina 111 voor kleurencodes).

Bedenk altijd dat u kunt intikken wat u wilt, de computer zal daardoor volstrekt geen schade berokkend worden. Het ergste dat u kan overkomen is dat u iets intikt dat de computer niet kan begrijpen. Zo had u bij voorbeeld in het rangschikkingsprogramma (pagina 12) de volgende regel kunnen intikken:

```
25Ø FOR X=1 TOO 1Ø
```

In dit geval zou de computer na draaiing van het eerste gedeelte van het programma meteen stoppen en zou er het volgende op het scherm verschijnen:

```
*** Not understood
25Ø FOR X=1 TOO 1Ø
```

Foutboodschappen komen uitvoerig aan de orde op pagina 235-241.

STOPPEN EN STARTEN

Wanneer u volledig vast komt te zitten en opnieuw wilt starten moet de 'reset'knop aan de achterkant van de machine ingedrukt worden. Hierdoor zult u weer op het punt belanden waar u was toen u de machine aanzette, alleen zal de computer zich elk programma herinneren dat zoeven door u is ingetikt. Deze mogelijkheid echter kunt u het beste reserveren voor noodgevallen. Doorgaans zult u de 'stop' of de 'hold' (vasthouden) toets gebruiken om een lopend programma stop te zetten. Wanneer u gebruik maakt van 'reset' moet u weer het woord RUN intikken om het programma

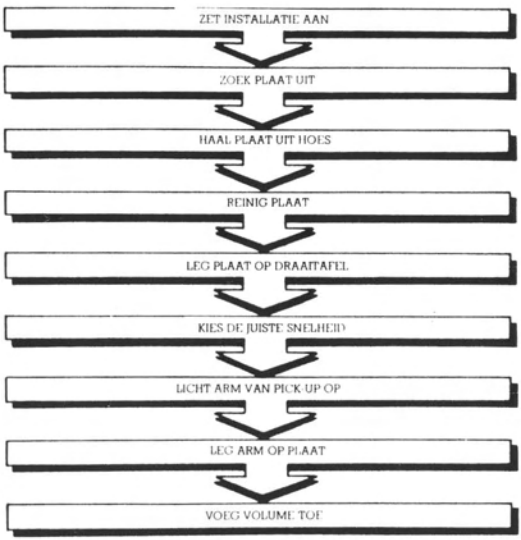
opnieuw te starten. De 'hold'toets wordt eenvoudigweg gebruik om het programma stop te zetten en de actie op elk gewenst moment te bevriezen. Dit is handig wanneer u bewegende beelden nader wilt bekijken — het is ook handig wanneer u alleen maar even wilt pauzeren halverwege een moeilijk spel zoals Space Invaders. Om het programma voort te zetten hoeft u alleen maar dezelfde toets aan te slaan.

Merk op dat de 'hold'toets uiterst geschikt is wanneer u een lang programma UITLIJST dat uit het beeld verdwijnt, en u de beweging stop wilt zetten om een speciale regel nader te bekijken.

De 'stop'toets wordt gebruikt wanneer u een lopend programma wilt stoppen om het te wijzigen (of uit te wissen) of UIT TE LIJSTEN en te controleren voordat u CONTINUE intikt om een en ander te hervatten.

Tot zover de pretjes. Hopelijk heeft u kennis met elkaar gemaakt. Waarschijnlijk heeft u nog niet begrepen hoe sommige programma's werkten; om die reden gaan we vanaf nu uw kennis vergroten.

In grote lijnen is het u al duidelijk wat een programma is; maar het schrijven van een programma behelst wel iets meer dan alleen maar het geven van instructies. Een programma kunt u beschouwen als een manier om met gebruikmaking van een computer een probleem op te lossen. Een computer is ofwel een gereedschap waarmee u uw intellectuele vermogens kunt uitbreiden en effectiever maken ofwel een prettig bezit waarmee u spelletjes kunt spelen of bedenken. Alle computers begrijpen instructies, meestal in de vorm van woorden of lijsten met getallen. Wij zullen woorden gaan gebruiken om met de Enterprise te communiceren. Elk woord is een kleine instructie; evenals de stukjes van een puzzle en de elementen van een verhaal worden deze kleine instructies samengevoegd tot grotere instructies en uiteindelijk tot volledige taken. In essentie komt programmeren hierop neer. Het geven van instructies aan de computer — in de door u gekozen volgorde. Onderstaand figuur laat zien (aan de hand van een alledaags voorbeeld) hoe een taak in verschillende kleine taken gesplitst moet worden om een programma te verkrijgen.



COMPUTERTALEN

Nu we weten wat een programma is gaan we een stapje verder. Zoveel verschillende manieren als er zijn om met een ander te praten, zoveel manieren zijn er om een computer te programmeren. En zo goed als er natuurlijke talen bestaan kennen we ook computertalen.

Computertalen hebben hun grenzen en zijn toegesneden op bepaalde soorten taken. Sommige zijn speciaal ontworpen voor programma's die lange lijsten met van alles en nog wat omvatten, andere zijn met name goed in het maken van tekeningen met de computer. En weer andere zijn er nog steeds om mensen het programmeren bij te brengen.

BASIC

De taal die u via deze handleiding leert staat bekend als BASIC. Deze taal maakt gebruik van woorden die een soortgelijke betekenis hebben als Engelse woorden. BASIC kan om die reden erg gemakkelijk geleerd en begrepen worden, zelfs wanneer u geen ervaring hebt met computers. Alle programma's in deze handleiding zijn in BASIC opgesteld, de taal die de computer begrijpt zolang de IS-BASIC cartridge ingestoken is. Onthoud dat van nu af aan alle instructies en alle informatie in deze handleiding naar BASIC verwijzen. Sommige andere talen zijn helemaal anders zowel wat filosofie als aanpak betreft, en meestal lijken ze in geen enkel opzicht op BASIC.

Bekijk deze regel eens:

```
1Ø PRINT "Hallo!"
```

Inmiddels zult u wel weten dat dit een programmaregel is — een kleine taak die deel zou kunnen uitmaken van een meer omvattende taak.

Zoals iedere zin in een verhaal de lezer iets meedeelt, zo laat elke regel in een programma de computer weten wat er gedaan moet worden. Wanneer u de computer een opdracht wilt laten uitvoeren heeft u verschillende regels nodig om dat te bewerkstelligen.

Het maakt nogal wat uit of er wel of niet een nummer aan het begin van een programmaregel in BASIC staat.

Tik PRINT "Hallo!"in. Wat op het beeldscherm verschijnt wanneer u 'enter' heeft ingedrukt laat u weten wat er gebeurt. De computer voert meteen uit wat hem is opgedragen — direct onder de woorden die u zoeven heeft ingetikt verschijnt "Hallo!". Bij eerdere programma's wachtte de computer steeds met de uitvoering ervan tot hem dat opgedragen werd. Het regelnummer maakt dit verschil uit.

Zonder regelnummer voert de computer uw instructies uit zodra u 'enter' indrukt. Wanneer u de regel wel een nummer geeft moet u RUN intikken (of functietoets 1 indrukken) om de computer te laten werken.

Dus nu weet u dat alle BASIC- programma's in regels onderverdeeld kunnen worden en dat elke regel een regelnummer moet hebben.

Wanneer u een programmaregel invoert met hetzelfde regelnummer als een regel die u al eerder hebt ingevoerd, wordt de regel die al in het computergeheugen zit automatisch gewist.

Wanneer u ooit van een regel af wilt hoeft u alleen maar het nummer daarvan in te tikken en vervolgens 'enter' in te drukken. De regel zal verdwijnen omdat het resultaat daarvan neerkomt op het invoeren van een lege regel! Met het commando DELETE (wissen) kunt u hetzelfde bewerkstelligen; ook kunt u hiermee meerdere regels verwijderen — DELETE 1Ø TO 1ØØ zou alle regels van 1Ø tot en met 1ØØ verwijderen.

De volgorde waarin regels de computer ingevoerd worden maakt niets uit. We hebben al gezien hoe de computer ze zelf numeriek ordent. Wat wel van belang is, is de numerieke volgorde.

De regelnummers geven aan in welke volgorde de computer instructies zal gaan uitvoeren. De computer begint bij het laagste nummer en werkt regel voor regel af — tenzij anders aangegeven door het programma — tot de laatste is bereikt. De 'tenzij anders aangegeven door het programma' is, naar verderop zal blijken, uiterst belangrijk. Er bestaan tal van manieren om een programma samen te stellen. Sommige hiervan houden in dat de computer de volgorde van regelnummers niet zal aanhouden, omdat bepaalde BASIC woorden hem laten weten in plaats daarvan

andere regels te gebruiken.

Dit betekent dat u precies moet weten welke dingen u de computer wilt laten doen en in welke volgorde — wat hetzelfde is als met zekerheid weten wat u wilt bereiken voordat u een programma gaat schrijven.

RECHTSTREEKSE MODUS

Wanneer u commando's intikt zonder regelnummer is er sprake van de rechtstreekse modus (direct mode). Op deze manier kunt u de computer als een rekenmachine gebruiken. Hieronder volgt. Op het toetsenbord zult u de volgende symbolen zien: +,*,/. Hiervan bent u misschien met * en / onbekend; * staat voor 'vermenigvuldigen' (in plaats van x) en / staat voor 'delen' (in plaats van :). Wanneer u

```
PRINT 2+2
```

intikt krijgt u 4 als antwoord zodra u 'enter' indrukt; u kunt met alle bekende wiskundige bewerkingstekens op deze manier berekeningen uitvoeren op de computer. Eveneens kunt u de vierkantswortel uit een getal trekken. Probeer maar

```
PRINT SQR (1ØØ)
```

en het antwoord zal de vierkantswortel uit 1ØØ zijn. SQR is een speciaal woord waarvan de computer de betekenis interpreteert als 'de vierkantswortel uit het getal tussen haakjes'. Van deze woorden hebben we er verschillende in BASIC. Een lijst hiervan kunt u in het naslaggedeelte terugvinden onder het kopje 'Ingebouwde functies en variabelen'.

Zodra deze woorden in de tekst voorkomen zullen we er tekst en uitleg bij geven.

De rechtstreekse modus zal van pas komen wanneer u de resultaten van berekeningen wilt verwerken in programma's. Toepassing van de rechtstreekse modus zal van geen enkele invloed zijn op programmaregels die u eventueel al heeft ingetikt en u hoeft niets speciaals te ondernemen om er gebruik van te maken. Het enige dat u moet doen is het regelnummer weglaten.

Vele BASIC commando's kunnen zowel bij toepassing van de rechtstreekse modus als binnen een programma uitgevoerd worden. Door te experimenteren zult u er achter komen wat dit betekent. Tik ze vervolgens door gebruik te maken van de voorbeelden die u in elk gedeelte aantreft, in via de rechstreekse modus. Uit de naslaglijst met sleutelwoorden (pagina 172) zal ook blijken welke commando's gebruikt kunnen worden bij toepassing

van de rechstreekse modus.

SLEUTELWOORDEN

Nu we op de hoogte zijn van het programmeren in BASIC en van de rechtstreekse modus, zullen we nader ingaan op sleutelwoorden. Een hiervan is PRINT, met een speciale betekenis in BASIC. Sleutelwoorden zijn de instructies die al eerder in deze handleiding aan de orde zijn gekomen. Hiervan zijn er vele en elk sleutelwoord draagt de computer op een enkel klein ding uit te voeren (eventueel meer dan een). PRINT bij voorbeeld draagt de computer op iets op het scherm te brengen.

Wilt u de computer ingewikkelde karweien laten uitvoeren, dan moet u vele sleutelwoorden en andere stukjes informatie gebruiken die samen een programma vormen. Wellicht kunt u dit zien als de computer een verhaal vertellen of een kind het alfabet bijbrengen — het geheel moet gedetailleerd, exact en in de juiste volgorde zijn.

STRINGS

Zoals u al heeft kunnen zien is het mogelijk om de computer een boodschap (Hallo!) op het scherm te laten zetten door de boodschap tussen aanhalingstekens (" ") in te tikken. Dit wordt door de computer als een string beschouwd (zie pagina 62 voor meer details over strongs). Dit kan op dit moment iterst verwarrend overkomen, daarom twee dingen die wellicht tot een beter begrip bijdragen.

Op de eerste plaats is het woord 'string' slechts een benaming voor een speciaal soort informatie waar de computer mee werkt. Op de tweede plaats kunt u in de veronderstelling verkeren — omdat strings in een programma tussen aanhalingstekens staan — dat we te maken hebben met de directe rede. Neem nu de regel PRINT "Hallo!": de computer zet alleen dat op het scherm dat u tussen aanhalingstekens heeft geplaatst.

TEKENS

Van tijd tot tijd zijn we het woord 'tekens' tegengekomen. Een teken is een letter, een cijfer of ieder ander symbool dat de computer beschikbaar stelt voor weergave-/communicatieve doeleinden. (Sommige hiervan hebben in feite helemaal geen betekenis, ze zien er alleen mooi uit en kunnen samengevoegd worden tot een fraaie tekening. We hebben hier te maken met grafische tekens.) Alle

tekens die de computer beschikbaar stelt worden te samen de tekenset genoemd.

VARIABELEN

Programmaregels, de rechstreekse modus en sleutelwoorden zijn nu aan de orde geweest; laten we nu eens iets anders onder de aandacht brengen.

Stelt u zich eens twee doosjes voor, X en Y. In elk doosje kunnen we een getal stoppen. Naderhand kunnen we dit getal eruit halen en vervangen door een ander.

Nu laten we de computer weten dat, telkens wanneer we het volgende programma laten lopen en we in beide doosjes een getal stoppen, de inhoud van X en die van Y bij elkaar opgeteld moeten worden.

```
10    !
20    !        60 en 70 verzoeken u de getallen in
25    !        te tikken. Ze wachten op uw
30    !        antwoord. Druk "enter" in wanneer
40    !        u elk getal heeft ingetikt.
50    !
60    INPUT PROMPT  "Tik getal X in: ":X
70    INPUT PROMPT  "Tik getal Y in: ":Y
80    !
90    !
95    !        Regel 120 telt getal X en Y bij elkaar
100   !        op en slaat de som op in een
105   !        variabele die we Z noemen.
110   !
120   LET Z=X+Y
130   !
140   !        Regel 160 geeft het antwoord op het
145   !        scherm weer.
150   !
160   PRINT X; "+"; Y; "="; Z
170   END
180   !
190   !        Regel 170 laat u en de computer
200   !        weten waar het programma eindigt.
220   !
```

Dit programma hoeft u misschien niet eens te draaien om te zien wat het bewerkstelligt. Het is niet onwaarschijnlijk dat u dit wel eens in uw schooljaren bent tegengekomen (of, als u ouder bent, in de wiskundeboeken van uw kinderen). Ziet u hoe u de computer precies kunt aangeven hoe te handelen.

OPMERKINGEN IN PROGRAMMA'S

De programmaregels die met een uitroepteken beginnen zijn er voor uw gemak — de computer memoreert ze weliswaar maar doet er verder niets mee. Deze regels bevatten opmerkingen die u informatie verstrekken over wat de afzonderlijke delen van een programma bewerkstelligen. Een uitroepteken kan ook gebruikt worden om de rest van een programmaregel voor opmerkingen te reserveren wanneer die regel al een of twee instructies bevat. Anderzijds kan in plaats van het uitroepteken ook het woord REM gebruikt worden, maar dit moet altijd meteen na het regelnummer volgen. De opmerkingen kunt u het beste als in bovenstaand voorbeeld weergeven, zodat ze duidelijker in het oog springen. Ze zijn niet van essentieel belang maar programma's worden daardoor wel begrijpelijk voor mensen.

INPUT PROMPT

INPUT PROMPT (letterlijk: de invoer souffleren) (zie programma op pagina 24) is de set sleutelwoorden die de computer laten weten dat de gebruiker een vraag gesteld moet worden. De computer gaat pas verder wanneer het antwoord ingevoerd is.

Wanneer u alleen INPUT gebruikt, zonder PROMPT eraan toe te voegen gevolgd door een paar woorden tussen aanhalingstekens (waarvan u alleen maar voordeel hebt), zal de computer zijn vraag stellen door slechts een vraagteken en de rode positie-wijzer in beeld te brengen.

De woorden na PROMPT bepaalt u zelf en zijn er alleen maar om u eraan te herinneren wat u dient in te tikken. Ze moeten altijd tussen aanhalingstekens ingevoerd worden (ofschoon deze niet op het scherm verschijnen). Een andere manier van "input prompt" is deze:

```
10 PRINT "Tik in getal X";
20 INPUT X
30 PRINT "Tik in getal Y";
40 INPUT Y
```

Zoals u kunt zien is deze methode iets omslachtiger dan INPUT PROMPT maar het resultaat blijft hetzelfde. Dit programma bewerkstelligt precies hetzelfde als regel 60 en 70 van het bovenstaande programma, alleen zal dit programma het mogelijk maken dat de computer

zijn eigen kleine "input prompt" afdrukt, wat neerkomt op een vraagteken. Een INPUT PROMPT regel weerhoudt de computer ervan een vraagteken af te drukken.

NOGMAALS VARIABELEN

Het woord variabele is een benaming voor getallen waarvan de waarde (d.w.z. de grootte) kan veranderen. X en Y zou u kunnen vervangen door ieder gewenst getal — of het nu Ø.ØØØØ4 is of 4ØØØØØ. Als X voor 4ØØØØØ zou staan, zouden we zeggen dat 4ØØØØØØ de waarde van X is.

Vaak zult u variabelen nodig hebben omdat u van tevoren niet weet wat de waarde van een getal is.

Laten we nog eens teruggaan naar de eerste bladzijden van deze handleiding.

Alle programma's die u daar heeft uitgeprobeerd bevatten variabelen, nu eens omdat de computer nieuwe bevatten variabelen, nu eens omdat de computer nieuwe getallen vormde uit andere, dan weer omdat u ze zelf tijdens het lopen van het programma vaststelde door ze in te tikken.

In andere gevallen is het handig variabelen een naam te geven en aldus te gebruiken voor getallen met erg veel cijfers die meerdere malen gebruikt zullen worden. Dit houdt in dat u ze niet telkens hoeft in te tikken wanneer ze in een programma nodig zijn.

Hieronder volgt nog een toepassing van variabelen:

```
1Ø LET A = Ø!                  Beginwaarde A=Ø.
2Ø DO!                         DO/LOOP start.
3Ø     LET A = A + 1 !         Tel 1 bij A op.
4Ø     PRINT A, !              Breng waarde van A in
                               beeld.
5Ø LOOP UNTIL A = 2Ø!          Ga terug, lus weer
                               doorlopen.
6Ø !                           Lus afgelopen wanneer
                               A=2Ø
7Ø END
```

Hier is sprake van een optelproces waarbij gebruik gemaakt wordt van een zogenaamde DO/LOOP. Telkens wanneer de Enterprise de lus doorloopt wordt het getal in A met 1 verhoogd. Vervolgens wordt aan het einde van de lus gecontroleerd of de waarde van A al 2Ø bedraagt. Zodra A de waarde 2Ø heeft is het

programma afgelopen.

De komma direct achter "A" in regel 40 geeft het geheel op het scherm een prettiger aanblik — door de komma weg te laten en het programma nog een keer te draaien kunt u zien wat er gebeurt.

Een DO/LOOP is niet de enige manier om een computer zichzelf te laten herhalen. Op pagina 70 en in het Naslaggedeelte zult u meerdere mogelijkheden aantreffen.

DEZE VARIABELE NOEM IK...

X, Y en Z zijn slechts drie naampjes die u aan een variabele kunt geven. Het hoeft niet slechts een enkele letter te zijn — indien gewenst kunt u meer dan dertig letters gebruiken. U kunt dus al uw variabelen een passende naam geven -NAMEN$ als het om iemands naam gaat (hier hebben we te maken met een string-variabele; zie verder pagina 63), SOM wanneer we met de uitkomst van een optelling te maken hebben.

Merk op dat het niets uitmaakt of u nu een variabele PRIJS, Prijs, prijs of pRijS noemt. De computer maakt geen onderscheid tussen hoofdletters en kleine letters in namen van variabelen of in sleutelwoorden.

Wanneer u eigen programma's gaat schrijven zult u snel genoeg inzien hoe benamingen voor variabelen u kunnen helpen het programma naderhand door te lezen. Dit is uiterst belangrijk wanneer u fouten moet opsporen, veranderingen aanbrengen of een gedeelte uit een programma halen voor gebruik in een ander programma — dit alles zult u uiteindelijk gaan doen.

Incidenteel zou u twee variabelen kunnen gebruiken met nagenoeg gelijkluidende namen — bij voorbeeld:

```
VAT_SUBTOTAL_JUNE_ACCOUNT_NO1
en
VAT_SUBTOTAL_JUNE_ACCOUNT_NO 2
```

En toch zou de Enterprise in staat zijn om met een oogopslag het verschil vast te stellen. Het probleem is of u dat ook zou kunnen. Het is niet erg verstandig om voortdurend benamingen van dit kaliber aan te wenden.

Af en toe kunnen ze wel van pas komen. Maar meestal zijn acht of tien tekens voldoende om het verschil duidelijk te maken of om snel te achterhalen

welke variabele voor welk getal staat. Variabelen bieden u dus de mogelijkheid om met getallen te werken waarvan de feitelijke waarde u onbekend is. De naam die u aan elk getal geeft verwijst naar datgene waarmee het verband houdt. De computer kan zelfs het verschil vaststellen tussen variabelen die uit 31 tekens bestaan.

"Teken" staat hier voor: een letter of een getal (preciezer uitgedrukt: een cijfer). De computer zal u niet begrijpen wanneer u spaties aanbrengt in variabelen, evenmin zal hij bewerkingstekens (+,= etc.) of interpunctietekens begrijpen met uitzondering van de punt (.) en het onderstrepingsteken, (_); de laatste is een prima substituut voor de spatie.

Variabelen moeten altijd met een letter beginnen, niet met een cijfer of een interpunctieteken. Hieronder volgen enkele "legale" variabelen — d.w.z. variabelen die door de computer geaccepteerd worden:

```
getal        A2$        TOTAL$        Hallo1
SUB_TOTAL_3             Naam          A1234
```

Merk op dat het dollarteken gebruikt kan worden. Dit teken heeft een speciale betekenis: immers hierdoor wordt de desbetreffende variabele gereserveerd als string-variabele. Strings zijn al in het kort uitgelegd op pagina 22 en zullen uitvoeriger aan de orde komen op pagina 62-69.

Hieronder volgen enkele variabelen die niet begrepen worden door de computer:

```
mijn variabele        2A          !!!A$
<>3NOW                3*THIS NUMBER
```

HET BENOEMEN VAN VARIABELEN

In het optel-programma luidde de eerste regel LET A = Ø. Probeer het programma ook eens zonder deze regel. Het zal geen effect sorteren, niet waar?

Maar als u gebruik maakt van een INPUT-opdracht waarmee u de computer laat weten dat hij u de waarde van een variabele moet laten intikken (zoals in het programma waarin twee variabelen van uw keuze bij elkaar opgeteld werden) hoeft u deze variabele niet te benoemen door gebruik te maken van LET.

DE TOEPASSING VAN VARIABELEN

Probeert u aan het begin van het optel-programma eens LET A=3 in plaats van LET A=Ø in te tikken. Op die manier komt u iets meer te weten over de werking van variabelen.

De computer verwacht de variabele die gewijzigd of benoemd wordt direct achter LET. Dus als variabele S=Ø variabele G=1Ø zou u LET S=G kunnen intikken om S in 1Ø te veranderen. G behoudt de waarde 1Ø maar de computer gaat ervan uit dat de waarde van G gelijk is aan die van S.

LET introduceert een variabele — u zegt als het ware: "Computer, hier volgt een variabele met de naam FIRSTNUM (eerste getal); op dit moment is het gelijk aan Ø of aan ieder getal dat je ervan wilt maken". Het is niet altijd noodzakelijk om LET te gebruiken, maar in het algemeen kunt u het beter wel doen. Hierdoor wordt het eenvoudiger uw variabelen meteen te ontdekken.

Indien gewenst kunt u een BASIC woord als naam van een variabele gebruiken. Maar als u dat doet, moet u LET gebruiken om de computer te laten weten dat hij op zoek moet naar een variabele met dezelfde naam als een van de woorden die hij begrijpt. Er zijn een paar BASIC woorden die niet als benaming voor een variabele aangewend kunnen worden — door te experimenteren zult u er snel genoeg achter komen om welke woorden het gaat. (Een veelzijdiger manier om variabelen te benoemen vindt u op pagina 86, in het hoofdstuk "De opslag van grotere hoeveelheden informatie".)

Hieronder volgt een nieuw programma dat alles illustreert dat u zoeven onder ogen heeft gekregen. Dit maal wordt elke regel uitgelegd met toepassing van de nieuw opgedane kennis over variabelen.

```
1Ø    LET A$= "De som van beide getallen is:"
2Ø    !
3Ø    !          1Ø benoemt string-variable A$.
4Ø    !
5Ø    LET SUM = Ø
6Ø    !
7Ø    !          SUM is de variabele die de uitkomst
75    !          van onderstaande optelling zal
8Ø    !          bevatten. Beginwaarde:Ø.
9Ø    !
```

```
100     INPUT PROMPT "Tik a.u.b. uw eerste getal in:
        ": FIRSTNUM
110     INPUT PROMPT "Tik uw tweede getal in: ":
        SECNUM
120     !
130     !           100 en 110 verzoeken u de getallen
135     !           in te tikken die opgeteld moeten
140     !           worden. U hoeft geen gebruik te
145     !           maken van LET omdat de twee
150     !           nieuwe variabelen "ingeput"
155     !           worden.
160     !
170     LET SUM = FIRSTNUM + SECNUM
180     !
190     !           SUM (die 0 bedroeg) wordt nu de
195     !           som van FIRSTNUM en SECNUM.
210     !
220     PRINT A$;SUM
230     !
240     !           Regel 220 laat de computer weten
245     !           dat op dezelfde schermregel de zin
250     !           van regel 10 en de waarde van SUM
260     !           moeten verschijnen.
290     !
300     END
```

Dit is een uitdrukking: 4 + 2*3-5

Dit zijn bewerkingstekens: ^, +, -, *, /, =, <, >, < >, < = en > =.

Alle bovenstaande bewerkingstekens kunnen op de Enterprise gebruikt worden. Sommige zullen u wel al bekend zijn, andere misschien niet. De symbolen +, -, *, / en = zou u nu al moeten kennen; de computer gebruikt * in plaats van x en / in plaats van:

Voor alle zekerheid volgt hier de rest:

^ betekent "tot de macht" of "machtsverheffing"; 2^3 is twee tot de derde macht.
< betekent "kleiner dan", bijv. 2<3
> betekent "groter dan", bijv. 3>2
< > betekent "groter of kleiner dan" of "niet gelijk aan".
< = betekent "kleiner dan of gelijk aan".
> = betekent "groter dan of gelijk aan".

De laatste vijf bewerkingstekens kennen we als vergelijkings-bewerkingstekens. Ze worden hoofdzakelijk toegepast in verband met variabelen, bij voorbeeld:

```
10    INPUT PROMPT "Tik a.u.b. eem getal in: ": A
20    !
30    !          Regel 10 vraagt om een getal
35    !          waarvan de waarde nog
36    !          vastgesteld moet worden.
40    !
50    If A < 0 THEN PRINT "Dit getal is negatief."
55    IF A = 0 THEN PRINT "Dit getal is nul!"
60    !
70    !          50 neemt een eenvoudige
75    !          beslissing. Indien A een negatief
80    !          getal is b.v. -9), zeg dat dan. IF is
90    !          van cruciaal belang. Als die regel
100   !          niet van toepassing is op getal A
110   !          gaat de computer op zoek naar een
120   !          regel die wel van toepassing is op
130   !          A. A< 0 in regel 50 betekent 'A is
140   !          kleiner dan 0'.
145   !
150   IF A > 0 AND A < 50THEN PRINT "Dit getal is
      groter dan 0 en kleiner dan 50".
```

```
160   !
165   !        150 is een andere "IF/THEN". Net
170   !        als in het Engels -als (IF) je niet wilt
180   !        komen, ga dan (THEN) maar naar
190   !        huis. Nu bekijkt de computer of A
195   !        groter is dan Ø maar kleiner dan 5Ø.
200   !        Zo niet bekijkt hij onderstaande
205   !        regel.
210   !
220   IF A > 5Ø AND A < 1ØØTHEN PRINT "Dit getal
      is groter dan 5Ø en kleiner dan 1ØØ."
230   !
240   !        Regel 3ØØ is de laatste
245   !        beslissingsregel. Hier wordt
250   !        bekeken of A meer dan
260   !        1ØØ bedraagt.
290   !
300   IF A > 1ØØ THEN PRINT "Dit getal is groter dan
      1ØØ."
310   END
```

In overeenstemming met de instructies die ontvangen worden beslist het programma in welke categorie uw getal valt. Dit is een van de manieren waarop de computer beslissingen kan nemen. Er zijn ook nog tal van andere manieren waarop dit kan gebeuren. Deze komen aan de orde op pagina 76.

Door middel van vergeleken worden. Ze zijn volstrekt niet van invloed op de waarde van getallen. De bewerkingstekens die hieronder ter sprake komen zijn stuk voor stuk wel op de een of andere manier van invloed op de waarde van getallen.

BEWERKINGSTEKENS EN HUN PRIORITEIT

Laten we nogmaals de uitdrukking: 4 + 2*3-5

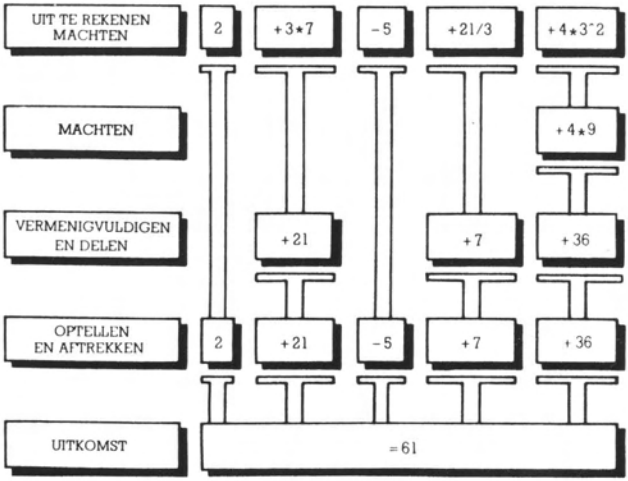
Wat is volgens u de uitkomst? Dertien? Omdat u elk gedeelte van links naar rechts afwerkt? Enfin, de uitkomst is vijf en wel hierom.

Machten worden eerst berekend. Wanneer 4^3 van de uitdrukking deel had uitgemaakt, was dit eerst uitgerekend. Daarna volgen vermenigvuldigen en delen. Eerst zou 2*3 of 6/2 uitgerekend worden voordat de computer overgaat tot optellen of aftrekken. Vermenigvuldigen en delen hebben dezelfde prioriteit. Zou een uitdrukking bij voorbeeld twee delingen en een vermenigvuldiging bevatten, dan zou de machine

ze van links naar rechts de een na de ander afwerken.

Hierna zakt de computer af naar het volgende prioriteitenniveau en houdt zich bezig met optellen en aftrekken. Deze bewerkingen worden ook van links naar rechts uitgevoerd en tegen deze tijd zou u een uitkomst moeten hebben — in ons voorbeeld 5.

Laten we nu eens een omvangrijkere uitdrukking bekijken en deze in kleinere deeltjes splitsen zoals de computer dat zou doen.



2 + 3 *7 -5 + 21/3 +4*3^2 is de uitdrukking die we hebben genomen.
3^2 = 9
Nu krijgen we het volgende:
2 + 3*7 - 5 + 21/3 + 4*9
Hierna komen de vermenigvuldigingen en delingen — het tweede niveau.
3*7 = 21 21/2 = 7 4*9 = 36
Nu hebben we:
2 + 21 - 5 + 7 + 36
En dit wordt van links naar rechts uitgewerkt, dus:
2 + 21 = 23 23 - 5 = 18 18 + 7 = 27 25 + 36 = 61
De uitkomst is dus 61.

Zoals u constateren kunt is dit bepalend voor de uitkomst van uw berekeningen. Wellicht verwacht u een bepaalde uitkomst terwijl de computer een andere geeft.

Indien gewenst kunt u de prioriteiten veranderen en verschillende gedeeltes van een uitdrukking met voorrang door de computer laten behandelen. Het enige dat u moet doen is het gedeelte dat u het eerst uitgerekend wilt hebben tussen haakjes plaatsen.

Let u maar eens op het verschil dat we krijgen met bovenstaande uitdrukking:

(2 + 3)*7 - 5 + 21/3 + 4*3^2
De computer behandelt (2 + 3) als een op zich staande eenheid en werkt deze dus het eerst uit. Zo krijgen we:
5*7 - 5 + 21/3 + 4*3^2

Daarna gaat de computer als gebruikelijk te werk en neemt de machten voor zijn rekening
3^2 = 9
Dus:
5*7 - 5 + 21/3 + 4*9

Hierna volgen vermenigvuldigen en delen — van links naar rechts:
5*7 = 35 21/3 = 7 4*9 = 36
Resteert:
35 - 5 + 6 + 36
om uit te werken:
35 - 5 = 3Ø 3Ø + 7 =37 37 +36 = 73.

De uitkomst bedraagt nu 73 en niet 61 zoals hiervoor. Naar u kunt zien, kunt u dus met een computer op allerlei manieren met getallen goochelen.

Bedenk dat hetzelfde prioriteitensysteem van toepassing is op uitdrukkingen die tussen haakjes staan. In geval van (2 + 3*4) zal de vermenigvuldiging dus eerst uitgewerkt worden en daarna de optelling.

Probeert u bovenstaande voorbeelden maar eens uit. Het best kunt u PRINT gebruiken in de rechtstreekse modus, bij voorbeeld:
PRINT 5*7 - 5 21/3 + 4*9.

Tot slot kunt u zo vaak als u wilt getallen met decimalen (b.v. Ø.23345 of Ø.ØØ98) en negatieve getallen gebruiken.

Al deze getallen zult u wel saai hebben gevonden. Maar als u eenmaal gewend bent aan de manier waarop de computer met getallen omgaat, zal dat niet meer het geval zijn. Hoewel u nooit een wiskundige

hoeft te worden om een computer bekwaam te programmeren, moet u niet vergeten dat u altijd gebruik moet maken van rekenkunde. Zelfs commando's voor grafische gegevensverwerking kunnen niet buiten getallen die uitgewerkt moeten worden — ofschoon dit vaak erg gemakkelijk is.

Een puntkomma laat de computer weten datgene dat aan weerszijden van de puntkomma staat achter elkaar op het scherm af te drukken. Een programma met vijf afzonderlijke PRINT opdrachten maar zonder puntkomma's zou er als volgt uitzien:

```
Hallo,
ik
ben
uw
computer
```

in plaats van

```
Hallo, ik ben uw computer
```

Er zijn ook nog andere manieren om dingen zodanig in beeld te brengen dat ze fraai ogen.

Probeert u het maar eens met een komma, zoals hieronder (dit programma zijn we al tegengekomen op pagina 26):

```
10  LET A = 0 !                    De waarde van A is 0
20  DO !                           Start DO/LOOP
30  LET A = A + 1 !                Tel 1 bij A op.
40        PRINT A,
50        !
60        !     Druk A af. De komma laat de
65        !     computer weten acht tekenposities
70        !     open te laten tussen deze en de laatst
75        !     afgedrukte A.
80        !
90  LOOP UNTIL A = 20 ! Lus nogmaals doorlopen
    indien A < 20
100 END
```

Ziet u hoe de komma een en ander in kolommen rangschikt? Normaliter wordt het scherm door de computer in 40 'tekenposities' horizontaal verdeeld en 24 verticaal. Doorgaans bewerkstelligt een komma dat tussen de afzonderlijke items telkens acht tekenposities open blijven.

Voor de opmaak van teksten kunt u met een komma ongeveer hetzelfde teweegbrengen als met een puntkomma, alleen brengt de komma spaties aan

tussen de ene string (het ene getal) en de andere (en het andere). Door gebruik te maken van de puntkomma wordt een en ander slechts achter elkaar afgedrukt op een regel.

Probeert u dit korte programma maar eens:

```
10 PRINT "Er was eens een oude man uit ST. Bees"
20 PRINT
30 PRINT "Die door een wesp op zijn hoofd gestoken
   werd."
40 PRINT
50 PRINT "Op de vraag of het pijn deed"
60 PRINT
70 PRINT "Antwoordde hij: 'nee,'"
80 PRINT
90 PRINT "'Voor mijn part mag ze me nog eens
   steken!'"
100 END
```

Regel 20, 40, 60 en 80 drukken elk een lege regel op het scherm af — zoals met een schrijfmachine de regelafstand geregeld kan worden. Het woord PRINT op zich betekent 'druk een lege regel af' of 'sla een regel over en ga verder op de volgende'.

Het is niet mogelijk dubbele aanhalingstekens op dezelfde manier als andere tekens op het scherm af te drukken (b.v. PRINT " " ") omdat dit verwarrend is voor de computer. Maar wanneer u de dubbele aanhalingstekens tweemaal intikt, krijgt u ze een keer op het scherm. Tikt u regel 70 van het laatste programma maar eens als volgt:

```
70 PRINT "Antwoordde hij" "nee"
```

PRINT AT

Met PRINT AT kunt u op een andere manier tekst weergeven. U weet al wat het effect van PRINT is. U kent ook de betekenis van het woordje AT (op) — AT laat de computer weten waar op het scherm bepaalde tekens weergegeven moeten worden.

Probeert u dit maar eens:

```
PRINT AT 20, 20: "Ik ben hier"
```

Stelt u zich eens voor dat het beeldscherm in posities is verdeeld; 40 horizontaal en 24 verticaal. Met

behulp van twee cijfers kan elke afzonderlijke positie geadresseerd worden. Bovenstaande string wordt afgedrukt op de 20-ste regel van boven en de 20-ste van links (of 20-ste kolom van links).

PRINT AT 10, 20 herhaalt de string, maar dan 10 posities hoger op het scherm.

Onderstaand diagram laat zien hoe het beeldscherm doorgaans uit 960 (24*40) tekenposities is opgebouwd; met behulp hiervan kunt u achterhalen waar zich elke afzonderlijke positie op het beeldscherm bevindt.

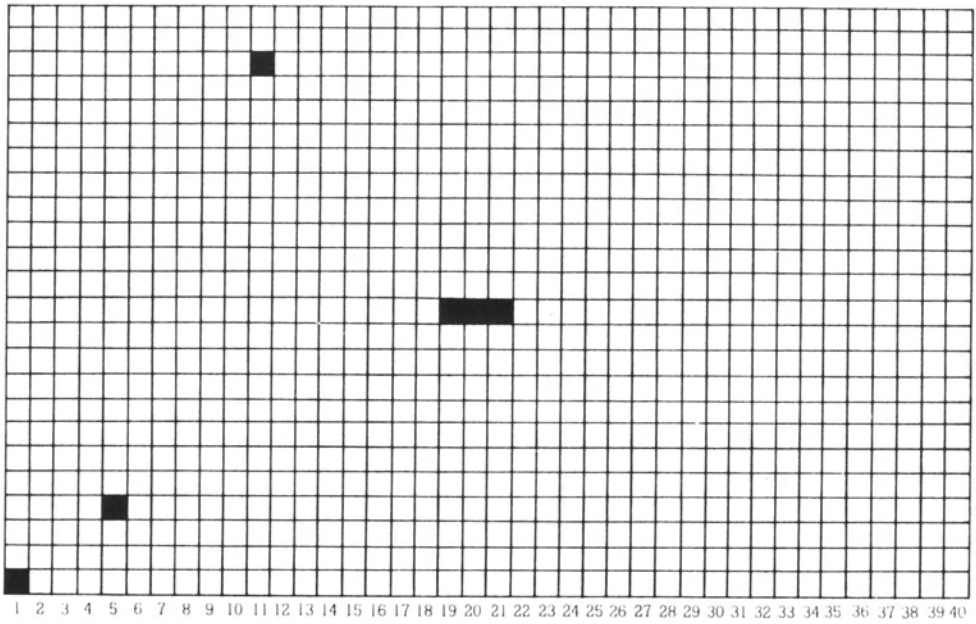
80-KOLOMMEN SCHERM.

De PRINT AT positie wordt geselecteerd door de computer te laten weten op welke hoogte (regel-nummer) en op welke breedte (kolom-nummer) u uw string wilt plaatsen. Een tekenpositie is een 'imaginair' vierkantje op het scherm dat een enkel teken kan bevatten. Eeen soortgelijk maar iets afwijkend systeem wordt gebruikt bij grafische gegevensverwerking. Zie hiervoor pagina 105-106.

Een aantal meer verfijnde manieren voor de opmaak van tekst wordt toegelicht op pagina 104 en verder.

Met het commando TEXT 80 kunt u in plaats van slechts 40 tekens wel 80 tekens op elke regel tikken; de tekens zijn nu natuurlijk wel kleiner. (Tegelijkertijd wordt het scherm op blank gesteld door dit commando.)

Met het commando TEXT 40 keert u weer terug naar een scherm met 40 kolommen.


1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24
1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31 32 33 34 35 36 37 38 39 40

U weet al dat u tikfouten kunt corrigeren door gebruik te maken van het spelpookje en de 'erase' toets. Nu zullen we een aantal verfijndere manieren bekijken om programma's te wijzigen — nieuwe regels invoegen, de regelnummers veranderen, gedeeltes van regels wijzigen, enzovoort. Soms kan dit uiterst gecompliceerd zijn. Om een en ander te vereenvoudigen voorziet de computer in 'tekstverwerking'; deze voorziening zal voor een deel in dit hoofdstuk geintroduceerd worden en meer volledig aan de orde komen in het volgende hoofdstuk.

Laten we de commando's voor de opmaak van programma's een voor een doornemen.

OPNIEUW NUMMEREN

Bij het intikken van programma's zal het u wel opgevallen zijn dat de regelnummers vaak bij 10Ø begonnen en verder gingen met 11Ø, 12Ø, 13Ø... In plaats hiervan hadden we ook 1, 2, 3, 4... kunnen gebruiken.

Doorgaans worden regels niet met 1, 2, 3, 4... genummerd omdat later wellicht nieuwe regels ingevoegd moeten worden — zoals het bij het schrijven van een verhaal kan gebeuren dat u zich halverwege realiseert dat u aan het begin iets vergeten hebt. Regelnummer 2,5 kennen we niet maar u kunt (als u geen kant meer uit kunt) RENUMBER (opnieuw nummeren) intikken en de computer zal alle regelnummers veranderen. Wanneer u RENUMBER STEP 1ØØ intikt zal de regelnummering steeds met 1ØØ oplopen (te beginnen bij 1ØØ). RENUMBER op zich begint te hernummeren vanaf 1ØØ in stappen van 1Ø. Dit laatste kan ook bereikt worden door 'shift' in te drukken en tevens functietoets 3.

STEP

STEP is een sleutelwoord dat in BASIC van tijd tot tijd opduikt in combinatie met een ander sleutelwoord (zoals hierboven met RENUMBER). STEP betekent 'telkens verhogen met ...'. STEP 1ØØ betekent dus 'telkens met 1ØØ verhogen', d.w.z. 1ØØ, 2ØØ, 3ØØ...

AUTO(MATISCH)

Een ander handig woordje bewerkstelligt dat de computer automatisch de programmaregels voor u invoegt. Dit woord is AUTO(matisch). Tik het in en het nummer 1ØØ zal op een nieuwe regel verschijnen.

Daarna kunt u uw programmaregel intikken en 'enter' indrukken. Nu verschijnt 11Ø, enzovoort. Wilt u

niet langer van de automatische nummering gebruik maken, dan hoeft u alleen maar de 'stop' toets aan te slaan.

De plaats in een programma waar u de automatische nummering wilt laten beginnen kunt u specificeren. AUTO AT 200 brengt teweeg dat de automatische nummering vanaf regel 200 begint.

Bij AUTO AT 200 STEP 100 zal de nummering vanaf 200 beginnen en daarna zal elke regel telkens met 100 verhoogd worden. U kunt elk getal van 1 tot 9999 gebruiken; 9999 is het hoogste regelnummer dat toegestaan is. (Wanneer u AUTO AT 9999 STEP 10 of iets dergelijks intikt brengt u de computer in verwarring.) Bedenk wel dat, wanneer bijvoorbeeld regel 250 via automatische nummering ingevoerd wordt, elke denkbare regel met nummer 250 die u al had ingetikt door de nieuwe regel vervangen wordt.

WISSEN EN UITLIJSTEN

Het woord NEW heeft u al gebruikt — hierdoor wordt het programma waarvan u gebruik maakt uit het geheugen van de computer verwijderd. Het woord DELETE (gevolgd door 'enter') bewerkstelligt hetzelfde.

Wanneer u DELETE 100 intikt wordt regel 100 van het programma gewist. DELETE 100 TO 140 bewerkstelligt dat regel 100 tot en met 140 gewist wordt.

Wanneer u slechts een regel wilt wissen is het beter alleen het nummer ervan in te tikken en 'enter' in te drukken. Het effect hiervan is hetzelfde als dat van DELETE regelnr...

In combinatie met DELETE kunt u de woorden FIRST (eerste) en LAST (laatste) gebruiken. Zo zou door DELETE FIRST TO LAST een volledig programma gewist worden — NEW is uiteraard een beter commando (of alleen DELETE). Maar het commando DELETE FIRST TO 100 is uiterst praktisch wanneer u alle regels tot en met 100 van een omvangrijk programma wilt wissen.

Het is ook mogelijk kleinere gedeeltes van een programma te wissen, d.w.z. afzonderlijke woorden of tekens binnen een programmaregel. Tabel 1 aan het einde van dit hoofdstuk laat zien hoe dat in zijn werk gaat. Wanneer u een regel zoekt die veranderd moet worden moet u LIST intikken. Zoals u weet zal dit woord het hele programma in beeld brengen. LIST gevolgd

door een specifiek regelnummer zal bewerkstelligen dat alleen die ene regel op het scherm verschijnt (direct boven de positie-wijzer). Door LIST... TO... in te tikken zal een aantal regels in beeld komen, te beginnen bij het eerste in het commando vermelde nummer tot en met het tweede. Evenals met DELETE kunt u FIRST en LAST in combinatie met LIST gebruiken.

Vervolgens moet u de positie-wijzer aan het begin of einde van het gedeelte van de regel plaatsen dat u wilt verwijderen. Hiervoor maakt u natuurlijk gebruik van het spelpookje. Wanneer u de regel eenmaal heeft veranderd moet u 'enter' indrukken terwijl de positie-wijzer zich nog steeds op dezelfde regel bevindt. Hierdoor zal de gewijzigde regel in plaats van de oorspronkelijke ingevoerd worden. (Merk op: zolang dit niet is gebeurd zal de computer zich de oorspronkelijke regel 'herinneren').

OVERSCHRIJVEN OF TUSSENVOEGEN

Wanneer u iets halverwege een programmaregel wilt veranderen hoeft u niet noodzakelijkerwijs de 'erase' of de 'del' (wissen) toets te gebruiken. Zolang de overschrijfmodus van toepassing is, wordt elk teken dat zich onder de positie-wijzer bevindt automatisch verwijderd zodra u een nieuw intikt — u 'overschrijft' en vervangt de oude gegevens. Dus wanneer u het getal 234 in 567 wilt veranderen, hoeft u alleen maar de positie-wijzer boven de 2 te plaatsen en de drie nieuwe tekens in te tikken.

Wanneer u nieuwe tekens wilt tussenvoegen zonder tegelijkertijd de oude te verwijderen, kunt u de computer instellen op de tussenvoegmodus. Om van 'overschrijven' op 'tussenvoegen' en omgekeerd om te schakelen moet u de 'CTRL' toets ingedrukt houden en 'ins' (tussenvoegen) aanslaan. In geval van de tussenvoegmodus krijgt de positiewijzer de vorm van een pijl die naar links wijst.

In de tussenvoegmodus wordt alles dat u halverwege een regel intikt tussen de al aanwezige tekens gevoegd — de tekens die rechts van de positie-wijzer staan schuiven op om plaats te maken voor de nieuwe. Dit houdt soms in dat woorden niet vergeten — u kunt ze weer in beeld krijgen door 'ctrl' ingedrukt te houden en functietoets 1 in te drukken (zoals beschreven in het volgende hoofdstuk). Zolang er

tekens 'uit het beeld' zijn kunt u het symbool '>' aan het eind van de regel waarnemen.

Een andere manier om 234 in 567 te veranderen is dus het oude getal wissen en vervolgens de 'tussenvoegmodus' kiezen teneinde het nieuwe getal ertussen te voegen.

HET INVOEGEN VAN REGELS

Om een geheel nieuwe programmaregel in te voegen hoeft u alleen maar te bepalen waar deze moet komen en hem met het juiste nummer in te tikken. Het regelnummer laat de computer weten waar de regel ingepast moet worden — zoals we helemaal aan het begin van deze handleiding hebben gezien.

DE POSITIE-WIJZER (CURSOR) VERPLAATSEN

We weten dat, wanneer het spelpookje in een bepaalde stand staat, de positie-wijzer hierdoor onafgebroken in die richting gaat totdat het spelpookje in een andere stand gebracht wordt. En u zult er wel al achter zijn gekomen dat de tekst, telkens wanneer de positie-wijzer de boven — of de onderkant van het scherm heeft bereikt, als het ware 'doorgedraaid' wordt om nieuwe regels in beeld te brengen.

Wanneer de positie-wijzer naar boven of beneden gaat kunt u deze een pagina per keer (een vol scherm) laten verspringen door op 'shift' te drukken. Dit is handig wanneer u snel van het ene naar het andere gedeelte van een lang programma wilt gaan (wanneer u in plaats van 'shift' 'ctrl' gebruikt, verspringt de positiewijzer een alinea per keer — dit komt nogal van pas bij tekstverwerking). Drukt u op 'ctrl', terwijl de positie-wijzer zijwaarts bewogen wordt, dan verspringt deze telkens een woord in plaats van een letter. Door 'shift' in te drukken en het spelpookje zijwaarts te bewegen wordt de positie-wijzer direct in de door u gewenste richting naar het begin dan wel het eind van de regel gebracht.

De volgende twee tabellen laten u alle verschillende manieren zien om tekens tussen te voegen en te wissen. De plaats van de positie-wijzer wordt altijd bepaald door gebruik te maken van het spelpookje.

Probeer deze functies uit met een programma — wellicht zit er al een in de computer. In het begin lijkt een en ander misschien een beetje verwarrend, maar als u eenmaal weet welke toetsen welke functie

hebben, zult u tot het oordeel komen dat tekstverwerking (zie volgende hoofdstuk) erg gemakkelijk is.

TABEL 1: WISSEN

FUNCTIE	TOETS (COMBINATIE)	UITZONDERINGEN/SITUATIE
Teken links uitwissen: teken links van de positie-wijzer wordt uitgewist en de rest van de regel sluit aan.	ERASE	Positie-wijzer gaat spatie naar links. Wanneer de positie-wijzer zich al in de meest linkse positie bevindt, wordt de regel met de laatste regel verbonden.
Regel links uitwissen: alles links van de positie-wijzer wordt uitgewist. Het gedeelte rechts ervan schuift naar links op en sluit aan.	SHIFT + ERASE	Positie-wijzer blijft in meest linkse teken-positie.
Het teken rechts wissen: het teken onder de positie-wijzer wordt gewist. Het rechter gedeelte van de regel schuift naar links op en sluit aan.	DEL	Positie-wijzer blijft in dezelfde positie. Het zijn de tekens die opschuiven! Bevindt de positie-wijzer zich aan het eind van de regel, dan schuift de volgende regel op.
Regel rechts wissen: alles rechts van de positie-wijzer wordt gewist.	SHIFT + DEL	Positie-wijzer blijft in dezelfde positie.
Het woord links uitwissen: wist de tekens links uit totdat het meest linkse teken van het woord is uitgewist.	CTRL + ERASE	Begonnen wordt met de tekenpositie links van de positie-wijzer. Alleen letters of cijfers worden gezien als deel van het woord. Tussengevoegde speciale tekens worden gewist.
Het woord rechts wissen: de tekens rechts worden gewist totdat het meest rechtse teken van het woord is gewist.	CTRL + DEL	Begonnen wordt met de tekenpositie onder de positie-wijzer; verder als hierboven.

TABEL 2: TUSSENVOEGEN

FUNCTIE	TOETS (COMBINATIE)	UITZONDERINGEN/SITUATIE
Een teken tussenvoegen: schuift tekens vanaf positie-wijzer tot het eind van de regel een positie op. Plaatst een spatie onder de positie-wijzer.	INS	In geval van de tussen-voeg-modus is dit gelijk aan het aanslaan van de spatiebalk, de positie-wijzer blijft op zijn plaats.
Een regel tussenvoegen: alle tekens vanaf positie-wijzer tot het eind van de regel 1 regel omlaag.	SHIFT + INS	Wanneer de positie-wijzer zich aan het begin van een regel bevindt, wordt zo een lege regel verkregen.
Flip-Flop invoeren: switchen tussen de tussenvoeg-modus en de overschrijf-modus.	CTRL + INS	Afhankelijk van de ene of de andere modus verschilt de positie-wijzer.
TUSSENVOEG-MODUS: Maakt het mogelijk de positie-wijzer in elke positie in een programma of tekst te brengen en tekens in te tikken die ingepast worden door de daarop volgende tekens naar rechts op te schuiven.		
OVERSCHRIJF-MODUS: Nieuwe tekens die ingetikt worden vervangen de al aanwezige. Er wordt geen spatie gecreeerd door tekens naar rechts op te schuiven. U tikt letterlijk tekens 'over' die al eerder ingetikt zijn.		

Uiterst waardevol is de mogelijkheid uw computer als een geavanceerde schrijfmachine te gebruiken. Wat u wilt kunt u intikken en met een printer — als u die heeft — afdrukken. Het voordeel van de computer tegenover een gewone schrijfmachine is dat fouten aanmerkelijk eenvoudiger gecorrigeerd kunnen worden; bovendien staan u verschillende speciale functies ter beschikking die van nut zijn bij het indelen en het verplaatsen van uw alinea's.

DE TEKSTVERWERKER

Onafhankelijk van de BASIC cartridge kan de ingebouwde tekstverwerker van de Enterprise gebruikt worden. Wanneer de cartridge erin zit en u van programmeren op tekstverwerken wilt overschakelen, moet u het woord TYPE tikken en 'enter' aanslaan. Op uw scherm verschijnt nu de aanwijzing "Press ENTER to exit BASIC". Druk vervolgens de ENTER-toets in. Het beeld op het scherm zal nu veranderen. Boven en onder in het beeld ziet u een tekst die aanwijzingen bevat over het gebruik van de functietoetsen voor tekstverwerkende doeleinden.

U krijgt hetzelfde beeld wanneer u de computer aanzet en de cartridge er niet in zit.

Onthoud dat het commando TYPE ieder denkbaar BASIC programma uit het computergeheugen wist. Door 'shift' aan te slaan en tevens functietoets 8 bereikt u hetzelfde als met TYPE.

Indien u de tekenset gedeeltelijk of in zijn geheel opnieuw heeft gedefinieerd (zie pagina 1ØØ), blijven deze definities behouden wanneer u op de tekstverwerker overgaat. Om naar de oorspronkelijke tekens terug te keren, tikt u de woorden CLEAR FONT en slaat u 'enter' aan, alvorens de BASIC-modus te verlaten.

Nu moet u een willekeurig aantal letters aanslaan en op 'enter' drukken, zodra u vindt dat u genoeg letters hebt ingetikt. Nu ziet u de positie-wijzer gewoonweg naar de volgende regel gaan; er zal geen foutboodschap verschijnen met de mededeling dat u iets heeft ingetikt dat de computer niet begrijpt. Wanneer u de computer voor tekstverwerking gebruikt, gedraagt hij zich louter als een schrijfmachine met extra mogelijkheden — de computer geeft weer wat u getikt heeft, maar hij onderneemt niets om dat te begrijpen.

Bij tekstverwerking heeft de 'enter' toets dezelfde functie als de 'wagenterugloop' — maar u hoeft deze toets alleen maar aan het eind van een alinea aan te slaan, niet aan het eind van elke regel. Bedenk wel dat de computer gebruik maakt van een methode (de word-wrap-around methode: het zonder onderbreking doortypen wanneer de regel aan het eind van de pagina komt) waardoor de regels automatisch de vereiste lengte krijgen.

Merk op dat de tekst tussen twee wagenteruglopen (de 'enter' toets) als alinea telt. Functies die van toepassing zijn op hele alinea's (zie hieronder) zijn van invloed op de alinea waar de positie-wijzer zich op een gegeven ogenblik bevindt.

Veranderingen in de tekst op het scherm kunnen op precies dezelfde-manier aangebracht worden als veranderingen in een programma — zie beide tabellen aan het eind van het vorige hoofdstuk. De speciale functies die we nu nader gaan bekijken zijn vooral bestemd voor toepassing tijdens het tikken van een document, ofschoon ze ook functioneren wanneer BASIC regels ingevoerd worden.

SPECIALE FUNCTIES

Om gebruik te kunnen maken van een van deze functies moet u de juiste functietoets indrukken en tevens de 'ctrl' of de 'alt' (alteration = verandering) toets ingedrukt houden.

'CTRL' + FUNCTIE 1 — OPNIEUW INDELEN

Deze funtie regelt de lengte van de regels van de alinea waar de positie-wijzer zich bevindt. Dit is handig wanneer u verschillende woorden hebt tussengevoegd of weggehaald en de regels daardoor ongelijk zijn geworden.

'CTRL' + FUNCTIE 2 — CENTREREN

Hierdoor wordt een regel in het horizontale midden van de 'pagina' geplaatst. Handig voor titels en opschriften.

'ALT' + FUNCTIE 2 — ALLE TABS ONGEDAAN MAKEN

Evenals een schrijfmachine biedt de computer de mogelijkheid met een 'tab' toets te werken. We nemen aan dat u graag met een enkele aanslag alle tabstops

ongedaan wilt maken voordat u de tabulator opnieuw instelt.

'CTRL' + FUNCTIE 3 — INSTELLEN/ONGEDAAN MAKEN VAN TAB STOPS

Tab is de afkorting van 'tabellariseren'. Dit betekent dat het scherm om redenen van ordening in verschillende 'kolommen' verdeeld wordt. Wanneer u een tabel met informatie wilt invoeren (zoals bij voorbeeld de tabel op pagina 54-55), kunt u evenveel tabs nemen als er kolommen zijn in uw tabel. De 'tab' toets kan dan gebruikt worden om van de ene kolom naar de andere te springen. Hierdoor kunt u snel en accuraat keurige kolommen samenstellen.

De tabulator kan gewoonweg ingesteld worden door de positie-wijzer (door middel van het spelpookje) naar de plaats te brengen waar u uw tab stop wilt hebben; daarna moet u 'ctrl' en functietoets 3 indrukken. Mocht de tabulator al zijn ingesteld op deze hoogte, dan wordt dit door hetzelfde commando ongedaan gemaakt.

'ALT' + FUNCTIE 3 — FORMAATREGEL

De 'formaatregel' boven in het beeld geeft de linker en de rechter kantlijn aan; eveneens wordt aangegeven waar tab stops zijn aangebracht (door middel van verticale streepjes). Met dit commando kunt u de formaatregel in beeld brengen en weer weghalen.

'CTRL' + FUNCTIE 4 — LINKER KANTLIJN

Vanzelfsprekend wilt u van tijd tot tijd de kantlijnen veranderen. De linker kantlijn (daar waar de regels beginnen) bepaalt u door de positie-wijzer naar de plek te brengen waar u de kantlijn wilt hebben; druk vervolgens op functietoets 4 en 'ctrl'.

'ALT' + FUNCTIE 4 — RECHTER KANTLIJN

Als hierboven, maar dan voor de rechter kantlijn.

'CTRL' + FUNCTIE 5 — OPHEFFEN KANTLIJNEN

Dit commando maakt het mogelijk woorden of tekens in de marge aan te brengen. Het symbool (*) op de statusregel bevestigt dat het commando gegeven is; wilt u dit commando ongedaan maken, dan moet u dezelfde toetsen nogmaals indrukken.

'ALT' + FUNCTIE 5 — OPNIEUW INSTELLEN KANTLIJNEN

Hierdoor worden de kantlijnen naar de meest linkse en rechtse positie van de 'pagina' geschoven. Met dit commando kunnen ook de TABS opnieuw ingesteld worden. Wanneer u een smalle kolom met tekst in het midden van het scherm heeft ingetikt en u voor de volgende alinea een grotere bladspiegel nodig heeft, moet u 'alt' indrukken en functietoets 5 en vervolgens 'enter'. Nu kunt u de kantlijnen geheel naar believen opnieuw instellen.

'CTRL' + FUNCTIE 6 — OMHOOGGAAN

Bij toepassing van dit commando gaat de alinea waar de positie-wijzer zich bevindt een regel omhoog; de regel die zich eerst boven de alinea bevond komt nu daaronder te staan. Dit commando kan herhaald worden totdat u de alinea op de gewenste plaats hebt.

'ALT' + FUNCTIE 6 — OMLAAGGAAN

Het omgekeerde van hierboven.

'CTRL' + FUNCTIE 7 — DE KLEUR VAN DE REGEL VERANDEREN

Hiermee kiest u voor de regel waar de positie-wijzer zich bevindt, een nieuw kleurenpaar voor tekst en achtergrond. Op een scherm met 8Ø kolommen heeft u de keuze uit vier kleurenparen; door telkens dit commando te geven krijgt u die kleuren in beeld. Op een scherm met 4Ø kolommen zijn slechts twee kleurenparen.

'ALT' + FUNCTIE 7 — DE KLEUR VAN DE ALINEA VERANDEREN

Als hierboven, maar nu van toepassing op een hele alinea.

De toepassingsmogelijkheden van de functietoetsen op zich (zonder 'ctrl' of 'alt') verschillen, naargelang u de computer voor programmeren of voor tekstverwerken gebruikt. Merk op dat we in sommige gevallen met 'faalveilige' apparaten te maken hebben — wanneer u bij voorbeeld bij gebruik van de tekstverwerker om wilt schakelen naar een scherm met 8Ø kolommen, moet u functietoets 5 aanslaan en daarna

wordt u gevraagd, of u dat wilt bevestigen door 'enter' in te drukken (of herroepen met de 'esc' toets). Zo wordt voorkomen dat u een document wist door abusievelijk de funtietoets aan de slaan.

Zie voor een compleet overzicht van de toepassingsmogelijkheden van de functietoetsen pp. 54 en 55.

HET AFDRUKKEN VAN TEKST

Nu u een en ander heeft ingevoerd zult u uw tekst waarschijnlijk wel willen afdrukken (vooropgesteld dat u een printer heeft) of op cassette vastleggen voor later gebruik.

Wanneer u gebruik wilt maken van de printer, moet deze natuurlijk op de computer aangesloten zijn (via de 'printer' sok aan de achterkant van de machine). De printer moet ingeschakeld worden en direct gekoppeld zijn aan de centrale verwerkingseenheid. Druk nu op functietoets 5. Een boodschap op het scherm zal u eraan herinneren, na te gaan of de printer correct is aangesloten. Druk op 'enter' en uw document wordt afgedrukt.

HET OPSLAAN VAN TEKST OP CASSETTE

De opslag van tekst op cassette is nagenoeg indentiek aan de opslag van een programma — zie pp. 56-57 voor de manier waarop een cassetterecorder op de computer aangesloten wordt. Evenals een programma moet een document dat u wilt opslaan een naam krijgen, en wel volgens de regels op p. 56. Het enige verschil is hierin gelegen dat de naam van het document — de 'bestandsnaam' — niet tussen aanhalingstekens getikt wordt.

Druk op functietoets 2. De computer zal u nu vragen de bestandsnaam in te voeren. Nadat u (bij) voorbeeld) LETTER heeft getikt, moet u de opneem — en afspeeltoetsen knop op de cassetterecorder indrukken en daarna 'enter' aanslaan. Uw tekst zal nu via afstandsbesturing opgeslagen worden, vooropgesteld dat de aansluiting klopt.

Eveneens biedt de tekstverwerker de mogelijkheid een document te laden, zoals u een programma zou laden (maar terwijl de band teruggespoeld wordt, moet u wel de plug uit de REM sok halen). U hoeft alleen maar functietoets 1 in te drukken, LETTER te tikken — of welke naam ook — en 'enter' aan te slaan.

VAN TEKSTVERWERKEN OP PROGRAMMEREN OVERSCHAKELEN

Wanneer u wilt stoppen met de tekstverwerker en weer wilt programmeren in BASIC, moet u functietoets 8 indrukken. De computer zal u nu de volgende aanwijzing geven "Press ENTER to start BASIC". Doe dit en u komt terug in BASIC.

Door WP te tikken schakelt u weer om naar de tekstverwerker en wordt uw laatste tekst gewist. Wanneer u functietoets 8 per ongeluk heeft ingedrukt, kunt u dit via 'esc' ongedaan maken.

Een aantal functietoetsen, of allemaal, heeft u al geprobeerd. Ze zijn van 1-8 genummerd en bevinden zich boven de cijfertoetsen.

Ze kunnen opnieuw gedefinieerd worden en elke denkbare functie krijgen. Maar zolang de toetsen niet opnieuw gedefinieerd zijn voorziet de computer elke toets van een functie die u tot op zekere hoogte wel van pas zal komen.

De functietoetsen kunt u opnieuw definieren door (bij voorbeeld) het volgende in te tikken:

```
SET FKEY 1 "PRINT"
```

De functie die de toets moet krijgen komt tussen aanhalings-tekens te staan. Het cijfer achter FKEY is het nummer van de toets die opnieuw gedefinieerd wordt.

Probeer bovenstaand commando maar in te tikken en druk vervolgens op de eerste funtietoets. Het woord PRINT zal op het scherm verschijnen. Dit betekent dat de toetsen gebruikt kunnen worden om veel voorkomende sleutelwoorden op het scherm te zetten via een enkele aanslag; zo hoeft u niet meer het hele woord in te tikken. Woorden als RUN, LIST en RENUMBER komen waarschijnlijk eerder in aanmerking, omdat het niet altijd nodig is iets erachter te zetten. Het uitlijsten van een programma via een functietoets gaat veel sneller dan wanneer het woord LIST ingetikt wordt en daarna 'enter' ingedrukt.

Een wagen 'terugloop' — hetzelfde als 'enter' indrukken — kan aan de definitie van een functietoets toegevoegd worden via de volgende aanvulling.

```
&CHR$(13)
```

op de desbetreffende definitie. Bij voorbeeld:

```
SET FKEY 1 "PRINT" &CHR$(13)
```

De grote tabel op de volgende bladzijde laat u de toepassingsmogelijkheden van elke functietoets zien, zoals we die tegenkomen wanneer u de computer inschakelt en de BASIC cartridge aangesloten is. Merk op dat u in combinatie met de functietoetsen de toetsen 'shift' 'ctrl' en 'alt' kunt gebruiken, wat inhoudt dat elke functietoets eigenlijk vier verschillende functies heeft.

Wanneer u de functietoetsen, nadat u ze opnieuw heeft gedefinieerd, weer de oorspronkelijke functie terug wilt geven moet u CLEAR FKEYS intikken.

Zoals u weet, hebben de functietoetsen op zich verschillende toepassingen, wanneer u met de tekstverwerker werkt. Hiervan krijgt u een overzicht in de kleine tabel.

WAT BEWERKT ELKE AFZONDERLIJKE FUNCTIETOETS?

	Alleen BASIC		BASIC of tekstverwerker	
Toets	**NORMAAL**	**+ SHIFT**	**+ CTRL**	**+ ALT**
1	START Programma gaat lopen. Indien geen programma, dan wordt programma vanaf diskette in geheugen geladen en gedraaid. Indien geen diskette, dan wordt vanaf cassette geladen.	VOORTGAAN Hierdoor gaat programma verder na 'stop' commando.	OPNIEUW INDELEN Regelt lengte van regels zodat een en ander binnen de marges blijft (fatsoeneert alinea na redigeren).	JUSTEREN EN OPNIEUW INDELEN Spaties in regels worden gelijkelijk verdeeld, de kantlijnen gelijkmatig gehouden.
2	UITLIJSTEN Het hele programma wordt uitgelijst.	UITLIJSTEN Het volledige programma wordt op de printer uitgelijst.	CENTREREN Plaatst tekst in het midden van beeld.	ALLE TABS ONGEDAAN MAKEN
3	AUTO Automatische nummering van 10 per keer. Druk op 'stop' om 'uit te schakelen'.	OPNIEUW NUMMEREN Het volledige programma wordt opnieuw genummerd, telkens met 10 oplopend.	TAB INSTELLEN/ ONGEDAAN MAKEN Waar de positie-wijzer zich bevindt wordt tab stop ingesteld of ongedaan gemaakt.	FORMAAT-REGEL De regel die de marges en tab stops aangeeft gaat aan en uit.
4	CASS REC REM 1 Schakelt besturingsknop van station 1 in of uit.	CASS REC REM 2 Als hiernaast, maar nu voor contact van station 2.	LINKER KANTLIJN Waar positie-wijzer zich bevindt komt linker kantlijn.	RECHTER KANTLIJN Als hiernaast, maar nu voor rechter kantlijn.
5	TEKST Stelt het volledige scherm beschikbaar voor tekst; zowel de tekstuele als de grafische pagina worden op blank gesteld.	WEERGAVE TEKST Schakelt over van de grafische naar de tekstuele pagina zonder een van beide te wissen.	OPHEFFEN KANTLIJNEN Maakt het mogelijk tekens in de marge aan te brengen.	KANTLIJNEN OPNIEUW INSTELLEN marges worden zo ingesteld dat de volle breedte van het scherm benut kan worden.
6	GRAFISCHE GEVENS-VERWERKING De eerste 20 regels voor grafische gegevens-verwerking, de laatste 4 regels voor tekst; stelt zowel de tekstuele als de grafische pagina op nul.	WEERGAVE GRAFISCHE GEGEVENS Overschakelen van tekstuele naar grafische weergave zonder een van beide te wissen.	OMHOOGGAAN Plaatst de alinea boven de regel die eraan voorafgaat.	OMLAAGGAAN Plaatst de alinea onder de regel die erop volgt.
7	KLIK Zet de toetsen-klik (telkens te horen wanneer u een toets aanslaat) aan of uit.	LUIDSPREKER Het volledige geluids-vermogen wordt uitgeschakeld (of weer ingeschakeld).	DE KLEUR VAN DE REGEL VERANDEREN	DE KLEUR VAN DE ALINEA VERANDEREN
8	INFO Verstrekt informatie over programma's in het geheugen.	TYPEN Brengt u in de tekst-verwerkings-modus.		

'Normale' functies (alleen van toepassing i.g.v. tekstverwerking)

1. LADEN Tekst kan vanaf cassette of diskette geladen worden.	2. OPSLAAN Tekst kan op cassette of diskette opgeslagen worden.	3. PRINTEN Stuurt tekst naar de printer.	4. HULP Brengt informatie in beeld over de verschillende tekst-verwerkings-functies.
5. TEKST 80 Wist tekst en stelt een scherm met 80 kolommen ter beschikking.	6. TEKST 40 Wist tekst en stelt een scherm met 40 kolommen ter beschikking.	7. KLIK Schakelt toetsenklik in of uit.	8. EXIT Stopt met tekst-verwerken; nu kunt u terugkeren naar BASIC etc.

In de gebruiksaanwijzing met installatie-aanwijzingen heeft u gezien hoe een cassette recorder gebruikt moet worden om een programma in het geheugen van de computer te laden. Maar u kunt een cassette recorder ook gebruiken om uw eigen programma's op te slaan.

Misschien wilt u wel een programma vastleggen dat u al uit een eerder gedeelte van deze handleiding hebt overgenomen — wellicht heeft u er zelf al het een en ander aan toegevoegd; we gaan er dus van uit dat er al een programma in het geheugen van de computer zit op het moment dat u dit gaat lezen.

Zorg er eerst voor dat de computer en de cassette recorder correct met elkaar zijn verbonden. Via de OUT-uitgang aan de achterkant van de machine worden programma's op cassette vastgelegd. Steek een van de grotere cassettestekkers in deze uitgang.

Steek vervolgens het andere uiteinde van het snoer in de MIC-ingang — of iets dergelijks — van de tape recorder.

Steek nu een van de kleine stekkers in de REM2-uitgang en het andere uiteinde daarvan in de kleine ingang van de recorder (waarschijnlijk met REM aangeduid) — vooropgesteld dat die ingang op uw recorder zit.

Het maakt niets uit of de computer in- of uitgeschakeld is wanneer u hiermee bezig bent!

Hierna moet uw programma een naam krijgen. Voor alle duidelijkheid nemen we aan dat u het programma 'mijnprogr' noemt. De naam mag uit maximaal achtentwintig tekens bestaan en kan letters, cijfers en de volgende interpunctietekens omvatten: '.', '»', '_' '/' en '/'. Desgewenst zou u het programma 'Mijn-programma-nummer-1' kunnen noemen. Tik in:

```
SAVE "mijnprogr"
```

Druk op de opneem- en afspeeltoetsen knop en vervolgens op 'enter'. U krijgt nu de volgende boodschap te zien:

```
SAVING mijnprogr
```

en daarna:

```
OK _
```

zodra het programma is vastgelegd. Omdat de cassetterecorder ingeschakeld is, zal de Enterprise het bandje automatisch op het juiste moment starten en stoppen — vooropgesteld dat uw recorder op afstand bestuurd kan worden.

Terwijl het programma op band opgenomen wordt, geeft de computer het geluid van deze operatie zachtjes via de ingebouwde luidspreker weer. Draai het geluid van de TV weg als u daar laast van hebt (SET TAPE SOUND OFF), maar waarschijnlijk zal dat voor u eerder een bevestiging zijn dat het opnemen goed verloopt.

CONTROLE

Het spreekt vanzelf dat u wilt controleren of het programma correct is opgenomen. Hiervoor gebruikt u het commando VERIFY (verifieren).

Spoel het bandje eerst terug naar de plaats waar u bent begonnen met het vastleggen van het programma. Als u gebruik heeft gemaakt van afstandsbediening, zal de computer het terugspoelen op uw recorder waarschijnlijk onmogelijk maken. Is dit het geval, tik dan TOGGLE REM1 in of druk op functie toets 4. Tik vervolgens

```
VERIFY "mijnprogr"
```

in, start het bandje en druk op 'enter'. Ook nu weer bestuurt de Enterprise de tape recorder.

Wanneer het programma zonder fouten is geladen, zal het volgende op het scherm verschijnen:

```
ok
```

Zoniet, dan verschijnt er een foutboodschap. In dit geval moet u de aansluitingen nagaan, controleren of het volume goed is geregeld en het opnameproces nogmaals doorlopen.

PROGRAMMA'S SAMENVOEGEN

Wanneer er al een programma in de computer zit en u een tweede eraan toe wilt voegen, moet u het commando MERGE (samenvoegen) gebruiken.

MERGE kan van pas komen wanneer u halverwege een lang programma bent dat u als een reeks subprogramma's hebt vastgelegd (kleinere programma's die deel uitmaken van een groter

programma). Wanneer u alle afzonderlijke delen die u op band hebt vastgelegd tot een volledig programma wilt samenvoegen, kunt u het commando MERGE toepassen.

U moet er wel op toezien dat alle regelnummers verschillend zijn wanneer u twee programma's wilt samenvoegen. De reden hiervoor is deze:

laten we eens aannemen dat er een klein programma in het geheugen zit. De regelnummers daarvan zijn 100, 110, 140 en 160. Stel, dat u met dit programma een ander klein programma wilt samenvoegen dat op band staat. De regelnummers hiervan zijn 140, 160, 180 en 200.

Wanneer u deze twee programma's probeert samen te voegen zonder het programma in het geheugen opnieuw te nummeren, worden de regels 140 en 160 in het geheugen vervangen door regel 140 en 160 van het programma op band. Het gevolg zou zijn dat u twee regels van uw oorspronkelijke programma kwijt bent.

Dus wanneer u twee programma's samenvoegt, worden de regels van elk programma zo 'ineengevlochten' dat u een groter programma krijgt.

Hieronder volgt hoe een programma dat op band staat met een ander samengevoegd kan worden.

De methode met de tape recorder is precies hetzelfde als die voor het laden (MERGEn is hetzelfde als laden, alleen wordt door het laden van een programma alles vervangen dat zich op dat moment in het geheugen bevindt — het programma dat al aanwezig is wordt vervangen door het nieuwe).

Als u niet meer zeker weet hoe een programma geladen moet worden, raadpleeg dan pagina 10 van de installatie-aanwijzingen.

Zie er dus op toe dat uw bandje op het begin van het programma staat dat u wilt tussenvoegen; vergewis u ook ervan dat alle regelnummers kloppen. Tik vervolgens het volgende in:

```
MERGE "nieuwprogr"
```

(nieuwprogr is de naam van het programma - de naam waaronder het aanvankelijk was opgenomen).

Start het bandje en druk op 'enter'. Nu worden de programma's samengevoegd. Doen!

BESTANDEN

Programma's die op band zijn opgeslagen kunnen beschouwd worden als bestanden. Dit geldt eveneens voor diskettes, het andere medium voor de opslag van programma's. Een 'bestand' kan ook een verzameling gegevens zijn voor gebruik door een computer. Dit wordt een gegevensbestand genoemd.

DISKETTES

Wanneer diskettes op de computer zijn aangesloten, kunnen alle commando's voor de tape recorder gebruikt worden, maar in plaats van de recorder worden automatisch de diskettes gebruikt. Alle operaties, zoals opnemen, laden, enz., zullen dan veel sneller verlopen.

In grote lijnen zijn strings al aan de orde gekomen. Hier zullen we een en ander in het juiste kader plaatsen.

Strings (tekst) zijn een van de twee belangrijkste soorten informatie — getallen zijn de andere soort — waar de computer mee kan werken. In feite moet u zich een computer niet anders voorstellen dan een informatieverwerkend apparaat. Waarschijnlijk is het u inmiddels duidelijk geworden dat een computer niet in staat is te denken. Hij doet precies dat wat u hem opdraagt, aan de hand van gegevens die u hem verstrekt. Programmeren is dus in wezen alleen maar een manier om gegevens te sorteren, te verzamelen en te manipuleren.

In een eerder stadium hebben we strings vergeleken met de directe rede. Aanhalingstekens worden gebruikt wanneer iemands woorden precies aangehaald worden; maar het is niet noodzakelijk dat de woorden tussen aanhalingstekens begrepen worden — ze zouden volstrekt betekenisloos kunnen zijn.

Dit is de essentiele gedachte achter strings. De computer begrijpt niet wat strings zijn of wat ze betekenen. De computer beschouwt ze gewoon als betekenisloze symbolen.

Alles in een programma tussen dubbele aanhalingstekens zal door de computer beschouwd worden als een string. Hier volgt een voorbeeld:

"Hoe oud bent u?"

Dezelfde tekens zouden in een boek kunnen staan als directe rede:

"Hoe oud bent u?", vroeg Jim.

Onthoud ook dat u getallen tussen aanhalingstekens kunt plaatsen:

"Ik ben 22", antwoordde Sally.

Probeert u deze voorbeelden maar eens:

```
PRINT 2*2
```

Dit voorbeeld laat de computer een klein sommetje maken: breng de waarde van 2 vermenigvuldigd met

de waarde van 2 in beeld, m.a.w. maak de som en breng de uitkomst in beeld.

```
PRINT "2*2"
```

Dit voorbeeld is bijna precies hetzelfde als het vorige. maar door de aanhalingstekens wordt 2*2 een string. Nu laten we de computer weten eerst een 2, vervolgens * en dan weer een 2 op het scherm te zetten.

Als u daarom vraagt kan de computer u vertellen hoe lang een string is. Probeer dit programma maar eens:

```
100     LET A$ = "COMPUTER"
110     LET B$ = "MICROCOMPUTER"
120     !
130     !     Regel 100 en 110 benoemen twee string-
140     !     variabelen, A$ en B$. String-variabelen
150     !     zijn hetzelfde als numerieke variabelen,
160     !     behalve dat ze een naam zijn voor een
170     !     aantal symbolen en niet voor de waarde
180     !     van getallen. De naam die aan een string-
190     !     variabele gegeven wordt moet altijd het
210     !     dollarteken ($) als laatste teken hebben.
215     !
220     LET A = LEN(A$)
230     LET B = LEN(B$)
240     !
250     !     A en B zijn numerieke variabelen die de
260     !     lengte van beide strings bevatten. LEN
270     !     (length = lengte) laat u het aantal tekens
280     !     in een string weten (d.w.z. de lengte
290     !     ervan).
300     !
310     PRINT "String A$ is";A;
        "tekens lang."
320     PRINT "String B$ is";B;
        "tekens lang."
330     END
```

Probeert u de inhoud van A$ en B$ maar te veranderen — de computer kan u altijd meedelen hoeveel symbolen elke string-variabele bevat.

SPATIES

Een spatie betekent misschien niets voor u en dat is

heel begrijpelijk. Maar de computer beschouwt een spatie als een symbool. Verander A$ en B$ (in het bovenstaande voorbeeld) maar in:

" " en " "

U zult zien dat de computer u vertelt hoeveel spaties elke variabele bevat, ook al heeft het de schijn dat beide leeg zijn. Een echt lege string zou er als volgt uitzien:

""

Zoals u kunt zien bevat deze variabele zelfs geen spaties. Dit wordt een *lege string* genoemd.

STRINGS BINNEN ANDERE STRINGS

De Enterprise kan enkele interessante dingen doen met strings. Zo kunt u bij voorbeeld van de ene string een andere maken. Hier volgt een programma dat zo iets bewerkstelligt:

```
100   LET BIGSTRING$ = "slechts een paar
      woorden"
110   !
120   !   Regel 100 benoemt een string-variabele.
130   !
140   LET SMALLSTRING$ = BIGSTRING$(9:17)
150   !
160   !   Regel 140 benoemt een andere string-
170   !   variabele maar met een verschil: deze
180   !   variabele begint met het negende teken
190   !   van BIGSTRING$ en eindigt met het
200   !   zeventiende teken.
210   !
220   PRINT BIGSTRING$
230   PRINT SMALLSTRING$
240   END
```

De computer kopieert letterlijk enkele tekens van de ene variabele en stopt ze in een aparte andere variabele. Deze aparte variabele wordt een substring genoemd. Om van de ene string een andere te maken hebben we de substring een naam gegeven — SMALLSTRING$ — en vervolgens het teken

gespecificeerd (tussen haakjes achter de naam van de grote string) waarmee we de substring willen laten beginnen en het teken waarmee de string moet eindigen.

Een opdracht als LET INITIAL$=NAME$(1:1) luidt dus als volgt: 'Roep de substring INITIAL$ op en laat deze de eerste letter van NAME$ bevatten'.

In al deze voorbeelden hebben we onze substring benoemd als een variabele op zich door haar een eigen naam te geven (een 'identificatiesymbool'). Maar dit is niet strikt noodzakelijk. In het bovenstaande programma zouden we regel 140 kunnen wissen en regel 230 kunnen veranderen in:

```
230     PRINT BIGSTRING$ (9:17)
```

.. hetgeen betekent: 'druk het negende tot en met het zeventiende teken af van de string met de naam BIGSTRING$'. De methode waarvoor u kiest zal afhangen van de reden waarom u gebruik maakt van substrings. Wanneer u dezelfde substring herhaaldelijk wilt gebruiken is het vaak handiger om er een aparte variabele van te maken. Zoniet, dan kunt u volstaan met bovenstaande methode.

INKEY$

INKEY$ is een uiterst nuttige en belangrijke string-functie. (Elk BASIC woord dat van invloed is op een string wordt een string-functie genoemd; de meeste hiervan hebben als laatste teken het dollarteken, waarmee aangegeven wordt dat het resultaat eveneens een string zal zijn.)

INKEY$ biedt u de mogelijkheid een toets aan te slaan terwijl het programma uitgevoerd wordt, teneinde het verloop van het programma te beinvloeden. Het heeft wel iets weg van een INPUT, maar de verschillen zullen al snel duidelijk worden. Probeer dit progamma maar eens:

```
100     PRINT "Heeft u dit hoofdstuk begrepen?"
110     PRINT
120     PRINT "Antwoord met j of n"
130     DO
140         LET A$ = INKEY$
150     LOOP UNTIL A$< >" "
160     !
```

```
170   !    Regel 140 stopt het resultaat van een
180   !    toets-aanslag in een string-variabele. U
190   !    zult nog wel zien waarom dit belangrijk is.
200   !    ------------------------------------------
210   IF A$ = "j" THEN
220         PRINT "Laten we hopen dat u gelijk
            hebt."
230   ELSE IF A$ = "n" THEN
240         PRINT "Lees het dan nog maar eens
            door."
250   ELSE
260         PRINT "U hebt een verkeerde toets
            aangeslagen."
270   END IF
280   !    ------------------------------------------
290   !    IF/THEN opdrachten bent u al eerder
300   !    tegengekomen. Het doel van ELSE 230
310   !    en 250 zal wel in voldoende mate blijken
320   !    uit de gewone betekenis van het woord.
330   !    'Voorwaardelijke' opdrachten als deze
340   !    komen ruimschoots aan de orde in het
350   !    hoofdstuk "Beslissingen".
355   !    ------------------------------------------
360   END
```

Wanneer het programma zijn vraag stelt, geeft uw volgende toetsaanslag het antwoord en laat de computer naar regel 210 en verder gaan.

In tegenstelling tot een INPUT opdracht verzoekt de INKEY$ functie u alleen maar een toets aan te slaan en u hoeft niet 'enter' in te drukken. (INKEY$ is dus uiterst geschikt voor spelletjes — voor antwoorden die slechts een letter vereisen — of voor hervatting van een functie die halverwege door een lus is onderbroken.) Onthoud dat 'shift' in combinatie met een andere toets slechts als een aanslag telt.

Een ander verschil tussen INKEY$ en INPUT is dat INKEY$ de computer niet automatisch laat wachten totdat u uw antwoord hebt gegeven. Daarom zijn regel 130 en 150 in het programma hierboven nodig. Schrapt u ze maar eens. U zult zien dat het programma dan in een fractie van een seconde helemaal uitgevoerd wordt en dat u geen tijd hebt om uw antwoord in te tikken. Zodra de computer bij regel 140 aankomt, beslist hij dat INKEY$ identiek is aan een lege string (er is immers

geen toets aangeslagen); ook A$ wordt dus een lege string.

Maar regel 13Ø en 15Ø luiden als volgt:
'blijf de lus doorlopen totdat A$ iets anders wordt dan een lege string — ga dan verder met de rest van het programma'.

Onthoud dat INKEY$ te vergelijken is met een variabele waarvan de waarde zeer snel verandert. Dit is het geval: de computer 'kijkt' ongeveer vijftig keer per seconde op het toetsenbord om aanslagen van toetsen te registreren; als u (bij voorbeeld) de 'a' toets aanslaat, is INKEY$ gelijk aan 'a' — maar wel slechts voor heel even. Normaal gesproken zal INKEY$ eenvijftigste seconde later weer in een lege string veranderen en deze waarde behouden tot de volgende aanslag.

Dit verklaart waarom regel 14Ø nodig is in het programma. Hierdoor wordt het resultaat van uw aanslag meteen in een aparte string-variabele gestopt die niet zal veranderen, zolang de computer het resultaat onderzoekt (kijkt of het aan bepaalde voorwaarden voldoet — zie hiervoor regel 21Ø-27Ø).

U zou regel 14Ø kunnen schrappen en regel 15Ø, 21Ø en 23Ø kunnen veranderen in:

```
15Ø     LOOP UNTIL INKEY$< >" "
21Ø     IN INKEY$ = "j" THEN
23Ø     ELSE IF INKEY$ = "n" THEN
```

Het programma kan niet meer goed functioneren omdat de waarde van INKEY$ tussen regel 15Ø en regel 21Ø verandert (de waarde wordt " ").

Het is natuurlijk overbodig om de waarde van INKEY$ in een afzonderlijke variabele te stoppen, als u er in de volgende programmaregels geen gebruik meer van maakt. Als u bij voorbeeld alleen maar het verloop van een programma wilt onderbreken, totdat u opdracht geeft om weer verder te gaan, kunt u iets dergelijks als hieronder tikken:

```
1ØØ     PRINT "Elke toets is goed om verder te gaan."
11Ø     DO
12Ø     LOOP UNTIL INKEY$ < >" "
```

UCASE$ EN LCASE$

Wat ook interessant is, is dat u strings in hoofdletters of

in kleine letters kunt veranderen.

De twee woorden die dit bewerkstelligen zijn UCASE$ en LCASE$. Dit programma laat u zien hoe dat in zijn werk gaat:

```
100   !
110   !   Dit programma zal een door u getikte
120   !   string eerst in hoofdletters veranderen en
130   !   daarna in kleine letters.
140   !
150   INPUT PROMPT "Sla a.u.b. enkele letters:": A$
160   PRINT A$
170   PRINT UCASE$ (A$)
180   PRINT LCASE$ (A$)
190   END
```

VAL

Het volgende behoort ook tot de mogelijkheden: u kunt een string die cijfertekens bevat veranderen in het getal dat de string zou zijn als er geen sprake was van een string. Het BASIC woord dat dit bewerkstelligt is VAL (afkorting van 'value' = waarde). Probeer dit maar eens:

```
100   INPUT PROMPT "Sla a.u.b. enkele tekens aan:
      ":A$
110   PRINT A$
120   PRINT VAL(A$)
130   END
```

Bij het bepalen van VAL (A$) komen alleen de cijfers in aanmerking die voor de eerste letter staan. VAL ("123AB45") is dus gelijk aan 123 en VAL ("AB12345") is gelijk aan Ø.

U kunt ook het tegenovergestelde doen: een getal in een string veranderen door gebruik te maken van het woord STR$.

STRINGS SAMENVOEGEN

We hebben al gezien hoe een gedeelte van een string in een substring ondergebracht kan worden. Nieuwe strings kunt u ook vormen door strings (of gedeeltes daarvan) met elkaar te verbinden. Strings met elkaar verbinden wordt aaneenschakelen genoemd; het enige dat u moet doen is het & teken tussen de rijen plaatsen — waarmee u zegt: 'de ene en de andere string'.

In het volgende voorbeeld wordt behalve van

substrings en de LEN functie ook van aaneenschakeling gebruik gemaakt:

```
100     INPUT PROMPT "Tik a.u.b. een woord:
        ":STRING$
120     CLEAR SCREEN
130     LET ABREV$ = STRING$ (1:1) & STRING$
            (LEN(STRING$): LEN(STRING$))&"."
135     !  ------------------------------------
140     !  In regel 130 maakt STRING$(1:1) van het
150     !  eerste teken van STRING$ een substring.
160     !  LEN(STRING$) geeft het aantal tekens
170     !  die STRING$ bevat; STRING$
180     !  (LEN(STRING$):LEN(STRING$)) maakt
190     !  dus van het laatste teken ook een
200     !  substring. Daarna worden beide
210     !  substrings met het '&' symbool met elkaar
220     !  verbonden aan een '.' gekoppeld. Aldus
230     !  wordt een nieuwe string-variabele
240     !  verkregen: ABBREV$ (abbreviation =
245     !  afkorting).
250     !  ------------------------------------
260     PRINT STRING$; "is afgekort"; ABREV$
270     END
```

U zou het op deze manier doen als u de afkorting herhaaldelijk zou gebruiken. Maar als u deze slechts een keer zou aanwenden, zou u regel 130 uiteraard kunnen schrappen en regel 260 veranderen in:

```
260     PRINT STRING$ (1:1) &STRING$
        (LEN(STRING$): LEN(STRING$))&"."
```

In dit geval zou u de & tekens kunnen vervangen door komma-punten.

We zijn al een aantal korte programma's tegengekomen die gebruik maakten van een lus om de computer zichzelf te laten herhalen. Wanneer de machine een bewerking meerdere malen achter elkaar moet uitvoeren, is het aanmerkelijk eenvoudiger hiervoor een lus te gebruiken dan steeds opnieuw hetzelfde stukje BASIC in te tikken.

Naar zal blijken is een lus ook een gemakkelijke manier om een volledig programma te besturen.

Afgezien van een uitzondering begint elke lus met een speciale opdracht en eindigt met een regel die de computer laat weten weer naar het begin te springen.

DO/LOOPS

Laten we met het gemakkelijkste type lus beginnen, de DO/LOOP. Deze bent u al tegengekomen en ze zien er als volgt uit:

```
 8Ø    INPUT PROMPT "Tik a.u.b. een getal in en ik
       zal er de 'tafel' voor afdrukken. ":A
 9Ø    LET B = A
1ØØ    DO !        Begin van DO/LOOP
12Ø      LET B = B + A
13Ø      PRINT B,
14Ø      !  ------------------------------------
15Ø      !        In regel 9Ø neemt een andere
155      !        variabele, B, waarde van uw getal
16Ø      !        aan. Vervolgens wordt in regel 12Ø
17Ø      !        en 13Ø de waarde van A (die niet is
175      !        veranderd) opgeteld bij die van B
18Ø      !        en de nieuwe waarde van B in
185      !        beeld gebracht. Telkens wanneer
19Ø      !        de lus doorlopen wordt gebeurt dit;
2ØØ      !        B neemt dus telkens toe met de
21Ø      !        waarde van A.
215      !  ------------------------------------
22Ø    LOOP UNTIL B > 15Ø ! Einde DO/LOOP
23Ø      !  ------------------------------------
24Ø      !        UNTIL B > 15Ø stopt het
241      !        programma. Zonder deze regel
242      !        gaat het eeuwig door wanneer u
243      !        niet tussenbeide komt (d.m.v. de
27Ø      !        'stop' of de 'hold' toets) UNTIL B>
28Ø      !        betekent 'tot B groter is dan 15Ø'.
285      !  ------------------------------------
29Ø    END
```

Zoals u kunt zien laat een DO/LOOP de computer in het rond gaan totdat aan een bepaalde voorwaarde is voldaan die de computer opdraagt te stoppen. Deze voorwaarde kan nader gespecificeerd worden door gebruik te maken van UNTIL of WHILE.

Afhankelijk van het woord dat u gebruikt heeft u de keus uit DO WHILE (OF DO UNTIL) en LOOP WHILE (of LOOP UNTIL). Vaak zijn beide alternatieven mogelijk, met name in een programma als het bovenstaande dat niet aangewezen is op een uiterst zorgvuldige rangschikking van de voorwaarden.

Het essentiele verschil tussen een voorwaarde aan het begin en een aan het einde van een lus is, dat dit van invloed is op het aantal keren dat een lus doorlopen wordt (een verschil van 1). Staat de voorwaarde aan het begin, dan wordt deze getest voordat de lus doorlopen zal worden. Dit komt hierop neer dat de lus niet doorlopen wordt, als er meteen aan de voorwaarde is voldaan. Staat de voorwaarde aan het eind, dan kan de voorwaarde pas aan het einde door de computer gelezen worden, wat inhoudt dat de lus altijd minstens een keer doorlopen wordt. WHILE en UNTIL zijn twee tegengestelde soorten voorwaarden. UNTIL B = 15Ø en WHILE B = 15Ø betekenen twee totaal verschillenden dingen (UNTIL betekent 'zolang dat niet het geval is' en WHILE 'zolang dat het geval is').

Probeer bovenstaande lus maar te wijzigen door UNTIL te vervangen door WHILE, grotere en kleinere getallen aan te wenden als antwoord op de INPUT PROMPT aan het begin en de voorwaarde achter DO in plaats van achter LOOP te plaatsen. Enig geexperimenteer zal u snel duidelijk maken wat u met een DO/LOOP kunt bewerkstelligen.

ERUIT STAPPEN

Op elk gewenst moment kunt u ervoor zorgen dat de machine uit een lus stapt door gebruik te maken van het woord EXIT. Welnu, EXIT is eveneens van toepassing op de FOR/NEXT lussen (deze komen hierna aan de orde). Het is dus nodig nader aan te geven welk soort lus u wilt beeindigen — de wijze waarop dat gedaan wordt is EXIT FOR of EXIT DO. Wanneer u een lus beeindigt zou u een of andere voorwaarde moeten gebruiken — bij voorbeeld:

```
1000    IF X > 25 THEN EXIT DO
```

Een vluchtige blik op het hoofdstuk over beslissingen zal u vertellen hoe u dit tevens kunt doen door gebruik te maken van SELECT CASE.

Onthoud dat het woord EXIT de enige juiste manier is om een lus voor het eigenlijke einde ervan te beeindigen. Gebruik voor dit doel nooit woorden als GOTO — deze kunnen zowel voor uzelf als voor de computer verwarrend zijn. (GOTO en het daarmee samenhangende commando GOSUB komen aan de orde op pagina 146-147.) EXIT bewerkstelligt alleen maar dat de computer naar de eerste regel na het einde van de lus springt.

FOR/NEXT LUSSEN

De FOR/NEXT lus verschilt nogal van de DO/LOOP. Mogelijk is de FOR/NEXT lus iets minder direct maar de toepassing is ondubbelzinnig.

Hieronder volgt een korte FOR/NEXT lus:

```
100     INPUT PROMPT "Hoe vaak moet de lus
        doorlopen worden?":X
110     !
120     !       Merk op dat in deze en andere
125     !       INPUT PROMPTs aan het eind van
130     !       de 'prompt' zelf een spatie is
140     !       aangebracht. Dit komt het beeld ten
145     !       goede.
150     !
160     FOR P = 1 TO X
170        PRINT "LOOP THE LOOP!"
180     NEXT P
190     END
```

Ziet u dat het getal dat u intikt het aantal keren aangeeft dat de computer de lus doorloopt? Dit is het belangrijkste doel van een FOR/NEXT lus — u kunt kort maar bondig aangeven hoe vaak de lus doorlopen moet worden. Voorwaarden hoeft u niet te gebruiken behalve in speciale situaties.

De FOR/NEXT lus kan ook met telkens drie, twintig, 0.2, 1000, 466.666 optellen en zelfs aftrekken. Het woordje STEP maakt dit alles mogelijk. Hieronder volgt een programma dat aftelt met telkens twee.

```
1Ø      FOR P = 4Ø TO Ø STEP -2
2Ø          PRINT P,
3Ø      NEXT P
4Ø      END
```

U kunt de machine niet zomaar vragen om van 4Ø tot Ø af te tellen zonder het commando STEP te geven. Dit gebeurt pas wanneer de machine weet dat er afgetrokken moet worden (negatieve getallen erbij optellen) van het getal in de FOR regel. Als u niet nader specificeert door gebruik te maken van STEP, zal een FOR/NEXT lus altijd met telkens 1 optellen. Neem ook nota van de PRINT P opdracht. De eerste regel van de lus creeert een variabele, in dit geval P, waarvan de waarde telkens wanneer de lus doorlopen wordt, verandert.

Het programma dat we als voorbeeld van een DO/LOOP hebben gegeven, kan ingekort worden wanneer we een FOR/NEXT lus aanwenden:

```
1ØØ     INPUT PROMPT "Tik a.u.b. een getal in en ik
         zal er de 'tafel' voor geven. ":A
11Ø     FOR P = Ø TO 15Ø STEP A
12Ø         PRINT P,
13Ø     NEXT P
14Ø     END
```

Dit programma bewerkstelligt bijna hetzelfde als de DO/LOOP (pagina 70), maar dan wel op een heel andere manier, DO/LOOPS zijn enkel iets gemakkelijker te lezen en te begrijpen wanneer u onbekend bent met computerprogramma's.

Merk op dat de FOR/NEXT lus altijd aan het begin getest wordt; is de eindlimiet, zoals aangegeven in de FOR regel, overschreden, dan wordt de lus niet doorlopen. Probeert u in het bovenstaande "tijdtafel"-programma maar getallen in te voeren die groter zijn dan 15Ø.

Misschien wilt u wel een aantal verschillende manieren proberen om dit programma door middel van PRINT AT (pagina 37) op het scherm te zetten. U zou bij voorbeeld, door het PRINT en het GRAPHICS commando te combineren, een tabel die uit rechthoeken bestaat kunnen ontwerpen waar uw

"tijdtafel" in ondergebracht kan worden. Deze keer laten we de rest aan uzelf over! Programmeren is van nature een avontuurlijke onderneming, u moet de dingen zelf zien te achterhalen. Misschien heeft u zelfs nog betere ideeen. Als u kleine kinderen hebt of een klein broertje of zusje, is dit het soort programma waarbij u het nuttige met het aangename kunt verbinden.

GENESTE LUSSEN

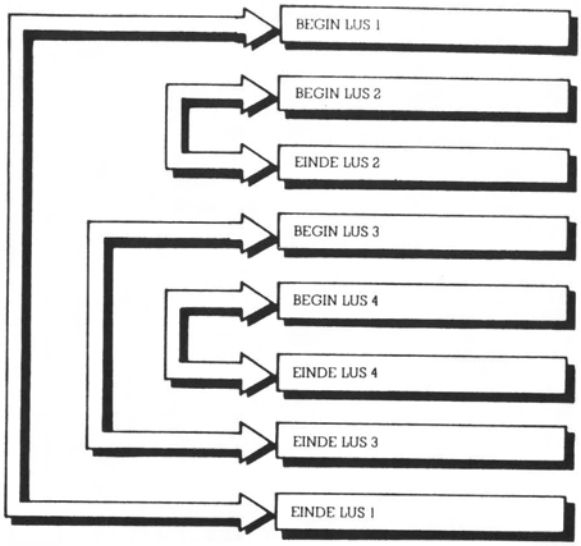
Deze woorden zouden wel eens raar kunnen klinken. Een geneste lus is echter slechts een lus die binnen een andere lus is geplaatst. Wat u moet onthouden is dit: de tweede lus mag nooit na de eerste eindigen. Hier hebben we een voorbeeld:

```
100     FOR P= 1 TO 20 !          Begin eerste lus.
110        PRINT P,
120        FOR A = 1 TO 4 !       Begin tweede lus.
130           PRINT P + 10; P - 10;
140        NEXT A !               Einde tweede lus.
145        PRINT
150     NEXT P !                  Einde eerste lus.
```

In de tweede lus kunt u weer een nieuwe insluiten, in de nieuwe lus weer eentje, enzovoort. Bij nadere beschouwing van een reeks geneste lussen eindigen deze in aflopende volgorde. Voor het begin van elke afzonderlijke lus geldt het tegenovergestelde. Zie scherm op volgende pagina.

Zoals u kunt zien kunt u lussen ook — als u echt vindt dat lussen de oplossing zijn voor al uw problemen — tussen twee beginpunten of twee eindpunten van andere lussen voegen.

Maar bij ingewikkelde patronen als hierboven is helderheid van geest vereist als u het spoor niet bijster wilt raken. De Enterprise biedt u ruggesteun door in te springen i.g.v. een lus, zoals u heeft kunnen zien in het laatste programma. Hierdoor is een en ander gemakkelijker te lezen. Regels met commentaar dragen ook bij tot een beter begrip. Doorgaans, wanneer u verschillende lagen met geneste lussen hebt, is het eenvoudiger de binnenste "lagen" als functie te definieren (zie pagina 76). Op deze manier blijft het doel en de stroom van het programma duidelijk.


BEGIN LUS 1
BEGIN LUS 2
EINDE LUS 2
BEGIN LUS 3
BEGIN LUS 4
EINDE LUS 4
EINDE LUS 3
EINDE LUS 1

In romans worden computers vaak voorgesteld alsof ze met rede begaafd zijn. Evenals de meeste mensen hebt u misschien ook wel gedacht dat dit het geval is. Nu weet u wel beter.

Computers beschikken niet over het denkvermogen van de mens, maar ze kunnen wel heel eenvoudige beslissingen nemen en vergelijkingen uitvoeren.

VERGELIJKINGEN DIE GEBRUIK MAKEN VAN IF/THEN

We zullen maar met vergelijkingen beginnen. Herinnert u zich nog de vergelijkings-bewerkingstekens die in het eerste gedeelte van deze handleiding ter sprake zijn gekomen? Deze tekens liggen ten grondslag aan de manier waarop computers vergelijkingen maken en soms beslissingen nemen, gebaseerd op die vergelijkingen.

Probeer het onderstaande programma maar. Dit programma laat u zien hoe m.b.v. vergelijkings-bewerkingstekens bepaald kan worden of twee strings verschillend zijn of niet.

```
100     LET THAT$ ="ketel"
110     INPUT PROMPT "Tik a.u.b. enkele letters
        in:":THIS$
120     !  ---------------------------------------
130     !     100 en 110 moeten bekend zijn. De beide
140     !     strings worden in 220 en 230 met elkaar
150     !     vergeleken met gebruikmaking van < >,
160     !     wat betekent "verschillend van". Zijn ze
170     !     niet verschillend, dan worden beide
180     !     afgedrukt. Verschillen ze wel, dan wordt
190     !     een keer "ketel" afgedrukt.
195     !  ---------------------------------------
200     PRINT
210     PRINT
220     IF THIS$=THAT$ THEN PRINT THIS$;" = ";
        THAT$
230     IF THIS$ < > THAT$ THEN PRINT THAT$
240     !  ---------------------------------------
250     !     220 en 230 maken gebruik van IF/THEN
260     !     om een beslissing te nemen. Als THIS$
270     !     hetzelfde is als THAT$, breng dan beide
290     !     in beeld. Verschillen ze, druk dan THAT$
292     !     af.
295     !  ---------------------------------------
300     END
```

IF/THEN is een van de twee belangrijkste manieren waarop de Enterprise een beslissing kan nemen. Het is een van de opdrachten in BASIC die helemaal Engels zijn. IF/THEN wordt in combinatie met logische bewerkingstekens toegepast om te kijken of variabelen en getallen aan bepaalde voorwaarden voldoen. IF/THEN wordt niet gebruikt om berekeningen uit te voeren met variabelen, maar wel om een programma in een andere richting te sturen indien een getal of een string tot op zekere hoogte ergens aan voldoet.

U kunt bij voorbeeld een programma voor een spel schrijven waarin elke speler slechts een bepaald aantal beurten per spelronde heeft. U neemt dus een variabele die telkens wanneer een speler zijn beurt heeft gehad met 1 verhoogd wordt.

Zodra de variabele gelijk is aan het aantal beurten dat elke speler mag hebben, kan een IF/THEN opdracht gebruikt worden om het programma te laten weten dat de variabele weer op Ø gezet moet worden voor de volgende speler, dat de eerste speler geinformeerd moet worden dat zijn beurt voorbij is, dat diens score vastgelegd moet worden, dat nu de volgende speler aan de beurt is, enz.

IF BLOKKEN

Een IF/THEN opdracht hoeft niet slechts uit een regel te bestaan U kunt ook een zogenaamd IF-blok gebruiken. Dit staat gewoon voor meerdere regels waardoor de computer meerdere vergelijkingen maken kan of beslissingen nemen. Dit houdt ook in dat u meerdere regels kunt nemen voor elke voorwaarde waardoor uw mogelijkheden voor het nemen van beslissingen in hoge mate verruimd worden.

Regel 22Ø en 23Ø van het laatste programma had u ook anders kunnen schrijven. De regels hieronder had u in plaats van de beide oorspronkelijke regels kunnen gebruiken. De uitwerking is hetzelfde.

```
22Ø     IF THIS$=THAT$ THEN
224             PRINT THIS$;"=";THAT$
228     ELSE
232             PRINT THAT$
236     END IF
```

De reden daarvoor is dat de computer slechts twee alternatieven had. De twee strings zijn ofwel hetzelfde

ofwel niet hetzelfde. Bovenstaande regels luiden dus als volgt: "Zijn THIS$ en THAT$ identiek aan elkaar, druk dan beide af. Is iets anders waar, druk dan alleen THAT$ af." Dit is een voorbeeld van een klein IF-blok — het bevat een IF opdracht, een THEN opdracht, een ELSE opdracht (het spreekt voor zich dat slechts een ELSE opdracht in een IF-blok gestopt kan worden) en de woorden END IF die de computer laten weten dat er geen vergelijkingen meer hoeven te worden gemaakt.

ELSE betekent "anders". Dit houdt in dat u IF/THEN kunt gebruiken om de computer bepaalde dingen te laten doen indien dat en dat "waar" is, en ELSE om andere mogelijkheden na te gaan. ELSE moet altijd op een aparte regel staan, nooit op dezelfde regel als een IF/THEN opdracht.

Het onderstaande voorbeeld is een IF-blok. Hier komen we ook een numerieke functie tegen die we al eerder hebben gezien (pp. 5-6).

```
100     PROGRAM "Tekenen"
110     RANDOMIZE
120     LET GETAL = RND (5)
130     !-------------------------------------------
140     !   110 en 120 vormen willekeurige getallen!
150     !   Deze getallen kunnen gebruikt worden
160     !   om patronen op het scherm te tekenen,
170     !   spelletjes als roulette te spelen en vele
180     !   andere dingen. Hier zullen we de
190     !   willekeurige getallen gebruiken om
200     !   figuren op het scherm te tekenen. De
210     !   commando's worden uitgelegd in het
220     !   hoofdstuk "Grafische gegevens-
225     !   verwerking". Het woord RND laat de
230     !   computer een willekeurig getal
235     !   produceren en RANDOMIZE zorgt ervoor
240     !   dat het getal, telkens wanneer het
245     !   programma gedraaid wordt, verschillend
255     !   is.
260     !-------------------------------------------
270     GRAPHICS
280     PLOT 600, 300;
290     OPTION ANGLE DEGREES
300     PLOT ANGLE 0;
310     IF      GETAL = 1 THEN
```

```
320         PRINT "Getal =";GETAL
330         PLOT FORWARD 100;
340         PLOT LEFT 90;
350         PLOT FORWARD 100;
360         PLOT LEFT 90;
370         PLOT FORWARD 100;
380         PLOT LEFT 90;
390         PLOT FORWARD 100
400   !     ---------------------------------------
410   !     270 to 300 bereiden de computer voor op
420   !     het tekenen van lijnen. In dit voorbeeld
430   !     zijn enkele zeer eenvoudige commando's
440   !     voor grafische gegevensverwerking
450   !     benut. LEFT geeft keerpunten aan;
465   !     FORWARD wordt gevolgd door een
470   !     aantal schermposities (gemeten volgens
480   !     de conventies voor grafische gegevens-
485   !     verwerking. De regels 330 tot 390 tekenen
488   !     een vierkant.
490   !     ---------------------------------------
500   ELSE IF GETAL = 2 THEN
510         PRINT "Getal =";GETAL
520         PLOT FORWARD 50;
530         PLOT LEFT 90;
540         PLOT FORWARD 100;
550         PLOT LEFT 90;
560         PLOT FORWARD 50;
570         PLOT LEFT 90;
580         PLOT FORWARD 100
590   !     ---------------------------------------
600   !     520 - 580 tekenen een rechthoek.
610   !     ---------------------------------------
620   ELSE IF GETAL = 3 THEN
630         PRINT "Getal =";GETAL
640         PLOT FORWARD 100;
650         PLOT LEFT 120;
660         PLOT FORWARD 100;
670         PLOT LEFT 120;
680         PLOT FORWARD 100
690   !     ---------------------------------------
700   !     640 - 680 tekenen een driehoek.
710   !     ---------------------------------------
720   ELSE
730     TEXT
```

```
74Ø       PRINT "Ik heb geen instructies voor
          getal ";GETAL
75Ø     END IF
76Ø     WAIT DELAY 3
77Ø     GOTO 1ØØ
```

Nu we hebben kunnen spelen met de grafische gegevensverwerking van de Enterprise keren we terug naar de IF/THEN opdracht. Zoals u heeft kunnen zien kunnen al deze getallen en BASIC woorden u tot steun zijn bij het tekenen van figuren — maar ook bij het maken van muziek en het voortbrengen van geluidseffecten. Het enige dat u moet doen is "gewone" BASIC commando's combineren met commando's voor grafische gegevensverwerking.

Het bovenstaande programma vormt zijn eigen willekeurige getal door gebruik te maken van een formule (u hoeft zich nooit druk te maken over de vraag hoe, tenzij u echt geinteresseerd raakt in de werking van het binnenste van de computer) waarmee getallen gecreeerd kunnen worden "uit het niets". Indien gewenst zou u incidenteel wel eens een reeks van deze willekeurige getallen kunnen afdrukken met behulp van een DO/LOOP, maar u zou dan niet in staat zijn om enig patroon in de reeks te ontdekken.

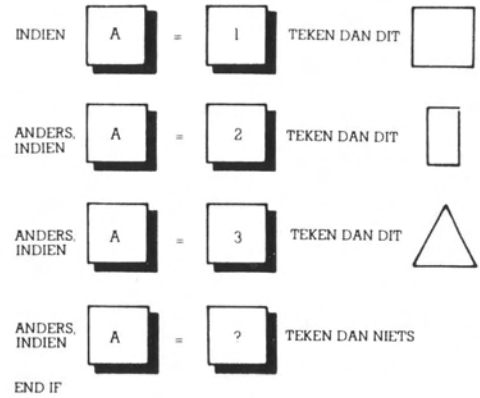
Daarna zal de computer afhankelijk van het getal dat door RND is voortgebracht aan de slag gaan en een van de drie mogelijke figuren tekenen. Als het getal niet in een van de drie IF regels genoemd wordt, zal de computer in plaats van de IF opdrachten de eenzame ELSE opdracht (regel 68Ø) uitvoeren die in vergelijking met de rest van het programma echt een beetje armzalig aandoet. Dit werd nu bedoeld met het splitsen van de voorwaarden in meerdere regels — een blok.

Het laatste programma laat precies zien hoe een IF-blok gebruikt moet worden. Elk mogelijk programma-verloop wordt gemarkeerd door IF en ELSE IF.

Een IF-blok moet altijd met END IF eindigen — anders zal de computer de rest van het programma niet uitvoeren als de laatste IF test is mislukt.

IF/THEN als een regel is om die reden geschikt voor het opsporen van uitzonderingen of voor het nemen van eenvoudige beslissingen. Als blok kan IF/THEN gebruikt worden om afhankelijk van een beslissing die

via het IF-blok aan het begin is genomen, een volledig en ingewikkeld programma uit te voeren.



SELECT CASE

Met SELECT CASE kunt u soortgelijke vergelijkingen maken en beslissingen nemen als met IF/THEN. CASE moet u interpreteren als "in geval van..." en niet zozeer als "indien... dan... of anders...".

Merk in het hierna volgende voorbeeld de afwezigheid op van de benaming van de variabele op de CASE regels. CASE 1,2,3 is goed, CASE X=1,2,3 is fout — de variabele die getest moet worden is al gegeven door de SELECT regel. Zoals u zult zien, kunt u met CASE sneller en duidelijker een keus maken uit verschillende alternatieven dan wanneer u IF/THEN toepast.

Evenals bij een IF-blok moet het eind van een SELECT-blok gemarkeerd worden — in dit geval door END SELECT in te tikken.

```
100   CLEAR SCREEN
110   !
111   !   Regel 100 wist het scherm. Daarna wordt
112   !   op het midden van het scherm een menu
113   !   afgedrukt -PRINT AT wordt gebruikt om
114   !   de letters op de juiste plaats te zetten.
115   !   Dan wordt u verzocht uw keuze in te
116   !   tikken; het getal dat u kiest wordt in
117   !   variabele A gestopt. Wanneer u een getal
118   !   intikt dat groter dan 3 of kleiner dan 1 is,
119   !   vraagt het programma u alleen maar
```

```
120  !     nogmaals een getal in te tikken. 410
121  !     draagt daar zorg voor door gebruik te
122  !     maken van een lus. Decimale getallen
123  !     worden ook geweigerd. Daar zorgt A < >
124  !     INT (A) voor.
125  !     ---------------------------------------
150  PRINT AT 9,18:"MENU"
200  PRINT AT 11,10:"1) Druk mijn naam af"
250  PRINT AT 12,10:"2) Druk jouw naam af"
300  PRINT AT 13,10:"3) Druk beide namen af"
360  DO
400       INPUT AT 16,10, PROMPT"Voer a.u.b. uw
          keuze in ":A
410  LOOP WHILE A < 1 OR A > 3 OR A < >
     INT(A)
450  CLEAR SCREEN
460  !     ---------------------------------------
465  !     Het programma vanaf regel 500 tot 1000
466  !     neemt een beslissing over wat er met uw
467  !     keus moet gebeuren en voert dat uit. De
468  !     getallen die telkens achter CASE staan
469  !     zijn de getallen waarmee A overeen kan
470  !     komen. 550 en 600 luiden als volgt: "In
472  !     geval van A=1 dient ENTERPRISE!!!" op
473  !     het midden van het scherm afgedrukt te
474  !     worden." 1000 is van essentieel belang:
475  !     deze regel laat de machine weten dat er
476  !     geen gevallen meer hoeven te worden
477  !     verwacht.
480  !     ---------------------------------------
500  SELECT CASE A
550  CASE 1
600     PRINT AT 9,18: "ENTERPRISE!!!"
650  CASE 2
700     INPUT PROMPT "Mag ik dan a.u.b. uw
        naam weten.": NAME$
750     PRINT AT 9,18:NAME$
800  CASE 3
850     INPUT PROMPT "Mag ik dan a.u.b. uw
        naam weten.":NAME$
900     PRINT AT 9,18:NAME$
950     PRINT AT 11,18: "ENTERPRISE!!!"
1000 END SELECT
1015 !     ---------------------------------------
1016 !     Regel 1200 tot 1300 lijken erg sterk op de
```

```
1020    !     check van A aan het begin van het
1022    !     programma. Ze zien erop toe dat het
1025    !     programma alleen eindigt of opnieuw
1027    !     begint als de eerste letter van A$ —
1030    !     vertaald naar een hoofdletter — ofwel "J"
1035    !     ofwel "N" is. GOTO 100 laat de computer
1037    !     weten alleen dan naar het begin van het
1040    !     programma te springen wanneer de
1041    !     eerste letter van A$ (vertaald naar een
1043    !     hoofdletter) "J" is. In het andere geval
1045    !     eindigt het programma — wat alleen kan
1046    !     gebeuren, zoals u kunt zien, als de eerste
1047    !     letter van A$ "N" is.
1048    !     ---------------------------------------
1050    PRINT
1100    PRINT
1150    PRINT
1200    DO
1250       INPUT PROMPT "Zou u dat nog eens willen
           herhalen?":A$
1300    LOOP UNTIL UCASE$(A$(1:1))="J" OR
        UCASE$(A$(1:1))="N"
1350    IF UCASE$(A$(1:1))="J" THEN GOTO 100
1400    PRINT
1450    PRINT
1500    PRINT
1550    END
```

Probeer een programma te schrijven dat een dobbelsteen simuleert. Hiervoor kunt u ofwel IF/THEN ofwel SELECT CASE gebruiken. Gebruik een willekeurig getal zoals in het programma op pagina 78. Dit getal moet natuurlijk tussen 1 en 6 komen te liggen.

```
RND (6)+ 1
```

zou daar dus zorg voor dragen. In de wetenschap dat u zes mogelijke uitkomsten heeft zou het u niet al te zwaar mogen vallen. Misschien kunt u ook nog gebruik maken van grafische gegevensverwerking om de dobbelsteen op het scherm af te beelden.

Onthoud dat u niet per se ELSE hoeft te gebruiken in geval van IF/THEN als een IF-blok. Evenals het aantal mogelijkheden binnen een blok of de selectie van CASEs zijn ELSE regels facultatief.

Case kan eveneens een "ELSE regel" bevatten, te weten CASE ELSE — net zoals CASE 1 of CASE "HALLO". Merk op dat CASE ook in combinatie met strings toegepast kan worden; bovendien heeft CASE als voordeel tegenover IF/THEN dat meerdere "waarheden" onder een noemer gebracht kunnen worden — voorbeeld:

```
CASE 1,3,5
PRINT "Dit zijn oneven getallen."
```

Experimenteer tenslotte een beetje met de ASCII-code (zie hiervoor "De tekenset" op pagina 121). Hiermee kunt u de computer strings in alfabetische volgorde laten rangschikken. De ASCII-code bestaat uit getallen die voor tekens staan. De toepassing van reeksen (die hierna aan de orde komen) is eveneens een steun bij dat soort programmeerwerk.

U zult wel ontdekt hebben dat programmeren hoofdzakelijk een manier is om gegevens te verwerken. Tot dusverre heeft u slechts kleine hoeveelheden gegevens in programma's gebruikt, hetzij regels en getallen die u als antwoord op INPUT opdrachten hebt ingetikt hetzij regels die de computer voor u in beeld heeft gebracht om te lezen.

Waarschijnlijk hebt u zich al afgevraagd hoe het zit met grotere hoeveelheden gegevens — een uitgebreide lijst met woorden of getallen, of verschillende alinea's met instructies voor een spel of een programma dat u van steun is bij het beheer van uw financien. De Enterprise biedt enkele uiterst efficiente manieren om lijsten met getallen of grote aantallen woorden te verwerken.

Gebruikmaking van een reeks is de manier waarop u bij voorbeeld een lijst met namen bijhoudt in een programma. Een reeks is te vergelijken met een grote variabele die meerdere kleine variabelen kan bevatten. We zijn begonnen met het vergelijken van variabelen met doosjes waarvan de inhoud kan veranderen. Een reeks kan beschouwd worden als een grote doos waarin een bepaald aantal kleine doosjes opgeslagen wordt. Maar u kunt een reeks ook zien als een bladzijde van een notitieblok waarop een lijstje gemaakt kan worden.

NUMERIEKE REEKSEN

Ook bij reeksen, evenals bij variabelen, maken we onderscheid tussen tekstuele en numerieke reeksen. Een numerieke reeks zou u bij voorbeeld als volgt kunnen omschrijven:

```
NUMERIC STORE (1 TO 1Ø).
```

Voer dit in met regelnummer 1ØØ. Zo wordt de computer geinformeerd dat een "houder" met de naam STORE apart gehouden moet worden met plaats voor 1Ø kleinere variabelen. Deze variabelen kennen we als de elementen van de reeks. In deze kleine ruimtes kunt u van alles en nog wat opslaan. De zevende variabele (of het zevende element) in STORE zou STORE(7) genoemd worden.

Tik nu de volgende regels in:

```
11Ø     FOR S = 1 TO 1Ø
```

```
120        INPUT PROMPT "Voer een getal in":
           STORE(S)
130     NEXT S
```

Hierdoor wordt het mogelijk getallen in uw reeks onder te brengen. Draai bovenstaande regels en tik na afloop PRINT STORE(9) of (8) of welk ander getal van 1 tot 10 ook. Zoals u ziet, kunt u getallen in de computer invoeren, maar u kunt ook de temperatuur van elke dag in de maand december vastleggen. Hiervoor zou u een reeks met de naam DECEMBER kunnen gebruiken waarvan de elementen van 1 tot 31 zijn genummerd. De temperatuur op een dag zou de inhoud vormen van een element en de nummers van de elementen zouden de datum aangeven.

Een reeks zou ook uit elementen kunnen bestaan die genummerd zijn van 56 tot 76, of van 123 tot 171, of zelfs van 12345 tot 12445. Indien gewenst kan een negatief getal gebruikt worden voor het hoogste of het laagste element — bij voorbeeld NUMERIC TABLE (-10 TO 10) of NUMERIC TABLE (-20 TO -10). De elementnummers zijn er slechts voor verwijzing.

TWEEDIMENSIONALE REEKSEN

Een reeks kan ook meer omvatten dan slechts een lijst. De beide hierna volgende figuren laten het verschil zien tussen een een- en een tweedimensionale reeks.

1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18

Een tweedimensionale reeks kan voorgesteld worden als een raster. U kunt het ook zo zien dat u deze keer uw doosjes in een kast stopt waarvan elke plank een nummer heeft. Op elke plank kunnen evenveel doosjes opgeborgen worden. Middels A (1 TO 4, 1 TO 4) krijgt u een reeks die als volgt gevisualiseerd kan worden:

1.1	1.2	1.3	1.4
2.1	2.2	2.3	2.4
3.1	3.2	3.3	3.4
4.1	4.2	4.3	4.4

Dit type reeks kan bijvoorbeeld gebruikt worden als dambord. Er kan dan informatie opgeslagen worden over de plaats van elke afzonderlijke schijf op het bord. U zou een reeks als deze ook kunnen gebruiken als een tabel met getallen of woorden — een veelzijdiger soort lijst.

TEKSTUELE REEKSEN

Tekstuele reeksen worden op dezelfde manier omschreven als numerieke reeksen, alleen wordt nu het woord STRING gebruikt. Bovendien is het met een STRING reeks mogelijk de lengte van elk element te veranderen — dit is niet mogelijk met een numerieke reeks.

De lengte van de elementen binnen een reeks veranderen betekent niets anders dan de inhoud van die elementen wijzigen. Evenmin als u een emmer telkens tot de rand hoeft te vullen als u er water in doet, net zomin hoeft u de elementen in een reeks volledig te benutten.

Het is u al bekend dat de Enterprise over een bepaalde hoeveelheid ruimte beschikt waarvan programma's weer gebruik kunnen maken.

Wanneer u een tekstuele reeks benoemt, zal de computer voor elk element een bepaalde hoeveelheid ruimte reserveren. Deze ruimte wordt de MAXLEN (maximale lengte) genoemd. De MAXLEN is 132 tekens, tenzij anders aangegeven. Dit is niet van toepassing op getallen; het geldt alleen voor tekstuele reeksen.

Als u in het geheugen van uw computer ruimte wilt reserveren — of elementen wilt hebben die langer zijn dan 132 tekens — moet u de computer als volgt daarvan op de hoogte stellen.

```
STRING ARRAY$(30 TO 50)*10
```

Door deze BASIC opdracht wordt een reeks gereserveerd met 20 elementen (genummerd van 30 tot 50); elk element kan 10 tekens bevatten. De MAXLEN (ARRAY$(45)) bedraagt dus 10. Hoe langer uw elementen zijn des te meer kunnen ze bevatten, maar daardoor wordt wel meer geheugen in beslag genomen en minder ruimte overgelaten voor andere dingen. U moet de inhoud van de elementen dus alleen dan vergroten, wanneer dat echt nodig is of wanneer u zeker weet dat er genoeg ruimte voor is. Als u het geheugen van de Enterprise uitbreidt, kunt u natuurlijk veel omvangrijkere programma's opslaan en veel meer kwijt in uw reeksen.

VARIABELEN BENOEMEN

Eenvoudige variabelen kunnen ook benoemd worden door gebruik te maken van STRING of van NUMERIC. Met NUMERIC A wordt een numerieke variabele benoemd. NUMERIC A,M,N,H,D, of iets dergelijks zou gebruikt kunnen worden om al uw variabelen in een keer te benoemen — aan het begin van een programma. STRING A$,B$,HALLO$*8... bewerkstelligt hetzelfde maar is nu van toepassing op string-

variabelen. Het enige verschil tussen een variabele benoemen en een reeks benoemen is, dat in geval van een reeks de elementen gespecificeerd worden.

READ/DATA

Veel BASIC woorden gaan in hand met andere BASIC woorden. READ en DATA (evenals het woord RESTORE — opnieuw instellen) zijn zulke woorden.

Ze komen in dit gedeelte van de handleiding aan de orde, omdat ze de aanwezigheid van grote hoeveelheden gegevens in een programma mogelijk maken.

Bovendien bieden ze nog een andere manier om getallen of strings in een reeks te stoppen. READ is het woord dat het zware werk verricht. Hier volgt een kort voorbeeld:

```
 80    LET P = 0
 90    DO UNTIL P = 0
100        READ P
110        PRINT P,
120    LOOP
130    DATA 1,2,3,4,5,6,7,8,9,10
150    DATA 11,12,13,14,15,16,17,18
160    DATA 19,20,21,22,23,24,25,0
170    END
```

De DATA regels bevatten de gegevens. Elk stukje informatie moet van het volgende stukje gescheiden worden door een komma — zo weet de computer dat hij het ene gegevensbestanddeel heeft gehad en dat nu het volgende aan de beurt is. READ laat de computer weten een gegevensbestanddeel nader te bekijken en er iets mee te doen — wat dan ook. In dit geval wordt het bestanddeel in de variabele P gestopt en daarna de inhoud van P op het scherm gezet.

Merk op dat dezelfde variabele gebruikt wordt voor alle gegevensbestanddelen. Dit maakt niets uit — de variabele wordt steeds opnieuw gebruikt. Met andere woorden: het programma leest (READ) een gegevensbestanddeel (DATA item), plaatst dit in de variabele die we P hebben genoemd, zet het op het scherm, leest het volgende gegevensbestanddeel, plaatst dit in P in plaats van het laatste gegevensbestanddeel, enzovoort. De computer zal elk gegevensbestanddeel slechts een keer lezen. Na lezing

van een bestanddeel weet de machine zich de positie ervan te "herinneren" en gaat verder met het volgende bestanddeel (van links naar rechts en van boven naar beneden lezend, zoals een boek gelezen wordt). Als alle gegevens gelezen zijn gaat de machine ervan uit dat er geen meer zijn — wordt de computer opgedragen verder te zoeken dan raakt hij in verwarring!

Het onderstaande programma demonstreert de toepassing van READ/DATA met strings.

```
150   CLEAR SCREEN
200   PRINT "Ik ga u een verhaal vertellen."
250   PRINT
300   PRINT "Daar gaan we dan!"
350   PRINT
400   PRINT
410   FOR X = 1 TO 5000
415   NEXT X
420   !  ---------------------------------------
430   !  Regel 410 en 415 specificeren slechts een
440   !  pauze. De DO/LOOP (550-810) is het
450   !  belangrijkste gedeelte. Laten we eens
460   !  kijken wat er gebeurt. Eerst wordt A$
470   !  gelezen — d.w.z. er wordt een DATA
480   !  opdracht gezocht (die zich overal in het
490   !  programma kan bevinden), gelezen en
500   !  gecontroleerd of het om "END" gaat. Zo
515   !  ja, dan stopt de computer de lus en gaat
520   !  verder met de rest van het programma.
525   !  Zo niet, dan draagt 700 de computer op
530   !  de string van de DATA regel op het
531   !  scherm te zetten en er een spatie achter
532   !  te zetten.
535   !  ---------------------------------------
550   DO
600         READ A$
650         IF A$= "END" THEN EXIT DO
700         PRINT A$;" ";
750         FOR Y = 1 TO 100
760         NEXT Y
810   LOOP
820   !  ---------------------------------------
830   !  De rest van dit programma ziet er
840   !  misschien een beetje verwarrend uit.
```

```
850  !     Maar als u bedenkt dat de DATA
855  !     opdrachten door een eerder gedeelte
870  !     van het programma gelezen worden en
873  !     dat 900-1100 eigenlijk pas uitgevoerd
875  !     worden nadat de gegevens gelezen zijn,
880  !     wordt de logica weer duidelijk.
890  !
900  PRINT
950  PRINT
1000 PRINT "                                EINDE"
1050 PRINT
1100 PRINT
1125 END
1150 DATA Er,was,eens,een,kleine,computer,
1200 DATA die,Enterprise,heette.,Het,was,een,erg,
1250 DATA gelukkige,computer.,De,beste,
     programmeurs,van,
1300 DATA het,land,Engeland,hadden,maanden,lang.
1350 DATA dag,in,dag,uit,gewerkt,om,van,de,
     Enterprise,
1400 DATA de,beste,computer,te,maken,ooit,
     vervaardigd.,
1450 DATA Thans,leert,u,BASIC,te,schrijven,op,de,
     Enterprise.,
1500 DATA Heeft,u,even,geluk!
1550 DATA END
```

Eventueel kunt u een modificatie in dit programma aanbrengen. Voeg nu 100 DO en 1120 LOOP aan dit programma toe. RUN het vervolgens. Zoals u kunt zien, gaat het een keer goed en verschijnt daarna de boodschap dat er geen gegevens meer zijn. Voeg dus regel 1110 RESTORE toe. Dit laat de computer weten terug te gaan en de gegevens opnieuw te gebruiken. Indien gewenst kunt u RESTORE 1400 invoegen waardoor de computer de gegevens in 1400 en verder gebruikt. Hierdoor wordt het mogelijk een gedeelte van de gegevens te selecteren en dit zo nodig meerdere malen te gebruiken.

Raak niet verward door het feit dat tekstuele gegevens niet tussen aanhalingstekens hoeven. Doordat u de computer laat weten READ A$ of X$ gaat deze op zoek naar strings — het dollarteken is hiervoor verantwoordelijk. Het zou wel erg vervelend zijn als u elk woord van een lange lijst met woorden die als

gegevens in een programma opgenomen moeten worden, tussen aanhalingstekens moest plaatsen — zie het maar als een mazzeltje.

Hetzelfde geldt voor INPUT opdrachten: u kunt "ja" of "nee" intikken als antwoord op een vraag gesteld door een INPUT PROMPT maar de aanhalingstekens kunt u weglaten. Via het dollarteken achter de benaming van een variabele komt de computer te weten dat hij een string ontvangt.

Maar u moet een DATA of een INPUT bestanddeel wel tussen aanhalingstekens plaatsen, als u een komma in de string wilt opnemen — anders verkeert de computer in de veronderstelling dat de komma het einde van de string markeert.

FUNCTIES DEFINIEREN

Een functie is een soort "programma binnen een programma" dat tot doel heeft een speciale taak uit te voeren — een reeks instructies die gereserveerd worden voor wanneer u ze nodig heeft en die telkens weer gebruikt kunnen worden.

Hier volgt een eenvoudig voorbeeld: stel, dat u in verschillende stadia van een programma een speciale boodschap op het scherm wilt brengen. Deze actie zou u als een functie kunnen definieren door iets dergelijks als hieronder in te tikken:

```
100     DEF WARNING
110         CLEAR SCREEN
120         PRINT AT 10,7: "EN NU OPGELET..."
130         !
140         SOUND !
150         !
155         !
160         !
170     END DEF
```

Regel 140 voegt een geluidssignaal eraan toe.

U moet de functie altijd een naam geven (in dit geval WARNING — waarschuwing) en met het sleutelwoord DEF (initie) introduceren. De definitie die dan volgt kan 1 regel of 1000 regels lang zijn, maar ongeacht de lengte moeten de woorden END DEF (einde definitie) de laatste regel vormen.

FUNCTIES OPROEPEN

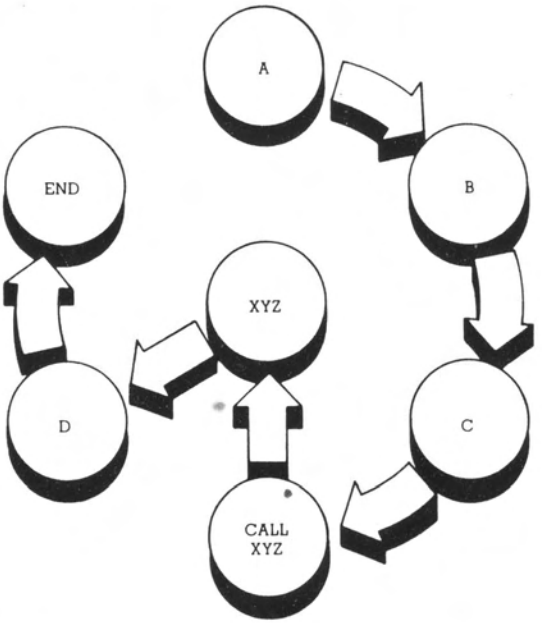
Een functie op zich kan niet functioneren. Tik bovenstaande regels in en probeer ze uit te voeren; vooralsnog gebeurt er niets. De functie moet geactiveerd worden door de opdracht CALL WARNING (roep waarschuwing op). Voer dit in als regel 180 en draai het programma; voer dezelfde instructie vervolgens in de rechtstreekse modus in. Onthoud dat de definitie van een functie ook na de CALL opdracht kan volgen — of zelfs na de END regel van het programma. Als u in plaats van regel 180

```
80      CALL WARNING
90      END
```

had ingetiket zou de functie ook nog haar werking hebben.

Telkens wanneer de computer een CALL opdracht

tegenkomt, stopt hij met datgene waarmee ie bezig is, gaat op zoek naar de functie die opgeroepen wordt en voert deze uit. Daarna keert de machine terug naar de plaats in het programma die direct na de CAll volgt. De tekening zal een en ander verduidelijken.



Onthoud dat een functie inactief is, zolang u de computer niet laat weten dat ze uitgevoerd moet worden. Als de computer alleen maar de opeenvolgende regelnummers afwerkt en het gedeelte van het programma bereikt waar zich een functie bevindt, slaat hij deze over en gaat verder met dat wat daarna komt. U heeft niets aan een functie als de naam ervan niet al ergens in het programma voorkomt.

BIJZONDERE EN ALGEMENE VARIABELEN

Doorgaans hebben functies betrekking op variabelen. Wil dit geen problemen opleveren, dan moeten enkele belangrijke regels in acht genomen worden.

Tik het volgende in:

```
100     DEF CUBE
110           INPUT PROMPT "Getal dat tot de 3-de
              macht verheven moet worden: ":Z
```

```
12Ø          PRINT Z; " tot de 3-de macht verheven
             is";Z*Z*Z
13Ø      END DEF
14Ø      CALL CUBE
```

— en voeg eraan toe, nadat u dit heeft gedraaid:

```
15Ø      PRINT Z
```

en draai het opnieuw. Nu zult u zien dat de computer, ofschoon de CUBE functie nog steeds functioneert, een foutboodschap geeft zodra hij bij regel 15Ø aankomt.

Hoe kom dat?

Het antwoord is, dat Z in dit geval zogenaamde bijzondere variabele is. Z maakt uitsluitend deel uit van de CUBE functie, de rest van het programma is er niet mee bekend. Omdat een functie door de computer als een apart klein programma behandeld wordt, kan dit programma zijn eigen prive variabelen gebruiken als ondersteuning bij de uitvoering van zijn taak.

Maar de rest van het programma kent de betekenis van deze prive variabelen niet; is de taak volbracht, dan zijn ze van geen enkele waarde meer. De computer weet dus niet wat afgedrukt moet worden, als hij bij regel 15Ø hierboven is aanbeland.

Laten we nu regel 11Ø opnieuw nummeren en er regel 9Ø van maken — zodat deze regel buitenn de functie gestiueerd wordt. Nu blijkt dat u wel een getal voor Z kunt intikken; het programma geeft u de derde macht van dit getal en drukt het zelf ook nog eens af. Anders gezegd: regel 15Ø is niet langer een bron van verwarring.

Door Z te introduceren voordat de functie opgeroepen wordt, heeft u Z veranderd in een algemene variabele.

Een algemene variabele is een variabele die voor het hele programma beschikbaar is.

Voeg er nu aan toe:

```
125      LET Z = 2Ø
```

en draai het programma opnieuw. Nu gebeurt het volgende: de functie haalt het getal uit het "doosje: (Z)" in het hoofdprogramma, voert er een berekening mee uit en drukt het resultaat af. Vervolgens wordt het getal

zelf veranderd en in het doosje teruggestopt. Daarna wordt dit nieuwe getal door het "hoofdprogramma" afgedrukt.

Onthoud dit: als een functie een regel bevat waarin een variabele ter sprake komt, en als deze variabele nog niet is geintroduceerd wanneer de functie opgeroepen wordt, beschouwt de functie de variabele als een bijzondere of prive variabele. Maar is de variabele reeds benoemd, dan zal de functie deze als een algemene variabele zien; elke nieuwe waarde die de variabele krijgt zal aan de rest van het programma doorgegeven worden.

Het benoemen van variabelen bent u al eerder in deze handleiding tegengekomen. (Moet u uw geheugen opfrissen, zie dan pagina 29 en 86). We hebben gezien dat u er het verstandigst aan doet elke variabele die u gebruikt te benoemen, hoewel dat niet strikt noodzakelijk is. Het benoemen van variabelen is met name dan belangrijk wanneer u veel gebruik maakt van functies.

Zoals u weet, kan een variabele benoemd worden door een NUMERIC of een STRING opdracht, dan wel door het woordje LET (bij voorbeeld LET A=Ø). In de voorgaande voorbeelden maakte het niet veel uit op welke manier er benoemd werd; maar als u werkt met functies moet u precies weten wat de verschillende manieren van benoemen bewerkstelligen. Binnen een functie wordt met een NUMERIC (of een STRING) opdracht altijd een bijzondere variabele gecreeerd. Voeg aan het bovenstaande programma (nadat u de oorspronkelijke regel 11Ø opnieuw heeft genummerd) dit toe:

```
11Ø     NUMERIC Z
115     LET Z = 3
```

Nu zult u zien dat het programma met twee volstrekt afzonderlijke Zetten werkt: de een binnen de functie (een "bijzondere" Z) en de ander daarbuiten (een "algemene" Z). Maar als u regel 11Ø schrapt, blijft er slechts een Z over. De LET opdracht in regel 115 creeert niet een nieuwe (bijzondere) variabele, maar verandert de algemene variabele die door regel 9Ø was geintroduceerd.

Het programma hieronder bevat enkele nogal

ingewikkelde voorbeelden van functies. Het is een opnieuw gestructureerde versie van een programma dat we in het hoofdstuk over beslissingen zijn tegengekomen. Dit programma zal niet alleen demonstreren hoe functies eruit zien binnen een programma, maar ook dat er altijd verschillende manieren zijn om een programma samen te stellen. Sommige hiervan ogen prima andere afschuwelijk, sommige zijn onbegrijpelijk andere uiterst efficient en weer andere zijn helemaal het tegendeel. Als u nu het programma in zijn totaliteit bekijkt, zult u waarschijnlijk ermee instemmen dat deze versie een aanmerkelijke verbetering is.

Zolang u wilt dat de computer meer namen afdrukt, zal het programma niet stoppen. Wilt u het stopzetten, dan moet u "N" (No = nee) intikken als u daarom gevraagd wordt.

```
100     DO
110         LET A = 0
120         LET A$ = ""
140         !-------------------------------
150         !   Regel 110 en 120 benoemen twee
160         !   algemene variabelen die gebruikt
170         !   (en gewijzigd) zullen worden door
175         !   de functies en daarna teruggegeven
180         !   aan het hoofdprogramma.
185         !-------------------------------
190         CALL MENU
200         FOR X = 1 TO 2000
210         NEXT X
220         !-------------------------------
230         !   Dan komt het hoofdprogramma dat
240         !   eerst het menu oproept en na
250         !   afwerking daarvan voor een
260         !   vertraging van 3 seconden zorgt,
270         !   zodat u kunt lezen wat er op het
275         !   scherm staat. Nadat u uw keus heeft
280         !   gemaakt uit het menu, werkt het
285         !   hoofdprogramma het CASE blok
290         !   door, voert uit wat u hebt gekozen
295         !   en roept vervolgens de ANSWER
300         !   functie op waardoor bepaald
305         !   wordt of het programma al dan niet
310         !   nogmaals uitgevoerd wordt.
```

```
330         !  ---------------------------------------
340         CLEAR SCREEN
350         SELECT CASE A
360         CASE 1
370             PRINT AT 9,18:"ENTERPRISE!"
380         CASE 2
390             INPUT PROMPT "Mag ik dan a.u.b.
                uw naam weten. ":NAME$
400             PRINT AT 9,18:NAME$
410         CASE 3
420             INPUT PROMPT "Mag ik dan a.u.b.
                uw naam weten.":NAME$
430             PRINT AT 9,18:NAME$
440             PRINT AT 11,18: "ENTERPRISE!"
450         END SELECT
460         FOR X = 1 TO 3000
470         NEXT X
480         CLEAR SCREEN
490         CALL ANSWER
500     LOOP WHILE A$ = "J"
510     END
520         !  ---------------------------------------
530         !  500 sluit de hoofdlus. Als A$"J" is
540         !  begint het programma weer van
550         !  voren af aan. Dit komt hierop neer
560         !  dat het programma pas eindigt
570         !  wanneer u "N" (of nee, of neen, etc.)
575         !  antwoordt in de ANSWER functie.
580         !  ---------------------------------------
590     DEF MENU
600         CLEAR SCREEN
610         PRINT AT 9,18:"MENU"
620         PRINT AT 11,9:"1) Druk mijn naam af."
630         PRINT AT 12,9:"2) Druk uw naam af."
640         PRINT AT 13,9:"3) Druk beide namen "
            af."
650         PRINT AT 16,1:"Wat is het getal van uw
            keuze? Voer dat a.u.b. in."
660         DO
670             INPUT A
680         LOOP WHILE A <1 OR A >3 OR
            A < > INT(A)
690     END DEF
700     DEF ANSWER
```

```
710         PRINT
720         PRINT
730         DO
740             INPUT PROMPT "Wilt u het nog
                eens over doen? ":A$
750             LET A$ = UCASE$(A$(1:1))
760         LOOP UNTIL A$ = "J" OR A$ = "N"
770     END DEF
```

Als u regel 110 en 120 verwijdert kan het programma niet uitgevoerd worden — want de twee variabelen die u dan niet hebt benoemd gaan specifiek behoren tot de functies waarvan ze deel uitmaken.

Tot dusverre hebben we funties toegepast die door een CALL opdracht geactiveerd worden en die een keur van effecten teweeg kunnen brengen, zoals bij voorbeeld ons nodigen tot het intikken van meer gegevens of boodschappen op het scherm brengen. Nu zullen we onze aandacht richten op een heel ander soort functie; een functie die slechts ten doel heeft een enkel getal terug te geven aan het hoofdgedeelte van het programma.

Verschillende van deze functies stelt de computer "gebruiksklaar" ter beschikking. Neem bijvoorbeeld maar SQR. Een programmaregel kan de volgende opdracht bevatten.
PRINT SQR(121), of PRINT SQR(P)of LET M = 2*SQR(N) + 1. De functie SQR berekent de vierkantswortel van het getal of de variabele tussen haakjes en maakt het voor u mogelijk deze vierkantswortel in een "uitdrukking" te gebruiken of er iets anders mee te doen. U bent ook al bekend met de functie INT. Zulke BASIC woorden verschaffen u de middelen om snel en gemakkelijk berekeningen uit te voeren die wel eens vaker nodig zouden kunnen zijn.

Stel, dat u een programma aan het schrijven bent waarin meerdere malen de "faculteit" van getallen voorkomt ("faculteit" betekent het produkt van een reeks factoren die telkens met een zelfde bedrag toenemen; vier faculteit is 4*3*2*1, zes faculteit is 6*5*4*3*2*1, etc.). Er bestaat geen "gebruiksklare" functie voor het berekenen van faculteiten. Maar, indien gewenst, zou u een functie kunnen bedenken met behulp van de methoden die u nu al onder de knie hebt. Dit zou u kunnen tikken:

```
100     DEF FACT
110     !
120     !               Deze functie neemt een
130     !               algemene variabele, F, van het
140     !               hoofdprogramma, wijzigt deze
145     !               en overhandigt de variabele
150     !               weer aan het
155     !               hoofdprogramma.
160     !
170             FOR Y = F - 1 TO 1 STEP - 1
180                LET F = F*Y
190             NEXT Y
200     END DEF
```

Teneinde (bij voorbeeld) dertien faculteit af te drukken of zeven faculteit in een "uitdrukking" te gebruiken zou u daarna het volgende kunnen toevoegen:

```
150     LET F = 13
160     CALL FACT
170     PRINT F
180     LET F = 7
190     CALL FACT
200     LET NUMMER = F*1.5 + 3
210     PRINT NUMMER
```

Maar zoals u kunt zien is dit aanmerkelijk omslachtiger dan wanneer een BASIC woord als SQR gebruikt wordt, want telkens wanneer de functie FACT opgeroepen wordt moet het getal waarop de functie van toepassing zal zijn eerst naar de variabele F geschreven worden. Gelukkig biedt de computer mogelijkheden deze beperking te omzeilen. Schrap al het bovenstaande en tik in plaats daarvan het volgende:

```
100     DEF FACT(X)
110             FOR Y = X - 1 TO 1 STEP - 1
120                LET X = X*Y
130             NEXT Y
140             LET FACT = X
150     END DEF
160     PRINT FACT(13)
170     LET NUMMER = FACT(7)*1.5 + 3
180     PRINT NUMMER
```

Door het op deze manier te doen maakt u van FACT min of meer een nieuw BASIC woord van uzelf dat op dezelfde manier (en net zo gemakkelijk) gebruikt kan worden als INT of SQR.

Regel 1Ø0 en 14Ø van het bovenstaande programma zijn nieuw voor u. De X tussen haakjes in de DEF regel laat de functie weten op zoek te gaan naar een getal tussen haakjes dat op het woord FACT volgt in een regel van het hoofdprogramma, en dit getal automatisch naar de bijzondere variabele X te schrijven. Regel 14Ø bewerkstelligt dat er een waarde toegekend wordt (zie regel 16Ø en 17Ø). Op deze manier kunt u de CALL opdracht achterwege laten.

PSEUDO VARIABELEN

De X tussen haakjes in regel 1Ø0 wordt in technische termen een pseudo variabele genoemd. X laat de functie weten dat er een getal gaat komen dat verwerkt moet worden. (In het laatste voorbeeld van dit hoofdstuk zien we een functie met twee pseudo variabelen waardoor de functie weet dat er twee getallen verwacht mogen worden.) Maar dat getal wordt misschien verstrekt door een algemene variabele in het hoofdprogramma. Schrap regel 16Ø-18Ø en vervang deze door:

```
16Ø     LET A = 11
17Ø     PRINT FACT(A)
```

De functie neemt nu een kijkje in doosje A en neemt notitie (kopieert) van het getal dat daar aangetroffen wordt. Vervolgens plaatst de functie een identiek getal in X dat voor de berekeningen gebruikt wordt — tijdens dit rekenproces verandert het getal in X, maar het getal in A blijft ongewijzigd.

Als we regel 16Ø en 17Ø veranderen kunnen we zien op welke specifieke manier een pseudo variabele functioneert.

```
16Ø     LET X = 11
17Ø     PRINT FACT(X)
```

U hebt nu een algemene variabele gedefinieerd met dezelfde naam als de pseudo variabele van de functie. Toch zult u zien dat het programma beide "doosjes" apart zal bekijken, ook al is er aan het begin van de

functie een getal van het ene doosje naar het andere "geschreven" (gekopieerd). U zou een extra regel aan de functie toe kunnen voegen:

```
145     LET X = 1ØØ
```

... en een extra regel aan het hoofdprogramma:

```
18Ø     PRINT X
```

... en toch zult u zien dat regel 18Ø "11" afdrukt, en niet 1ØØ. Regel 145 wijzigt alleen maar de "bijzondere" X.

PARAMETER REFERENCING

Zoëven heeft u een functie die gebruik maakte van een pseudo variabele, een berekening met een getal zien uitvoeren dat een "kopie" was van een algemene variabele. Maar hoewel de functie een getal aan het hoofdprogramma teruggaf, bleef de variabele die gekopieerd werd ongewijzigd.

Het is heel wat anders wanneer u REF ("reference": verwijzing naar) aan de psuedo variabele vooraf laat gaan — zoals uit het onderstaande eenvoudige voorbeeld zal blijken.

Bij dit programma kunt u twee getallen intikken — voor A en voor B — die vervolgens door het programma veranderd worden door A tot de macht B te verheffen en B tot de macht A:

```
1ØØ     INPUT PROMPT "FIRST NUMMER":A
11Ø     INPUT PROMPT "SECOND NUMMER":B
12Ø     CALL POWERS (A,B)
13Ø     PRINT A,B
14Ø     END
15Ø     DEF POWERS (REF X, REF Y)
16Ø          LET Z = X
17Ø          LET X = X^Y
18Ø          LET Y = Y^Z
19Ø     END DEF
```

Regel 15Ø introduceert twee pseudo variabelen. Wanneer de functie wordt opgeroepen, wordt de waarde van de eerste variabele tussen haakjes in regel 12Ø overgedragen naar de variabele X en de waarde van de tweede variabele naar Y. Dit zijn we al eerder tegengekomen, alleen was er in andere voorbeelden

slechts sprake van een pseudo variabele; omdat deze functie twee getallen (niet slechts een) aan het hoofdprogramma zal teruggeven, moet ook hier een CALL opdracht eraan te pas komen om de functie te activeren.

Het verschil dat u krijgt door REF in regel 15Ø te introduceren is eenvoudigweg dit: wanneer de functie de berekeningen heeft uitgevoerd wordt de nieuwe waarde van X weer verplaatst naar de algemene variabele A (en Y weer naar B).

Het heen en weer geschuif van waarden tussen functies en andere gedeeltes van een programma wordt "parameter passing" (het doorgeven van parameters) genoemd. Als de functie bewerkstelligt dat de algemene variabelen (of reeksen, enz.) die de getallen voor verwerking hebben geleverd een wijziging ondergaan, is er sprake van "parameter referencing". Zie hiervoor A en B in het laatste voorbeeld.

Door middel van de krachtige grafische gegevensverwerking van de Enterprise kunt u enkele indrukwekkende figuren verkrijgen die visuele effecten teweegbrengen. Hier bent u misschien al achter gekomen toen u de demo-cassette gebruikte; ook enkele programma's die we al eerder in deze handleiding zijn tegengekomen hebben een fractie van de mogelijkheden laten zien van grafische gegevensverwerking met een hoog scheidend vermogen.

In het eerste gedeelte van deze handleiding hebben we u PRINT AT uitgelegd. Dit commando maakt gebruik van een systeem waardoor het scherm in een aantal "posities" verdeeld wordt — imaginaire vierkantjes — zodat u nader aan kunt geven waar u iets afgedrukt wilt hebben. PRINT AT 1,1 bewerkstelligt dat een string (of een getal) linksboven in het beeld geplaatst wordt.

De commando's voor grafische gegevensverwerking benutten een soortgelijk systeem om lijnen en puntjes op het scherm te zetten en bieden u de mogelijkheid diagrammen en figuren te tekenen. Maar nu zijn de "schermposities" veel kleiner. Hier volgt een kort programma dat een lijn tekent:

```
100     GRAPHICS
110     PLOT 640, 360; 1000, 700
120     END
```

De eerste opdracht zult u wel al eerder zijn tegengekomen. Het woord GRAPHICS is een snelle en gemakkelijke manier om een blanco "pagina" de selecteren waarop u figuren kunt tekenen.

Het commando PLOT wordt gebruikt om puntjes te verkrijgen of lijnen te tekenen. De vier getallen die in regel 110 op PLOT volgen corresponderen met twee posities op de grafische pagina; dit programma trekt een lijn van het midden van het scherm (640, 360) naar een positie (1000, 700) ergens rechtsboven in het beeld.

Merk op dat deze "coordinaat" — getallen veel groter zijn dan de met PRINT AT gebruikte getallen die de plaats van strings en kolommen aangeven. Dit betekent niet dat er sprake is van een groter vlak; de clou is dat de "grafische pagina" verdeeld is in een aanmerkelijk groter aantal uiterst kleine posities,

waardoor u veel preciezer te werk kunt gaan dan op een "tekst" pagina. Anders gezegd: wanneer u met grafische symbolen werkt is het scheidend vermogen veel groter.

Een ander verschil tussen "grafische gegevensverwerking" en "gewone gegevensverwerking" is, dat op de grafische pagina de beginpositie (Ø,Ø) zich linksbeneden op het scherm bevindt en dat het eerste getal van beide coordinaten de horizontale positie is. Dit stemt overeen met de (x,y) conventies die doorgaans van toepassing zijn op grafisch tekenen.

Het zal u wel zijn opgevallen, toen u het programma draaide, dat er onder in het beeld nog vier regels over waren van de gewone tekstpagina. Hierdoor kunt u een en ander blijven invoeren, terwijl de tekening in beeld blijft.

Deze scheiding tussen de grafische standaardpagina en vier regels van de pagina voor tekst is gemaakt om de bewerkingen gladjes te laten verlopen; telkens wanneer u het eenvoudige commando GRAPHICS geeft wordt dat resultaat verkregen. Zo krijgt u een schermoppervlak voor het uitzetten van grafieken dat 128Ø posities horizontaal bij 72Ø posities verticaal meet; de coordinaten van de positie rechtsboven op het scherm zijn dus (1279,719).

Beide gedeeltes van het beeld kunt u ofwel afzonderlijk op blank stellen door respectievelijk CLEAR GRAPHICS en CLEAR TEXT, ofwel tegelijkertijd door het commando CLEAR SCREEN.

Wanneer u DISPLAY TEXT tikt, zal het hele scherm weer beschikbaar zijn als 'tekst'-pagina. Zo schakelt u met DISPLAY GRAPHICS weer over naar de grafische pagina zonder ook maar iets te wijzigen van datgene dat al op het scherm stond. Merk het verschil op tussen deze commando's en de simpele woordjes TEXT en GRAPHICS — die bewerkstelligen dat de 'tekst' — en de grafische pagina op blank gesteld worden.

Zodra u van "kanalen" en de meer verfijnde kenmerken van grafische gegevensverwerking op de hoogte bent, staat het u vrij zelf de afmetingen van de grafische pagina te bepalen en elk gedeelte van het scherm te gebruiken.

PUNTEN OF LIJNEN

Laten we de PLOT opdracht in het programma hierboven zo veranderen dat de regel lekker leest:

```
110     PLOT 100, 100
```

Draai het programma nu opnieuw. Een punt zal op het scherm verschijnen.

Voeg nu na 100, 100 een kommapunt toe en daarna:

```
115     PLOT 1000,700
```

... en draai het programma nog een keer. Wederom verschijnt een lijn. Verwijder nu de kommapunt uit regel 110 en draai het programma. Nu verschijnen twee punten. Hoe komt dat?

Het antwoord is dit: de kommapunt zorgt ervoor dat het videosignaal aan blijft staan. Staat het videosignaal "aan" en wordt er een lijn getrokken tussen twee punten, dan blijft er een zichtbare lijn op het scherm achter. Wilt u het videosignaal aan laten staan, dan is de kommapunt na een PLOT opdracht noodzakelijk.

Het videosignaal kunt u zich het best voorstellen als een trekpen. Een PLOT commando met coordinaten zal de pen op het papier zetten en op zijn minst een punt voortbrengen. Als u een lijn wilt trekken tussen twee schermposities, moet u beide coordinaten intikken en deze met een kommapunt van elkaar scheiden. Hierdoor wordt het commando als volgt gelezen: "Zet een punt in positie (100,100) en houd de pen op het papier, terwijl een rechte lijn getrokken wordt naar (1000,700)". Als u de kommapunt weg zou laten, zou onze fictieve pen weliswaar verplaatst worden maar niet het papier raken.

U kunt ook de commando's SET BEAM ON en SET BEAM OFF gebruiken om de pen op het papier te zetten of ervan af te halen.

Hieronder volgt een "mazelen"-programma:

```
100     RANDOMIZE
110     INPUT PROMPT "Hoeveel mazelen?":B
120     GRAPHICS
130     LET Z = 0
140     DO
150         LET X = RND (1279)
160         LET Y = RND (719)
170         PLOT X,Y
180         LET Z = Z + 1
```

```
190     LOOP UNTIL Z = B
200     END
```

Dit programma zal op willekeurige posities puntjes op het scherm zetten. Door 170 te veranderen in:

```
170     PLOT X,Y;
```

zou u de "mazelen" met elkaar kunnen verbinden.

Uiteraard kan elk grafisch commando opgenomen worden in de definitie van een functie. Het volgende voorbeeld laat zien dat u van tijd tot tijd verschillende schermposities in een PLOT commando kunt onderbrengen en onderling met elkaar verbinden door lijnen te trekken:

```
100     DEF DIAGRAM
110         GRAPHICS
120         PLOT 504,544;564;464;516;488;504;544;
            460;464;516;448
130     END DEF
```

Draai dit en tik daarna CALL DIAGRAM in de rechtstreekse modus.

Als u achter een tweetal coordinaten een komma plaatst zal er in die positie geen punt komen te staan; het videosignaal zal alleen naar die positie gebracht en vervolgens uitgeschakeld worden. We zullen nog wel zien dat dit noodzakelijk is als u de instructie PLOT PAINT wilt geven.

TURTLE COMMANDO'S

Nu zullen we een aantal andere commando's bekijken waarmee u eveneens lijnen kunt trekken. We hebben hier te maken met de zogenaamde "turtle" commando's (turtle: zeeschildpad) die aanvankelijk gebruikt werden om een traag-bewegend robotdier te besturen. Met deze commando's hoeven we niet de coordinaten van een hele reeks schermposities uit te rekenen.

```
100     OPTION ANGLE DEGREES
110     GRAPHICS
120     PLOT 300, 150;
130     PLOT ANGLE 80;
140     PLOT FORWARD 500;
```

```
150     PLOT BACK 320;
160     PLOT RIGHT 35;
170     PLOT FORWARD 420;
180     PLOT BACK 285;
190     PLOT RIGHT 100;
200     PLOT FORWARD 340
210     END
```

Het zal u zeker wel zijn opgevallen dat dit veel weg heeft van het rondleiden van een dier over het scherm. Niettemin is bij enkele programmaregels enige tekst en uitleg nodig.

Regel 100 laat de computer weten dat we de hoeken in graden gaan meten en niet in radialen. (Een radiaal is bij benadering 57 graden; er zijn enkele wiskundige bewerkingen waarbij radialen beter van pas zouden komen.)

De GRAPHICS opdracht in regel 110 bewerkstelligt dat het videosignaal naar positie 0,0 verplaatst en vervolgens uitgeschakeld wordt. (CLEAR GRAPHICS zou hetzelfde effect hebben.) Regel 120 is dus nodig om de beginpositie aan te geven van waaruit het videosignaal (ons "dier") zich in beweging moet zetten.

Hierna moeten we het dier in de juiste richting sturen. Door het commando PLOT ANGLE 0 zou het op zijn plaats blijven en uitkijken op de rechterkant van het scherm.

PLOT ANGLE 90 zou het dier recht omhoog sturen. Het PLOT ANGLE commando in ons voorbeeld luidt: "beschouw het videosignaal als zijnde parallel aan de onderkant van het scherm en draai het vervolgens het aangegeven aantal graden tegen de wijzers van de klok in".

Een PLOT FORWARD of een PLOT BACK commando wordt gevolgd door het vereiste aantal grafische schermposities. PLOT RIGHT of PLOT LEFT zorgt ervoor dat het dier een andere richting inslaat en deze commando's worden gevolgd door het aantal graden die de hoek van draaiing aangeven (ten opzichte van de laatste richting).

Dit zijn we al eerder in een programma tegengekomen.

Onthoud dat u ook bij "turtle" commando's kommapunten moet gebruiken om het videosignaal ingeschakeld te houden. (Kijk maar eens wat er

gebeurt als u enkele kommapunten uit het laatste programma verwijdert of ze vervangt door komma's.)

ELLIPSEN EN CIRKELS

Het volgende programma tekent een ellips:

```
100     GRAPHICS
110     PLOT 640,250,
120     PLOT ELLIPSE 100,200,
130     END
```

Regel 110 geeft het middelpunt van de ellips. Het eerste getal achter PLOT ELLIPSE geeft de horizontale afstand aan (het aantal schermposities) tussen het middelpunt en de omtrek van de ellips; het tweede getal geeft de verticale afstand aan. Als deze twee getallen hetzelfde zouden zijn, zou het programma een cirkel tekenen. Merk de komma's op aan het eind van regel 110 en 120. Als u een hiervan weg zou laten zou het middelpunt van de ellips gemarkeerd worden door een punt op het scherm. Nu evenwel blijft het videosignaal in deze positie en wordt uitgeschakeld.

KLEUREN

Waarschijnlijk weet u al dat de Enterprise 256 kleuren kan weergeven. Maar tot nu toe heeft u nog niet veel kans gehad om ze ook te gebruiken. Hier zullen we u laten zien hoe u de vele kleurschakeringen onder de knie kunt krijgen die de Enterprise in beeld kan brengen.

Met het volgende programma kunt u alle 256 kleuren tegelijkertijd weergeven:

```
100     !
110     GRAPHICS 256 !
120     !
130     !
140     LET Z = 0
150     FOR Y = 0 TO 560 STEP 80
160           FOR X = 32 TO 1052 STEP 32
170             SET INK Z
180             PLOT X, Y; X, Y + 70
190             LET Z = Z + 1
200           NEXT X
210     NEXT Y
220     END
```

Let op het getal dat deze keer op GRAPHICS volgt.

De Enterprise identificeert elke kleur via een codenummer dat tussen Ø en 255 ligt. Voor het gemak is er een speciale functie, RGB, waarmee u kleuren kunt selecteren door hoeveelheden rood, groen en blauw met elkaar te vermengen. Dit komt uitvoeriger aan de orde verderop in dit hoofdstuk.

KLEURMODI

In het bovenstaande programma moest u — om alle kleuren ineens ter beschikking te hebben — het getal 256 na GRAPHICS tikken. Anders gezegd: u moest de passende *kleurmodus* kiezen.

Merk op dat in dit programma de lijnen op het scherm dikker zijn dan die welke u al eerder bent tegengekomen in grafische programma's. Dit houdt hiermee verband, dat uw tekeningen minder fijn worden naarmate u meer kleuren ter beschikking hebt. Dit om geheugenruimte in de computer te besparen.

We hebben al gezien dat de 'grafische' pagina een groter 'scheidingsvermogen' (mate van precisie) kent dan een 'tekst' — pagina. Maar nu moeten we nog een aantal verschillen in scheidingsvermogen onder de loep nemen die verband houden met de kleurmodus waarvan gebruik gemaakt wordt. We onderscheiden vier kleurmodi met een hoog scheidingsvermogen (HIRES), waarvan elk een specifieke verhouding biedt tussen het aantal kleuren dat weergegeven kan worden, en het aantal 'puntjes' per horizontale rij die voor tekenen beschikbaar zijn. Hieronder volgen de kleurmodi:

GRAPHICS HIRES 2 — wanneer u dit commando hebt gegeven, kunnen slechts 2 kleuren per keer in beeld gebracht worden. Maar er staan u wel 64Ø afzonderlijke puntjes per horizontale rij ter beschikking.

GRAPHICS HIRES 4 — 4 kleuren we 16Ø puntjes per regel.

GRAPHICS HIRES 16 — nu hebben we 16Ø puntjes per regel.

GRAPHICS HIRES 256 — in een keer kunt u wel 256 kleuren in beeld brengen en u heeft de beschikking over 8Ø afzonderlijke puntjes per regel.

Wanneer u alleen GRAPHICS zonder kleurmodus tikt, krijgt u GRAPHICS HIRES 256. Tikt u GRAPHICS op zich voor het eerst, nadat de computer is ingeschakeld of 'reset', dan is GRAPHICS gelijk aan GRAPHICS HIRES 4.

De kleurmodus is niet van invloed op het aantal verticale puntjes; op de grafische standaardpagina (met de 4 regels tekst aan de onderkant) bedraagt dit aantal altijd 19Ø puntjes.

Ondanks de verschillen in scheidend vermogen maken alle kleurmodi gebruik van hetzelfde coordinatensysteem.

Met andere woorden: PLOT Ø,Ø;64Ø,36Ø zal altijd een lijn trekken vanuit de linker benedenhoek naar het midden van de "pagina", ofschoon de fijnheid van de tekening af zal hangen van de modus. (In feite zou hetzelfde coordinatenstelsel gebruikt kunnen worden voor een scheidend vermogen dat twee keer zo hoog ligt als dat van de Enterprise.)

GRAFICS MODI

Behalve HIRES zijn er ook nog twee ander grafische modi.

LORES is identiek aan HIRES, maar maakt van minder computergeheugen gebruik en stelt slechts de helft van het horizontale scheidingsvermogen ter beschikking. Kleurmodi worden op dezelfde manier gespecificeerd als met HIRES; voorbeeld:

GRAPHICS LORES 16

levert een grafische pagina met 16 kleuren en een laag scheidingsvermogen op.

De ATTRIBUTE modus is een speciale vorm van beeldweergave: een tussenweg tussen de tekstuele en grafische modus. Deze modus kan gebruikt worden voor het afdrukken van tekens of voor het uittekenen van commando's, en voorziet in 16 kleuren. Wilt u optimaal gebruik ervan maken, dan moet u uiterst voorzichtig omgaan met de ATTRIBUTE — 'vlag'.

De kleurmodus hoeft niet gespecificeerd te worden, het commando luidt:

GRAPHICS ATTRIBUTE

KLEUREN KIEZEN

Laten we weer terugkeren naar de kleurmodus die 256 kleuren ter beschikking stelt. Met deze modus kunt u van alles tekenen in elke gewenste kleur door de tekencommando's vooraf te laten gaan door het commando SET INK en het codenummer van de kleur. Voordat u gebruik maakt van CLEAR GRAPHICS kunt u ook SET PAPER tikken om de kleur van de "achtergrond" te kiezen.

Door oefening komt u erachter welke kleur met

welk nummer correspondeert: 18 is lichtgroen, 91 is helder geel, enz. Maar als u een speciale kleur wilt gebruiken waarvan u de code niet kent, heeft u ook nog een alternatieve manier om kleuren te specificeren.

Elke mogelijke kleur kan verkregen worden door rood, groen en blauw met elkaar te combineren. Wit krijgt u bij voorbeeld wanneer u deze drie kleuren mengt. Geel wordt verkregen door alleen rood en groen te mengen. (Dit verbaast u misschien maar bedenk wel dat het mengen van verf andere resultaten oplevert dan het mengen van lichtbronnen.) Zwart is helemaal geen kleur. Complexe kleuren kunt u verkrijgen door rood, groen en blauw in steeds wisselende verhoudingen te mengen.

Volgens dit principe functioneren de kleuren op de Enterprise; elke kleur kan gedefinieerd worden als een vermenging van bovengenoemde kleuren door het woord RGB te tikken, gevolgd door drie getallen (tussen haakjes) die met een komma van elkaar gescheiden worden. Deze getallen, die in het bereik van Ø — 1 moeten liggen, geven (achtereenvolgens) het percentage rood, groen en blauw aan in een door u gewenste vermenging.

SET INK RGB (1,.5,.5) levert roze als tekenkleur op. RGB (.4,.4,Ø) levert mat geel op; RGB (.6,.6,.4) is een grijze tint, enz.

De acht kleuren hieronder kunt u heel eenvoudig selecteren door gewoon de naam ervan in te tikken (b.v. SET INK GREEN). De met die kleuren corresponderende codes zijn de volgende:

ZWART	= RGB (Ø,Ø,Ø)
ROOD	= RGB (1,Ø,Ø)
GROEN	= RGB (Ø,1,Ø)
GEEL	= RGB (1,1,Ø)
BLAUW	= RGB (Ø,Ø,1)
MAGENTA	= RGB (1,Ø,1)
CYAAN	= RGB (Ø,1,1)
WIT	= RGB (1,1,1)

HET PALET

Wanneer u het commando GRAPHICS HIRES 16 geeft, kunt u per keer slechts 16 kleuren in beeld brengen, maar u heeft dan wel veel vrijheid te bepalen welke kleuren dat zijn. Dit bereikt u door een "palet" samen te stellen — een lijst met geselecteerde kleuren die u ter

beschikking gesteld moeten worden.

Tik eerst SET PALETTE en lijst vervolgens acht kleuren uit die u wilt gaan gebruiken. Deze kleuren kunt u geheel naar believen kiezen uit de volledige scala van 256 kleuren en ze specificeren ofwel door hun standaardcode te gebruiken, met hun naam (als deze in bovenstaande lijst voorkomt) of door ze als een "menging" te definieren. U zou bij voorbeeld het volgende kunnen tikken:

```
10Ø    SET PALETTE 67,31,WHITE,
       4,RGB(Ø.,3,.8),RGB(.7,.7,.1),
       187,19Ø
```

— en deze kleuren zouden dan van Ø-7 genummerd worden in uw "palet".

Voor de overige acht kleuren heeft u minder keuzevrijheid. De kleuren die in uw palet van 8-15 zijn genummerd moeten alle tot een enkele groep verwante kleuren behoren. U kunt ze selecteren door middel van het commando SET BIAS, gevolgd door welk getal ook dat deel uitmaakt van de groep die u op het oog hebt. Wanneer u bij voorbeeld SET BIAS 67 tikt (of welk getal ook tussen 64 en 71), wordt de kleur met het standaard codenummer 64 (de eerste in deze groep) nummer 8 in uw palet; standaard nummer 65 wordt dan nummer 9, enz. U zult wel beseffen dat u met SET BIAS het "filter" op of de "vermenging" van de "primaire Teletekst" kleuren specificeert.

Nu kunt u elke mogelijke kleur van uw palet kiezen om de "verf aan te maken" voor de volgende lijn of figuur. Het volgende moet u onthouden: wanneer u geen gebruik maakt van de kleurmodus met 256 kleuren, verwijst elk commando zoals bij voorbeeld SET INK 6 of SET PAPER 6 naar de kleur met nummer 6 in het palet en niet naar de kleur die nummer 6 als standaard codenummer heeft. Onthoud ook dat de nummers in het palet bij Ø beginnen en niet bij 1.

In de 4-kleurenmodus kunnen alleen de kleuren die van Ø-3 in het palet zijn genummerd gebruikt worden; het heeft dus geen zin, meer dan 4 kleuren in een SET PALETTE commando op te nemen. Evenzeer zult u in de 2-kleurenmodus alleen de paletkleuren Ø en 1 kunnen specificeren.

Het kan wel eens voorkomen dat u slechts een

kleur in het palet wilt veranderen zonder de overige kleuren te wijzigen. In dit voorbeeld wordt paletkleur 3 veranderd:

```
SET COLOUR 3,110
```

Uiteraard kunt u ook elke modus gebruiken zonder u druk te maken over uw palet. Als er niets door u gespecificeerd wordt, zal de computer altijd gebruik maken van voorgeprogrammeerde "systeemgekozen" kleuren.

TOEPASSING VAN HET PALET

Onthoud dit: als u de set kleuren in uw palet wilt veranderen, zal dat niet alleen van invloed zijn op de lijnen en figuren die u daarna gaat tekenen, maar ook op de lijnen en figuren die al op het scherm staan. Het vermogen om met slechts een enkel commando alle kleuren die in beeld zijn te wijzigen ligt ten grondslag aan enkele het meest opvallende en uiterst snel werkende grafische effecten.

In het volgende programma dat ellipsen met willekeurige afmetingen en kleuren tekent, zullen we ermee beginnen alle kleuren in het palet gelijk te schakelen; dus een lijn in de kleur van codenummer 2 zal er anders uitzien dan een lijn in de kleur met codenummer 3, en beide kunnen niet onderscheiden worden van het "papier". Het getekende zal in feite onzichtbaar zijn. Daarna, zodra een toets aangeslagen wordt, zal er niet meer getekend worden en zal het programma door telkens de inhoud van het palet te veranderen — de verschillende "verven" van kleur doen veranderen, zodat ze tegen elkaar en tegen de achtergrond afsteken.

Het programma eindigt met een oneindige lus — tenzij onderbroken zal de lus nooit eindigen. Wilt u de uitvoering stopzetten, sla dan de "stop"toets aan.

```
100     RANDOMIZE
140     GRAPHICS HIRES 4
150     SET PALETTE BLUE,BLUE,BLUE,BLUE,0,0,0,0
160     !  ------------------------------
170     !     Onzichtbare weergave
180     !  ------------------------------
190     DO
200           SET INK RND*3 + 1
```

```
210         PLOT 625,330,
220         PLOT ELLIPSE RND*500,RND*300,
230     LOOP WHILE INKEY$= " "
240     !
250     !   Breng een en ander in beeld zodra toets
255     !   aangeslagen wordt.
260     !
270     DO
280         SET PALETTE BLUE,BLUE,RED,GREEN
290         !
300         FOR X = 1 TO 500!     Vertraging van
310         NEXT X !              bijna 1 seconde.
320         !
330         SET PALETTE BLUE,RED,GREEN,BLUE
340         FOR X = 1 TO 500
350         NEXT X
360         SET PALETTE BLUE,GREEN,BLUE,RED
370         FOR X = 1 TO 500
380         NEXT X
390     LOOP
400     !
410     !   Gebruik de "stop"toets om het
415     !   programma stop te zetten.
420     !
```

Merk overigens op dat de PALETTE, de INK en de PAPER commando's ook gebruikt kunnen worden, wanneer u met een 'tekst'-pagina werkt. Zie voor meer details het Naslaggedeelte onder het kopje 'Video-opties'. Probeert u het onderstaande eens:

```
SET 102: COLOUR 1, MAGENTA
SET 102: INK 3
```

PLOT PAINT

Deze instructie bewerkstelligt dat een regelmatige figuur met de actuele kleur "verf" ingevuld wordt. Hieronder volgt een programma dat een cirkel binnen een andere cirkel tekent en vervolgens twee verschillende kleuren inkleurt:

```
100     GRAPHICS HIRES 4
110     SET PALETTE WHITE,YELLOW,BLUE
120     PLOT 400,400,
130     PLOT ELLIPSE 200,200,
140     PLOT ELLIPSE 80,80,
```

```
150    SET INK 3
160    PLOT PAINT
170    PLOT 400,250,
180    SET INK 2
190    PLOT PAINT
200    END
```

Het gebied dat ingekleurd wordt door PLOT PAINT bevat dus het videosignaal en is omgeven door een ononderbroken lijn die een andere kleur heeft dan de positie waar het videosignaal zich bevindt. Elke onderbreking van de lijn zal resulteren in het "uitlopen" van de verf, waardoor het hele scherm ingekleurd wordt. Merk de komma's op aan het einde van regel 130, 140 en 170; zo wordt voorkomen dat een punt verschijnt op de positie van het videosignaal. Anders had PLOT PAINT het bij deze punt gelaten.

DE LIJN: VORM EN MODUS

De LINE STYLE biedt u de mogelijkheid met verschillende soorten onderbroken lijnen te werken. Wanneer u bij voorbeeld SET LINE STYLE 10 tikt, zullen alle lijnen die al getekend zijn (totdat de vorm opnieuw ingesteld wordt) de vorm aannemen van langgerekte, dicht op elkaar volgende gedachtenstreepjes. SET LINE STYLE 9 geeft u lijnen die afwisselend uit streepjes en puntjes bestaan. We onderscheiden 14 verschillende lijnsoorten (die genummerd zijn vanaf 1) waarmee u kunt experimenteren.

Met het commando SET LINE MODE kunt u bepalen, op welke manier lijnen die op het scherm getekend worden, inwerken op de figuren die al op het scherm staan. In geval van lijnmodus 0 (de 'systeemgekozen' modus) zal elke nieuwe lijn, al aanwezige lijnen of figuren gewoonweg 'overschrijven'. Bij toepassing van de overige modi (van 1-7 genummerd) zullen de oude en de nieuwe 'verven' op de pagina zich op verschillende manieren vermengen en zo de tekenkleur opleveren; het Naslaggedeelte verstrekt meer bijzonderheden.

PAGINA'S EN KANALEN

Tot nu toe heeft u het commando GRAPHICS (of TEXT) gebruikt om een blanco "pagina" te verkrijgen waarop getekend (of geschreven) kon worden. Doorgaans zult u hiermee waarschijnlijk kunnen volstaan, maar naarmate u meer ervaring opdoet zult u misschien wel

gebruik willen maken van de grotere flexibiliteit die verkregen kan worden door "kanaal" - nummers te specificeren (meestel van 1 tot 1Ø0 genummerd) voor uw verschillende pagina's (tekstueel en grafisch). Het onderstaande kunt u het best lezen in combinatie met het hoofdstuk over "Kanalen" en met de desbetreffende delen van het "Naslaggedeelte" (zie bij voorbeeld "Video-opties" en het sleutelwoord OPEN).

Er zijn twee belangrijke voordelen wanneer u voor uw pagina's nieuwe "kanalen" opent:
(1) Meerdere verschillende pagina's die complete tekeningen of teksten bevatten, kunnen tegelijkertijd in het computer-geheugen ondergebracht worden; elk van deze pagina's kan met een commando in beeld gebracht worden.
(2) Het paginaformaat kan nader aangegeven worden. U kunt dus met een grafische voorstelling (bijna) het hele scherm opvullen (maar onthoud wel dat de statusregel boven in het beeld niet gebruikt kan worden) of u kunt geheugen uitsparen door het paginaformaat klein te houden.

In het volgende voorbeeld wordt een kleine "tekst"pagina gecreeerd die op het midden van het scherm komt te staan:

```
 9Ø    SET BORDER CYAN
1ØØ    SET VIDEO MODE Ø
11Ø    SET VIDEO COLOUR Ø
12Ø    SET VIDEO X 2Ø
13Ø    SET VIDEO Y 1Ø
14Ø    OPEN £1: "VIDEO:"
15Ø    DISPLAY £1:AT 7 FROM 1 TO 1Ø
16Ø    PRINT £1: "Een klein formaat "tekst"pagina..."
17Ø    END
```

Regel 14Ø wijst kanaal nummer 1 toe aan onze nieuwe videopagina. Maar eerst moeten de "videomodus", de "kleurmodus" en de afmetingen van de pagina gespecificeerd worden. Videomodus Ø is een tekstpagina met 4Ø kolommen; modus 1 is een grafische pagina; modus 2 is een tekstpagina met 8Ø kolommen.

VIDEO COLOUR selecteert de kleurmodus, en wel als volgt:

VIDEO COLOUR Ø — 2-kleurenmodus
VIDEO COLOUR 1 — 4-kleurenmodus
VIDEO COLOUR 2 — 16-kleurenmodus
VIDEO COLOUR 3 — 256-kleurenmodus

Een tekstpagina moet altijd in kleurmodus Ø; u heeft dan toch nog enige kleurkeuze.

De regels 12Ø en 13Ø geven de breedte en de lengte van de pagina, uitgedrukt in 'tekenposities'.

Regel 15Ø is een instructie waardoor het bovenste gedeelte van de pagina (gerekend als 1Ø tekenrijen) in beeld gebracht wordt, te beginnen bij rij 7 op het scherm. In dit geval wordt natuurlijk de volledige pagina in beeld gebracht.

Merk op dat het PRINT commando in regel 16Ø het kanaalnummer moet bevatten.

Desgewenst kan de lengte van de pagina groter zijn dan de hoogte van het scherm. (Het maximum bedraagt 255 teken-rijen.) Het onderstaande programma definieert een pagina die acht kolommen breed en 3Ø rijen lang is, en tekent er een ellips op; daarna wordt eerst het bovenste gedeelte van de tekening en vervolgens het onderste gedeelte in beeld gebracht.

```
100     SET VIDEO MODE 1
110     SET VIDEO COLOUR 2
120     SET VIDEO X 8
130     SET VIDEO Y 30
140     OPEN £1: "VIDEO:"
150     PLOT £1: 128,540,
160     PLOT £1:ELLIPSE 115,500,
170     DISPLAY £1:AT 10 FROM 21 TO 30
180     FOR X = 1 TO 1500
190     NEXT X
200     DISPLAY £1:AT 10 FROM 1 TO 10
210     END
```

In het volgende voorbeeld worden 3 "pagina's" gedefinieerd die tegelijkertijd op verschillende plaatsen op het scherm worden gebracht:

```
100     SET VIDEO MODE Ø
110     SET VIDEO COLOUR Ø
120     SET VIDEO X 42
```

```
130     SET VIDEO Y 8
140     !
150     OPEN £1: "VIDEO:" !          "Tekst" pagina.
160     !
170     SET VIDEO MODE 1
180     SET VIDEO COLOUR 3
190     !
200     OPEN £2: "VIDEO:" !          Grafische
205     !                            gegevensverwerking,
206     !                            256 kleuren
210     !
220     SET VIDEO COLOUR 1
230     !
240     OPEN £3: "VIDEO:" !          Grafische
245     !                            gegevensverwerking, 4
246     !                            kleuren.
250     !
260     DISPLAY £1: AT 9 FROM 1 TO 8
270     DISPLAY £2: AT 1 FROM 1 TO 8
280     DISPLAY £3: AT 17 FROM 1 TO 8
290     PRINT £1: "Tekst..."
300     SET £2: BEAM ON
310     PLOT £2: 100,100;
320     SET £3: BEAM ON
340     END
```

Nadat u dit heeft gedraaid zult u niet kunnen zien wat u tikt. Tik TEXT of sla functietoets 5 aan om gegevens weer normaal te kunnen verwerken.

Merk op dat elke "pagina" een eigen "palet" heeft, ofschoon op alle pagina's een SET BIAS commando van toepassing is.

DE RAND INSTELLEN

Tot slot kunt u een kleur kiezen voor de rand die het volledige zichtbare beeld omgeeft — de "werktafel" waarop de verschillende tekstuele of grafische "pagina's" liggen. Het commando SET BORDER staat helemaal los van alle "palet" commando's (die alleen van toepassing zijn op speciale pagina's); deze kleur moet dus ingesteld worden door het standaard codenummer, de naam van een kleur of door een 'menging' te specificeren. Voorbeeld: SET BORDER 255, SET BORDER WHITE of SET BORDER RGB (1,1,1) zal het volledige beeld met een witte rand omgeven.

Wanneer de kleur van de rand bepaald wordt en

kanaal 1Ø1 (het kanaal van de grafische standaard-pagina) niet open is, moet het commando gegeven worden in combinatie met het juiste kanaalnummer (meestal 1Ø2, het nummer voor de standaard 'tekst'-pagina — bij voorbeeld: SET £1Ø2: BORDER 116.

De Enterprise beschikt over een vooraf gedefinieerde verzameling tekens die te samen de standaard tekenset vormen. Hiervoor zijn de tekendefinities aangehouden van de ISO (International Standards Organisation), beter bekend als ASCII — wat staat voor American Standard Code for Information Interchange (Amerikaanse nationale standaardcode voor informatie-uitwisseling).

Elk teken binnen de tekenset heeft een codenummer dat tussen Ø en 127 ligt. U kunt naar een standaardteken verwijzen met het teken zelf, zoals bij voorbeeld in PRINT "N", of met de code ervan. PRINT CHR$(78) is hetzelfde als PRINT "N" - probeer maar.

De computer maakt gebruik van deze codes om naar de tekens in zijn binnenste te verwijzen; maar het is niet echt noodzakelijk dit te begrijpen tenzij u een vergevorderd programmeur bent. Hier volgt een eenvoudige uitleg: als u bij voorbeeld een toets aanslaat — aangenomen de "a" — zal de computer een signaal ontvangen dat laat weten dat deze toets is aangeslagen. De computer zal dan meteen de code van dat teken registreren (niet de vorm) en die code versturen naar het gedeelte van de computer dat het scherm bestuurt en de code omzet in details, zodat het teken in beeld gebracht kan worden.

Wat voor u telt is dat u een toets aanslaat en tegelijkertijd het teken ziet verschijnen. Dit bewijst maar weer eens hoe snel een computer werkt, vooral wanneer u bedenkt dat de hierboven beschreven bewerking in feite veel complexer is; elk gedeelte van die taak wordt weer gesplitst in veel kleinere bewerkingen.

Hieronder volgt een programma dat de volledige tekenset afdrukt en u de mogelijkheid biedt een teken in te tikken waarvan de ASCII-code afgedrukt zal worden:

```
100   FOR N = 33 TO 159
110   !
125   !   Er zijn 128 vooraf vastgestelde tekens.
130   !   Deze FOR/NEXT lus maakt het mogelijk
140   !   de "stuur" — tekens weg te laten. Deze
150   !   zijn van Ø-32 genummerd (32 staat voor
155   !   "spatie").
160   !
```

```
170         PRINT CHR$(N),
180     NEXT N
190     DO
195         DO
200           INPUT PROMPT "Sla een teken aan de
              code ervan zal verschijnen: ":C$
210           IF LEN(C$) > 1 THEN
220             PRINT "Slechts een teken per keer,
                a.u.b.!"
221             !  ------------------------------
222             !  Het kleine IF-blok ziet erop toe
223             !  dat u slechts een teken aanslaat.
224             !  Merk op dat DO/LOOPS dit het
225             !  hele programma door
226             !  controleren. LEN geeft de lengte
227             !  van een string (regel 210).
230             !  ------------------------------
235           END IF
240         LOOP WHILE LEN(C$) > 1
250         PRINT "De ASCII-code voor";C$
260         PRINT "is:";ORD(C$)
261         !  ------------------------------
262         !  ORD geeft de ASCII-code voor een
263         !  teken.
264         !  ------------------------------
265         DO
270           INPUT PROMPT "Meer tekens, j/n?
              ":A$
280         LOOP WHILE A$ < > "j" AND A$ < > "n"
290     LOOP WHILE A$= "j"
300     END
```

Zoals u kunt zien, zijn ORD en CHR$ tegengesteld aan elkaar, ORD geeft de ASCII-code voor een teken en CHR$ drukt een teken af voor een ASCII-code.

UW EIGEN TEKENS ONTWERPEN

Nu u het een en ander weet over ASCII-codes, zijn een paar woorden op hun plaats over het ontwerpen van uw eigen tekenvormen. Voor elke ASCII-code haalt de computer de bijbehorende vorm uit zijn geheugen op — deze vorm kunt u echter opnieuw definieren.

U moet zich voorstellen dat een teken is opgebouwd uit negen rijen van telkens acht kleine lantaarntjes. Om de vorm van een teken te verkrijgen zullen sommige lantaarntjes aangestoken worden en

andere niet. Zo krijgt u een patroon van minuscule puntjes die zo klein zijn en zo dicht bij elkaar zitten dat het wel lijkt of ze een geheel vormen.

Als u dus een teken (opnieuw) wilt ontwerpen, is het raadzaam een vierkant raster te tekenen van 8x9 hokjes. Daarna kunt u gestalte geven aan het teken door puntjes aan te brengen in de juiste vierkantjes.

Wilt u deze gegevens vervolgens verwerken, dan moet u zich voorstellen dat alle puntjes enen zijn en de open vierkantjes nullen. Voor elke rij kunt u zo een wisselend patroon van nullen en enen samenstellen. Het onderstaande programma ontwerpt een klein tekentje en drukt het vervolgens af.

```
100     NUMERIC N (1 TO 9)
110     !
120     !     Een standaardteken bestaat uit 8 puntjes
130     !     horizontaal en 9 rijen verticaal. Elke serie
140     !     van negen zulke getallen vormt de
150     !     definitie van het teken. Regels 280-300
160     !     slaan deze getallen op in de reeks N.
170     !
190     DATA 00111110
200     DATA 01000001
210     DATA 01010101
220     DATA 01000001
230     DATA 00100010
240     DATA 00010100
250     DATA 00001000
260     DATA 00000000
270     DATA 00000000
280     FOR ROW = 1 TO 9
290           READ N (ROW)
300     NEXT ROW
310     SET CHARACTER 63,BIN (N(1)), BIN(N(2)),
        BIN(N(3)), BIN (N(4)), BIN(N(5)), BIN(N(6)),
        BIN(N(7)), BIN(8)), BIN(N(9))
320     !
330     !     BIN draagt de computer op de enen en
340     !     nullen als binaire cijfers te behandelen.
350     !
360     PRINT "?"
370     PRINT CHR$ (63)
380     END
```

Regel 31Ø is de opdracht waardoor de gegevens voor uw teken opgeslagen worden. Met dit programma hebt u het vraagteken opnieuw gedefinieerd! Het getal 63 is de ASCII-code en al die nullen en enen zijn de puntjes en de open vierkantjes (of de brandende en de niet brandende lampjes) die te samen het teken vormen. Het teken wordt vervolgens afgedrukt door het commando PRINT "?" of PRINT CHR$(63). Probeer het vraagteken ook in de rechtstreekse modus te tikken.

Als u eenmaal een programma heeft gedraaid dat tekens opnieuw definieert, kunt u deze tekens blijven afdrukken totdat u CLEAR FONT hebt getikt of de reset-knop twee keer ingedrukt of de computer uitgezet.

Het zal u wel niet zijn ontgaan hoeveel pleizer u kunt hebben met deze door de gebruiker gedefinieerde tekens.

U zou bij voorbeeld het volledige alfabet opnieuw kunnen ontwerpen. Programmeren in gotisch of cursief schrift zou een volstrekt nieuwe dimensie aan uw computeractiviteiten kunnen toevoegen! Ruimtespelletjes zullen er een stuk levendiger op worden wanneer u gebruik maakt van een paar zelfgemaakte buitenaardse wezens (u zou bij voorbeeld een creatuur kunnen maken door verschillende tekens door elkaar af te drukken).

De demonstratiecassette bevat een ander programma dat u de mogelijkheid biedt uw eigen tekens te ontwerpen en op cassette vast te leggen.

Geluidseffecten zijn voor de meeste programma's, in het bijzonder spelletjes, van groot belang. "Ernstige" programma's kunnen van geluid gebruik maken om signalen te geven. Spelletjes hebben aanmerkelijk hogere eisen als het gaat om het tijdig toevoegen van de juiste geluiden.

De Enterprise biedt de mogelijkheid deze geluiden via uw koptelefoon of muziekinstallatie in stereo te beluisteren. Bent u dit van plan en wilt u niet tegelijkertijd de ingebouwde mono luidspreker horen, dan kunt u deze het zwijgen opleggen door SET SPEAKER OFF te tikken of "shift" in te drukken in combinatie met functietoets 7.

Er staan u twee BASIC sleutelwoorden ter beschikking om het geluid te regelen. Dit zijn SOUND en ENVELOPE. Een SOUND opdracht verstrekt de computer algemene informatie over de duur van een geluid, de toonhoogte waarop begonnen wordt, de maximale geluidssterkte en nog een paar dingen. Een ENVELOPE opdracht bestaat uit een reeks instructies die de verschillen in toonhoogte en volume nader aangeven, zoals we die terugvinden in het geluid gedurende de afspeeltijd.

DE SOUND OPDRACHT

Hieronder vindt u een voorbeeld van een SOUND opdracht:

```
100   SOUND PITCH 40, LEFT 127, RIGHT 191,
      DURATION 200, ENVELOPE 10
```

Laten we de verschillende bestanddelen die de programmaregel specificeert, een voor een bekijken.

Het getal achter PITCH geeft aan wat de hoogte is van de eerste toon die gespeeld gaat worden. In theorie kan dit elk getal zijn tussen Ø (waardoor het geluid in feite bijna onhoorbaar laag zou worden) en 127. Goede resultaten worden verkregen in het bereik tot ongeveer 83. Binnen dit bereik zal elke verhoging met 1 resulteren in een verhoging van de toonhoogte met een halve toon. Toonwaarde 37 is gelijk aan de "eengestreepte C". Als er geen toonwaarde aangegeven wordt, wordt 37 gebruikt als de "systeemgekozen" waarde.

Het volgende bestanddeel in de reeks bepaalt de geluidssterkte die naar de linker luidspreker gestuurd

wordt. Het getal achter LEFT Moet tussen Ø en 255 liggen.

Wanneer LEFT Ø aangegeven wordt zal de luidspreker zwijgen. LEFT 255 (de "systeemgekozen" waarde) maakt het grootste volume mogelijk dat de machine kan voortbrengen — mede afhankelijk van de verdere instructies die in de "envelop" zitten. In het laatste voorbeeld betekent LEFT 127 dat het signaal naar de linker luidspreker nooit meer dan de helft van het maximale volume kan bedragen, ongeacht de verdere inhoud van de envelop.

Zo geeft RIGHT het maximale volume aan dat naar de rechter luidspreker gaat. In ons voorbeeld ontvangt deze luidspreker driekwart van het volume.

Het getal achter DURATION geeft de duur van het signaal aan, uitgedrukt in "tellen" — een tel bedraagt eenvijftiende van een seconde; in dit geval duurt het signaal dus 4 seconden. (De "systeemgekozen" waarde is 5Ø tellen.)

ENVELOPE verwijst naar het nummer van de ENVELOPE opdracht die in conbinatie met deze SOUND instructie gebruikt moet worden. (Er is een ingebouwde envelop, nummer 255, die de systeemgekozen waarde is; geldige nummers voor uw eigen enveloppes liggen tussen Ø en 254.)

Onthoud dat al deze bestanddelen die op het sleutelwoord volgen in elke denkbare volgorde gerangschikt kunnen worden. Verderop zullen we nog ingaan op een paar andere dingen die hieraan toegevoegd kunnen worden. Omdat in alle gevallen gebruik gemaakt wordt van systeemgekozen waarden, zal het woord SOUND op zich ook effect hebben.

ENVELOPES

Hieronder volgt een voorbeeld van een envelop:

```
9Ø      ENVELOPE NUMBER 1Ø;2, 8, 63, 5Ø; Ø, 24,
        - 16, 1ØØ; - 5, 47, - 39, 5Ø
```

Ook hier wordt het sleutelwoord gevolgd door een nogal lange lijst met gegevens.

Laten we met het eerste getal beginnen: NUMBER 1Ø identificeert de envelop, zodat SOUND instructies ernaar kunnen verwijzen (zie hierboven).

Daarna verschijnt een puntkomma, gevolgd door een reeks van vier getallen (van elkaar gescheiden

door komma's). Deze getallen geven de veranderingen in volume en toonhoogte aan tijdens de eerste "fase" — de eerste hoeveelheid tijd die aan het signaal is toegewezen (in dit geval duurt de fase 1 seconde).

Het eerste getal uit de reeks duidt aan dat de toonhoogte in deze fase met twee halve tonen zal toenemen ten opzichte van de beginwaarde, zoals aangegeven door de SOUND instructie. Zou er -1,5 hebben gestaan, dan zou de toonhoogte met drie kwarttonen zijn gedaald; Ø geeft aan dat de toonhoogte gelijk blijft, etc.

Het volgende getal (8) specificeert de verandering in volume, en wel voor de linker luidspreker. In een ENVELOPE opdracht is de eenheid van volume gelijk aan het vierenzestigste deel van het maximale volume dat toegelaten wordt tot de luidspreker — d.w.z. een vierenzestigste deel van het volume dat aangegeven wordt door de SOUND opdracht. Het getal 63 zal het volume altijd op het maximum instellen, -63 zal het volume helemaal uitschakelen (elk getal dat meer dan 63 of minder dan -63 bedraagt wordt genegeerd).

Helemaal in het begin heeft het signaal natuurlijk absoluut geen volume. In ons geval zal dus tijdens de eerste fase van de envelop de geluidssterkte die naar de linker luidspreker gezonden wordt toenemen van nul tot een achtste van de maximale sterkte.

Het derde getal uit de reeks (63) heeft een soortgelijke functie als het tweede getal, met dit verschil dat het van toepassing is op de rechter luidspreker.

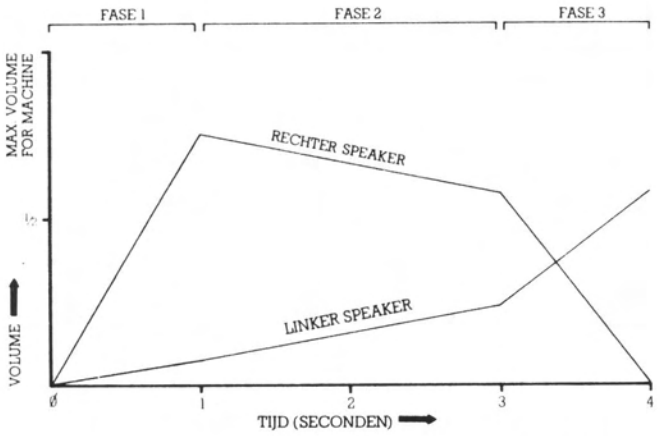
De SOUND en de ENVELOPE instructies genereren dan te samen telkens twee waarden die het volume bepalen — een waarde voor de linker en een waarde voor de rechter luidspreker. Wanneer u geen gebruik maakt van stereo-apparatuur, wordt het volume bepaald door beide afzonderlijke waarden gewoonweg samen te voegen.

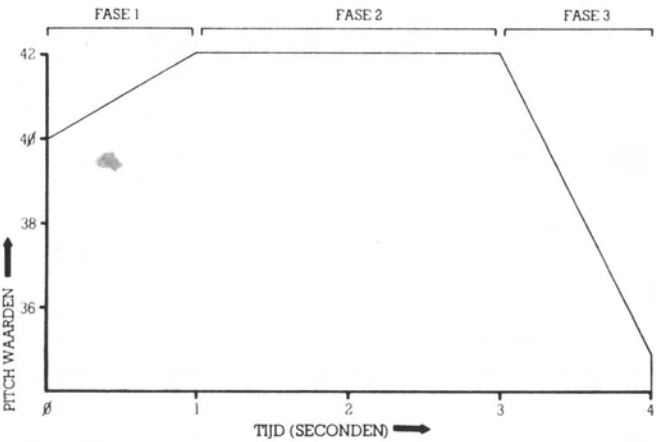
Het vierde getal uit de reeks (5Ø) geeft aan hoeveel tijd de eerste fase van de envelop in beslag neemt. Ook hier wordt tijd in "tellen" uitgedrukt; vijftig tellen zijn gelijk aan 1 seconde.

Vervolgens komen we aan bij een volgende reeks van vier getallen; deze getallen functioneren op dezelfde manier als die uit de eerste reeks, alleen zijn ze van toepassing op de tweede fase van de envelop.

De volgende vier getallen bepalen op hun beurt de derde fase nader.

Als u beide programmaregels die we nu hebben bekeken intikt en draait, zullen ze te samen een geluid voortbrengen dat als volgt door onderstaande grafieken weergegeven kan worden:





Uiteraard kunnen muzieksignalen veel complexer zijn dan het bovenstaande. Verderop zullen we zien dat de inhoud van een envelop kan bestaan uit wel 255 fasen.

Onthoud dat alle instructies die een envelop bevat betrekkelijk zijn: de uitwerking ervan is afhankelijk van

het volumebereik en de aanvankelijke toonhoogte, zoals nader aangegeven door de SOUND opdracht.

Onthoud ook dat de envelop gedefinieerd moet worden in een programma regel die eerder uitgevoerd wordt dan een SOUND opdracht die daarnaar verwijst.

WACHTLIJSTEN VOOR MUZIEKJES

Eventuele andere opdrachten die de computer moet uitvoeren zullen geen hinder ondervinden, wanneer een muziekje gespeeld wordt. In het onderstaande programma zullen commando's voor afdrukken en grafische gegevensverwerking tijdens de uitvoering begeleid worden door muziek:

```
100   ENVELOPE NUMBER 20; -2, 63, 63, 100; -3, 0,
      0, 100; 3, -36, -36, 100; 2, -12, -12, 100; 0, 0, 0,
      100
110   SOUND PITCH 61, LEFT 255, RIGHT 0
120   CLEAR SCREEN
130   PRINT AT 10,0: "Dadelijk zal er een grafiek
      getekend"
140   PRINT AT 12,0: "worden die de envelope
      weergeeft"
150   PRINT AT 14,0: "waarvan deze geluiden
      gebruik maken."
160   PRINT AT 16,0: 'Het volume wordt aangegeven
      door het"
170   PRINT AT 18,0: "blauwe gebied, de toonhoogte
      wordt"
180   PRINT AT 20,0: "aangegeven door de gele lijn."
190   SOUND PITCH 43, LEFT 127, RIGHT 127,
      DURATION 200, ENVELOPE 20
200   !
210   !     Het muziekje van regel 190 zal
220   !     ophouden na de eerste twee
230   !     fasen van de envelope.
240   !
250   FOR X = 1 TO 2000
260   NEXT
270   SOUND PITCH 25, LEFT 0, RIGHT 255,
      DURATION 500, ENVELOPE 20
280   GRAPHICS
290   PLOT 0, 0; 250, 540; 500, 540; 750, 270;
      1000, 180; 1250, 180; 1250, 0; 0, 0
300   PLOT 100, 20,
310   PLOT PAINT
```

```
32Ø     SET INK 3
33Ø     PLOT Ø, 45Ø; 25Ø, 35Ø; 5ØØ, 1ØØ; 75Ø, 35Ø;
        1ØØØ, 45Ø; 125Ø, 45Ø
34Ø     END
```

Bij regel 11Ø registreert de computer uw instructies en begint het muziekje te spelen; maar de computer wacht niet op het einde hiervan om verder te kunnen gaan met de volgende programmaregel. Het muziekje van regel 11Ø blijft spelen terwijl de computer zijn boodschap afdrukt (regel 13Ø-17Ø). In regel 18Ø verschijnt een nieuwe SOUND opdracht. Nu gebeurt het volgende: de computer registreert deze opdracht en zal het nieuwe muziekje gaan spelen zodra het vorige is afgelopen — met andere woorden: het muziekje dat in regel 18Ø wordt gespecificeerd, wordt op een "wachtlijst" geplaatst achter het muziekje van regel 11Ø. Hetzelfde geldt voor regel 25Ø; het derde muziekje wordt onderaan op de wachtlijst geplaatst achter het tweede. De grafiek wordt nu getekend en ondertussen volgt het ene muziekje het andere op.

RELEASE

De Enterprise maakt het mogelijk een of meer fasen aan het eind van een envelop enigszins anders dan de rest te behandelen. Deze fasen worden voorafgegaan door het woord RELEASE (van de rest losmaken). Doorgaans luidt het "release" stadium van een envelop een periode in waarin het geluid geleidelijk wegsterft — eigenlijk is de muziek dan afgelopen en zijn alleen de resterende effecten ervan gewenst.

Neemt dit voorbeeld maar eens:

```
1ØØ     ENVELOPE NUMBER 4;. 1, 63, 4Ø, 5; -.3,
        -32, -2Ø, 2Ø; 2, Ø, Ø, 25; RELEASE; Ø, -16,
        -1Ø, 1Ø; Ø, -15, -1Ø, 15
```

In dit voorbeeld duren de drie fasen voor het RELEASE stadium in totaal een seconde, de beide RELEASE fasen duren te samen een halve seconde. Het verschil tussen de non-RELEASE en de RELEASE fasen is dit: wanneer de "speelduur", zoals aangegeven in de SOUND opdracht, afgelopen is voordat de non-RELEASE fasen zijn beeindigd, worden deze afgebroken; maar de RELEASE fasen worden zelfs uitgevoerd wanneer de "speeltijd" voorbij is —

vooropgesteld dat er geen ander muziekje op de wachtlijst staat.

Als er dus sprake is van DURATION 25 in een SOUND opdracht die gebruik maakt van bovenstaande envelop, zal de derde fase niet uitgevoerd worden; de computer zal na de tweede fase direct de RELEASE fasen gaan uitvoeren — of het volgende, eventueel aanwezige muziekje op de wachtlijst.

Maar in geval van DURATION 60 zal de computer de drie non-RELEASE fasen en de eerste RELEASE fase uitvoeren — en zal daarna verder gaan met het volgende, eventueel wachtende muziekje.

In geval van DURATION 100 worden alle fasen uitgevoerd en zal er een pauze van een halve seconde zijn voordat het volgende muziekje begint.

ONDERBREKEN

In essentie weet u nu hoe u geluiden kunt voortbrengen; maar er zijn nog een paar andere interessante eigenschappen. Zo kunt u bij voorbeeld een muziekje onderbreken en plaats laten maken voor een ander. Laten we nog eens terug gaan naar het programma dat een grafiek tekende met op de achtergrond geluidseffecten. Verwijder de regels 130-180, 230-240 en 260-330 en voer in plaats daarvan in:

```
220     ENVELOPE NUMBER 30; 0, 63, 63, 1; 0, 0, 0, 24
230     PRINT "Sla de i aan als u dit muziekje wilt
        onderbreken (interrupt)."
240     DO
250     LOOP UNTIL INKEY$ = "i"
260     SOUND PITCH 37, LEFT 255, RIGHT 255,
        DURATION 25, ENVELOPE 30, INTERRUPT
```

De insluiting van INTERRUPT in regel 260 betekent, dat het muziekje dat gespeeld wordt door een ander onderbroken wordt zodra deze regel bereikt wordt — en alles dat op de wachtlijst staat wordt genegeerd.

GELUIDSBRONNEN

De mogelijkheid bestaat om per keer meer dan een muziekje te spelen. De Enterprise incorporeert vier toongeneratoren waarvan elk zijn eigen onafhankelijke 'wachtlijst' met muziek kan hebben.

Als u een muziekje op de wachtlijst van (bijvoorbeeld) toongenerator 2 wilt plaatsen, hoeft u alleen maar SOURCE 2 op te nemen in uw SOUND opdracht.

De toongeneratoren zijn van Ø-3 genummerd. Nummer 3 is de zogenaamde 'ruisgenerator' (die de toonwaarden negeert). Tot nu toe hebben we alleen toongenerator Ø gebruikt, de 'systeemgekozen' bron.

Wanneer u meerdere klanken tegelijkertijd wilt afspelen, wilt u er uiteraard van verzekerd zijn dat de verschillende geluidsbronnen gesynchroniseerd zijn. Door de instructie SYNC in een SOUND opdracht onder te brengen kunt u de ene klank op precies hetzelfde ogenblik laten beginnen als een andere klank — of als meerdere klanken, wat afhangt van het getal dat na SYNC is ingevoerd.

Voer een aantal SOUND opdrachten in en maak drie afzonderlijke 'wachtlijsten'. Voeg aan de eerste programmaregel van elke wachtlijst de instructie SYNC 2 toe, waardoor we iets dergelijks krijgen als hieronder:

```
12Ø    SOUND PITCH 4Ø, LEFT 255, RIGHT 127,
       DURATION 2Ø, ENVELOPE 6, SOURCE 1,
       SYNC 2
```

Wanneer de regels uitgevoerd worden (ervan uitgaande dat u ook de juiste enveloppes heeft ingevoerd), zullen alle drie wachtlijsten tegelijkertijd beginnen.

Preciezer uitgedrukt, de instructie SYNC 2 wordt als volgt geinterpreteerd: 'Zodra dit muziekje bovenaan de wachtlijst staat moet het tegengehouden worden, totdat nog twee muziekjes op andere wachtlijsten op punt van beginnen staan'. Dit houdt in dat een volgend muziekje dat op punt van beginnen staat eveneens automatisch tegengehouden wordt, totdat nog een ander muziekje bovenaan de wachtlijst staat. Pas dan komt beweging in alle drie de wachtlijsten.

Het commando CLEAR QUEUE, gevolgd door het nummer van een toongenerator, bewerkstelligt dat elk geluid dat door deze bron wordt voortgebracht uitgeschakeld wordt en alles op de wachtlijst geschrapt. Onthoud dat een SOUND opdracht met een INTERRUPT instructie geen klanken zal onderbreken die afkomstig zijn van andere 'bronnen'.

INGEWIKKELDER KLANKEN

Tenzij anders geïnstrueerd gaat de computer ervan uit, dat al uw muziekenveloppes tussen de 1 en 2Ø fasen bevatten. Maar u kunt het volgende tikken als u (bij-

voorbeeld) enveloppes wilt definieren met wel 25 fasen:

```
100   CLOSE 103
110   SET SOUND BUFFER 25
120   OPEN 103: "SOUND:"
```

De computer zal dan de extra geheugenruimte voor dit doel ter beschikking stellen. De bovenstaande regels sluiten en heropenen een 'kanaal'; wilt u weten hoe kanalen werken, zie dan het hoofdstuk hierover alsook het Naslaggedeelte (onder het sleutelwoord OPEN). Kanaal 103 is het gebruikelijke 'systeemgekozen' kanaal voor geluidsuitvoer.

Het aantal fasen die nader aangegeven worden door het SET SOUND BUFFER commando, kan van 1 tot 255 oplopen ofschoon in de praktijk de complexiteit van een envelop bepaald wordt door de maximale lengte van een BASIC programmaregel (250 tekens). SET SOUND BUFFER is alleen van toepassing op een kanaal dat vervolgens geopend wordt, niet op een kanaal dat al open is — dit verklaart de aanwezigheid van regel 100 en 120 in het laatste programma.

Tot slot kan een SOUND opdracht het woord STYLE bevatten, gevolgd door een nummer dat tussen 0 en 255 ligt. Vooral wanneer u gebruik maakt van de 'ruis'-generator (geluidsbron 3) is het interessant om te kijken wat de effecten zijn van de verschillende klankkleuren. Zie hiervoor het Naslaggedeelte, in het bijzonder de 'Geluidsopties'.

We hebben ons een hele tijd bezig gehouden het programmeren in verschillende onderdelen te splitsen en elk hiervan afzonderlijk te bekijken.

Maar we hebben ons nog niet afgevraagd hoe een taak vanuit de optiek van een computer bekeken zou moeten worden, of hoe taken op zo'n manier in een programma ondergebracht kunnen worden dat we dit ook naderhand nog gemakkelijk kunnen begrijpen. Dit alles zal nu aan de orde komen.

We zullen hier uitgaan van een eenvoudig probleem. Als u maar eenmaal de idee achter het plannen van programma's hebt gevat, lijkt het wel of u steeds moelijker problemen steeds gemakkelijker kunt oplossen.

HET PROBLEEM

We gaan ons bezighouden met de vraag wat bepaalde artikelen ons gaan kosten als we ze tegen een gereduceerde prijs krijgen.

Om deze vraag te beantwoorden hoeft u slechts twee dingen te weten: de oorspronkelijke prijs en de korting waartegen het artikel aangeboden wordt. Met behulp van deze twee cijfers kunt u gemakkelijk het bedrag uitrekenen dat van de oorspronkelijke prijs afgaat, en de eigenlijke prijs die u gaat betalen.

Het eerste dat u te doen staat wanneer u een programma schrijft is *precies* te bepalen, wat u met het programma wilt bereiken. In dit geval geldt het volgende:

(1) het programma gaat uit van twee door u ingevoerde getallen, de oude prijs en het kortingspercentage;
(2) het programma bepaalt welk bedrag gelijk is aan het percentage van de prijs;
(3) de korting wordt van de oude prijs afgetrokken om de feitelijke prijs te verkrijgen en
(4) het programma deelt u de uitkomsten mee en vraagt of u hetzelfde wilt doen met andere prijzen.

Dit spreekt voor zich. Het volgende stadium behelst de vraag hoe het programma dit karwei moet uitvoeren. Er zijn altijd verschillende manieren om een programma samen te stellen, maar ook enkele nuttige algemene regels.

MODULAIRE PROGRAMMA'S

Een keurig goedgeschreven programma zou modulair moeten zijn, d.w.z. duidelijk gestructureerd en

samenhangend. De reden hiervoor is deze: als u het programma eenmaal hebt geschreven moet u het kunnen begrijpen. Het heeft weinig zin om de ene na de andere regel BASIC te schrijven, als u later niet eens veranderingen kunt aanbrengen omdat u het programma niet begrijpt.

U kunt het het best als volgt zien: wanneer u een programma schrijft moet uw belangrijkste doel zijn, dat het programma langs de meest directe weg correct kan functioneren en dat u het snel kunt begrijpen zodra het geschreven is. Hoe duidelijker het programma geschreven wordt des te eenvoudiger zal het zijn om zowel de aanvankelijke problemen als de tekortkomingen van het programma eruit te pikken; de tekortkomingen zullen aan het licht komen als het programma al een tijdje in gebruik is.

De volgorde van een modulair programma ziet er in wezen als volgt uit:

(1) het benoemen van algemene variabelen;
(2) het hoofd- (besturings-) programma;
(3) een END instructie (die het punt markeert waar de computer stopt met de uitvoering van het programma; deze instructie hoeft in de numerieke volgorde niet de laatste regel te zijn);
(4) functies die zich bezighouden met gedeeltes van de hoofdtaak;
(5) indien gebruikt, DATA instructies (deze kunnen ook ergens aan het begin van het programma staan).

Wanneer u een structuur als deze aanhoudt zal het u aanmerkelijk minder moeite kosten om een complex en efficient programma te schrijven. Het zal u tijd schelen (en soms frustraties) wanneer u bij voorbeeld precies weet waar een functie geplaatst moet worden en waar u die later moet zoeken.

DE PROGRAMMA-ELEMENTEN

Algemene variabelen (die door het hele programma gebruikt worden) komen als eerste aan de beurt. Dit houdt verband met het volgende: als de computer bij een regel aankomt met een variabele die nog niet is benoemd, is het mogelijk dat hij stopt omdat het programma niet begrepen wordt. Als u de algemene variabelen dus allemaal tegelijk en meteen aan het begin benoemt, bent u van die zorg bevrijd. Bovendien hebt u zo alle belangrijke variabelen bij elkaar en weet u meteen waarvoor ze staan.

Dan hebben we ook nog de bijzondere variabele; deze wordt gebruikt in een functie en niet daarbuiten. De bijzondere variabelen worden pas benoemd als de functie geschreven wordt — en wel aan het begin daarvan.

Het hoofdprogramma moet om twee redenen op de tweede plaats komen. Ten eerste kan een programma gemakkelijker geschreven worden wanneer het gedeelte dat alle andere gedeeltes bestuurt ergens aan het begin staat. Ten tweede hoeft u zo alleen maar naar het begin van het programma te kijken, zelfs als het al een tijdje geleden is dat u het geschreven hebt, om in herinnering te roepen wat het programma bewerkstelligt. Het hoofdprogramma maakt bovendien gebruik van sommige, zoniet alle algemene variabelen die benoemd moeten zijn, voordat ze gebruikt worden.

Het hoofdgedeelte van het programma roept deelprogramma's aan en waarden van functies. Een functie kunt u beschouwen als een 'black box'. Het hoofdprogramma stopt getallen of strings in de 'black box' die dan nieuwe getallen en strings teruggeeft.

Een functie kan ook beschouwd worden als een afzonderlijk programma. Een functie kan ook weer eigen functies op een lager niveau bevatten. Hierdoor kunnen programma's samengesteld worden die een doorzichtige hierarchie hebben.

De DATA instructies hoeven niet per se aan het eind van een programma te staan, maar ze moeten wel allemaal op dezelfde duidelijk herkenbare plaats staan. Het kan behoorlijk vervelend zijn wanneer u een stel daarvan moet nalopen of moet tellen hoeveel er zijn. Als ze alle bij elkaar staan spaart u tijd, wanneer u op zoek moet gaan naar een foutief gegevensbestanddeel.

DATA instructies kunnen ook gebruikt worden binnen een functie, vooropgesteld dat ze alleen daarbinnen aangewend worden.

De END instructie markeert uiteraard het einde van het hoofd- (besturings-) programma. We hebben al eerder opgemerkt dat deze instructie niet altijd de laatste regel van een programma is. In geval van functies of subroutines springt de computer 'uit' het hoofdprogramma om er gebruik van te maken; van de gebruikelijke volgorde wordt nu afgeweken. De END instructie zou om die reden daar moeten staan waar het

besturingsprogramma eigenlijk ophoudt.

Vergeet ten slotte niet in al uw programma's veel commentaarregels op te nemen die zo verdeeld worden dat ze in het oog springen. Sommige programma's in deze handleiding bevatten veel meer commentaar regels dan regels met BASIC. De commentaarregels zorgen ervoor dat u precies weet wat het programma bewerkstelligt. Ze bieden u de mogelijkheid alles toe te lichten dat later misschien wel eens niet meer voor zich spreekt.

Denk nu niet dat programmeren met niets anders te maken heeft dan met ziekelijke netheid. Dat is zeker niet het geval. Maar zo goed als u zich bepaalde methoden eigen moest maken toen u leerde lezen en schrijven, evenzeer geldt dit voor programmeren. Dit hoofdstuk zal tijdens het leerproces telkens een hele steun zijn wanneer u een programma — hoe klein ook — op papier zet en invoert.

HET VOORBEELD

Laten we nu weer terugkeren naar ons prijsprobleem.

Eerst moet u bepalen wat het hoofdprogramma gaat doen. In dit geval zal het u enkel vragen de prijs van de artikelen en het kortingspercentage in te voeren; deze waarden worden doorgegeven aan het functie-gedeelte van het programma en daarna worden u de uitkomsten meegedeeld. Het hoofdprogramma wordt afgesloten met de vraag of u nog meer berekeningen wilt uitvoeren.

Hierna moet u uitrekenen hoeveel functies u nodig heeft. In ons voorbeeld heeft u slechts een functie nodig; deze kan zowel de vraag 'Hoeveel bedraagt de korting' als de vraag 'Hoeveel zal het gaan kosten' afhandelen — beide uiterst eenvoudige berekeningen. Laten we de functie 'DISCOUNT' (korting) noemen.

Als volgende stap moeten we uitwerken hoe de computer deze berekeningen uitvoert.

In dit programma geven we de computer een getal — bij voorbeeld 1Ø0 — en een percentage — aangenomen 20%.

Eerst moet de computer nagaan wat 20% van 1Ø0 is.

De computer zal eerst 1% uitrekenen — wat hetzelfde is als eenhonderdste — en de uitkomst met het percentage vermenigvuldigen. Onze berekening ziet er dus als volgt uit:

(prijs/1Ø0)*kortingspercentage

Daarna moet de machine uitrekenen wat de artikelen u gaan kosten; om hierachter te komen wordt de korting van de oorspronkelijke prijs afgetrokken.

Nu moet u min of meer weten welke variabelen we nodig hebben. Hieronder volgen ze (inclusief benaming):

- de oorspronkelijke prijs — PRICE
- het kortingspercentage — DISC
- de korting in ponden - TOTDISC (total discount)
- de gereduceerde prijs — NEWPRICE

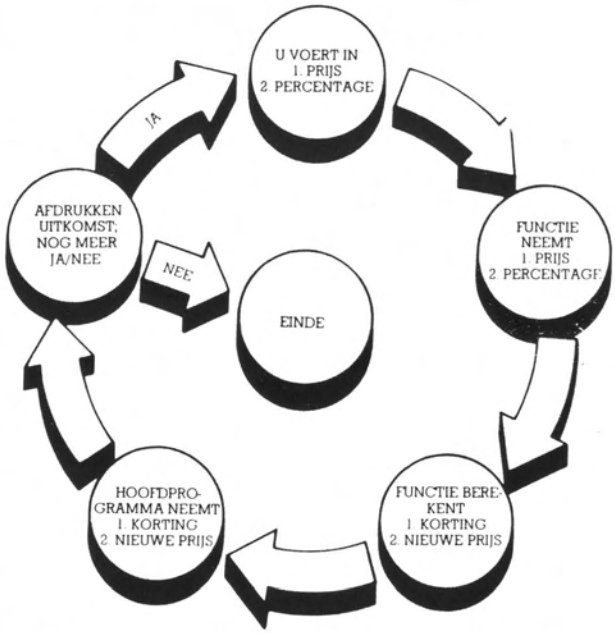
De eerste twee variabelen zullen ingevoerd worden in antwoord op een INPUT odracht. Deze hoeft u dus niet te benoemen. Nu weten we dus dat we slechts twee algemene variabelen aan het begin hoeven te benoemen.

De functie kan nu onmiddellijk geschreven worden. U hoeft de computer niet te laten weten dat u met percentages werkt — dat kan hij niet begrijpen. Een percentage is slechts een bepaald aantal honderdsten.

```
DEF DISCOUNT
   LET TOTDISC=(PRICE/1ØØ)*DISC
   LET NEWPRICE=PRICE—TOTDISC
END DEF
```

Nu we in grote lijnen weten in welke volgorde de programma-onderdelen staan, en we precies weten wat we willen bereiken kunnen we ermee beginnen het programma in BASIC om te zetten. Met een complexer programma zou u veel meer uit te werken hebben en waarschijnlijk heel wat BASIC eerst op papier moeten uitschrijven voordat u begon.

Voordat u ook maar iets invoert moet u eerst eens een blik werpen op onderstaande figuur:



Deze figuur laat u de volgorde zien waarin het eigenlijke BASIC geschreven wordt; de pijlen (te beginnen bij de cirkel 'U VOERT IN') geven de volgorde aan waarin het programma uitgevoerd wordt.

Voer eerst de te benoemen variabelen in:

```
100     NUMERIC TOTDISC, NEWPRICE
110     !       Dan het hoofdprogramma:
120     DO
130             CLEAR SCREEN
140             INPUT PROMPT "Voer a.u.b. de prijs
                in":PRICE
150             INPUT PROMPT "Voer a.u.b. het
                aangeboden kortings percentage in":
                DISC
160             CALL DISCOUNT
170             CLEAR SCREEN
180             PRINT "Een kortig van ";DISC;
                "procent op artikel geprijsd";PRICE
190             PRINT "zou bedragen ";TOTDISC;"."
200             PRINT
```

```
210         PRINT "De nieuwe prijs zou bedragen ";
            NEWPRICE;"."
215         PRINT
218         PRINT
220         INPUT PROMPT "Wilt u meerdere
            kortingen te weten komen?": ANS$
230     LOOP WHILE UCASE$(ANS$(1:1)) = "J"
240     END
```

Dit is het hoofdprogramma. Nu hoeft u alleen nog maar de functie toe te voegen:

```
250     DEF DISCOUNT
260         LET TOTDISC=(PRICE/100)*DISC
270         LET NEWPRICE=PRICE - TOTDISC
280     END DEF
```

Als u dit programma draait zult u zien dat het werkt, maar het is wel slechts het geraamte van een programma dat nog enige opvullig behoeft. Het onderstaande programma is in wezen hetzelfde, alleen is het veel keuriger gerangschikt dan datgene, dat u zo even heeft ingevoerd. Alles dat u niet begrijpt moet op de ! regels komen te staan (daarvan zijn er voldoende!).

```
100     NUMERIC TOTDISC, NEWPRICE
120     CLEAR SCREEN
160     PRINT AT 9,5:"EEN PROGRAMMA DAT
        KORTINGEN"
170     PRINT AT 11,5: "UITREKENT."
180     !  ------------------------------------
190     !  160 en 170 drukken een titel af op het
200     !  midden van het scherm, wat mooier staat
205     !  dan boven in in het beeld. 220 en 230
206     !  zorgen ervoor dat de titel ongeveer 5
207     !  seconden in beeld blijft.
210     !  ------------------------------------
220     FOR X = 1 TO 2500
230     NEXT X
240     CLEAR SCREEN
310     !  ------------------------------------
320     !  360 tot 440 vormen het hoofdprogramma.
325     !  Ze besturen de DISCOUNT functie. Het
330     !  hoofd programma is het gedeelte via
```

```
335     !     welk het programma indien gewenst
340     !     verlaten kan worden.
350     !     ------------------------------------
360     DO
365           CLEAR SCREEN
370           INPUT PROMPT "Voer a.u.b. de prijs van
              de artikelen in  FL.":PRICE
375           PRINT
380           INPUT PROMPT "Voer a.u.b. het
              aangeboden kortings percentage
              in":DISC
390           CALL DISCOUNT
400           PRINT "Een korting van ";DISC;
              "procent op artikel geprijsal FL.";PRICE
405           PRINT
410           PRINT "zou bedragen FL."; TOTDISC;"."
415           PRINT
420           PRINT "De nieuwe prijs zou bedragen FL.";
              NEWPRICE; "."
425           PRINT
430           INPUT PROMPT "Wilt u nog meer
              kortingen te weten komen?":ANS$
440     LOOP WHILE UCASES$(ANS$(1:1)) = "J"
445     END
450     DEF DISCOUNT
460           LET TOTDISC=(PRICE/100)*DISC
470           LET NEWPRICE=PRICE - TOTDISC
480     END DEF
```

Het programma hierboven heeft een kleine tekortkoming. Waar het u vraagt of u nog meer kortingen wilt uitrekenen, is het programma niet berekend op een foutieve invoer. Het kan al snel gebeuren dat u bij voorbeeld een 't' tikt in plaats van 'j', wat zou resulteren in beeindiging van het programma. Een goed programma moet berekend zijn op fouten van de kant van de gebruiker.

Hierdoor zal er geen wezenlijk verschil optreden met onze programma's tot nu toe, maar dat zal wel het geval zijn als de programma's langer en gecompliceerder worden. Het is uitermate frustrerend wanneer u een tikfout maakt die ofwel het programma voortijdig beeindigt ofwel alles vast laat lopen.

Het laatste wordt een 'crash' genoemd. Een programma 'crasht' wanneer iets zo verkeerd loopt dat

de computer het programma niet langer uitvoert en overschakelt naar de rechtstreekse modus, met als gevolg dat het programma opnieuw gedraaid moet worden. Deze toestand kan door van alles en nog wat veroorzaakt worden; meestal is er een programma debet aan dat iets probeert uit te voeren dat niet is toegestaan in BASIC. Dit wordt een fatale fout genoemd.

Het onderstaande programma heeft niet voor alles een oplossing maar het laat u wel zien hoe u kunt voorkomen dat een programma voortijdig eindigt of niet langer uitgevoerd kan worden omdat u iets verkeerds heeft ingevoerd.

```
100     DO
110         !  ------------------------------------
120         !  Het hele programma is een lange
125         !  DO/LOOP. In 480 tot 500 zult u nog
130         !  een DO/LOOP aantreffen, en wel
140         !  binnen de andere. Zolang de ene
145         !  lus maar binnen de andere is
150         !  gesitueerd, is dat prima. Een lus
160         !  binnen een andere lus is een
165         !  geneste lus. 175 tot 290 is
170         !  eveneens een geneste lus.
172         !  ------------------------------------
175         DO
180             INPUT PROMPT "Voer een heel
                getal in dat tussen 1 en 5 ligt:
                ":NUM
190             !  ------------------------------
200             !    180 spreekt voor zich. 290 ziet
205             !    erop toe dat het getal klopt:
210             !    eerst wordt gecontroleerd of
220             !    het getal niet groter is dan 5 of
225             !    kleiner dan 1, en daarna of het
230             !    een heel getal is — het laatste
235             !    wordt bereikt door ervoor te
240             !    zorgen dat het variabele NUM
245             !    niet groter of kleiner is dan
250             !    een heel getal. Als NUM
260             !    aanvaardbaar is, wordt de
280             !    LOOP niet herhaald.
285             !  ------------------------------
290         LOOP WHILE NUM > 5 OR NUM < 1 OR
```

```
            NUM < > INT(NUM)
300         SELECT CASE NUM
310         CASE 1
320             PRINT "Getal 1"
330         CASE 2
340             PRINT "Getal 2"
350         CASE 3
360             PRINT "Getal 3"
370         CASE 4
380             PRINT "Getal 4"
390         CASE 5
400             PRINT "Getal 5"
410         END SELECT
420         !-------------------------------------
430         !   Regels 300 tot 400 kiezen het juiste
435         !   antwoord op uw getal. Als het getal
440         !   niet 1 is, gaat de computer naar de
445         !   volgende regel om die te
450         !   controleren, enz. Zie voor SELECT
460         !   CASE het hoofdstuk "beslissingen".
470         !-------------------------------------
480         DO
490             INPUT PROMPT "Wilt u nog een
                getal proberen? ":ANS$
500         LOOP UNTIL UCASE$(ANS$(1:1))="J" OR
            UCASE$(ANS$(1:1)) ="N"
510         !-------------------------------------
520         !   480 tot 500 zijn de geneste lus. De
530         !   lus zal doorlopen blijven worden tot
540         !   de eerste letter van uw antwoord
550         !   — vertaald naar een hoofdletter —
555         !   "J" of "N" is. Dit is in wezen
560         !   hetzelfde als de controle die
565         !   hierboven toegepast werd op het
590         !   variabele NUM, maar de methode
600         !   is anders. 640 laat de computer
610         !   teruggaan naar het begin, zolang
620         !   de eerste letter van ANS$ "J" is. Dit
622         !   betekent dat de machine in geval
625         !   van "N" uit de lus stapt.
630         !-------------------------------------
640     LOOP WHILE UCASE$(ANS$(1:1))="J"
650     END
```

Bedenk wel dat de hier toegepaste methode slechts

een manier is om invoergegevens te controleren. Veel functies die door de computer ter beschikking gesteld worden, kunt u op de een of andere manier voor dat doel gebruiken. Het geheim van het samenstellen van efficiente programma's is gelegen in vindingrijkheid — inventiviteit ligt ook ten grondslag aan het plezier dat u kunt hebben.

Uw programma's zullen ten slotte veel prettiger overkomen, als u ervoor zorgt dat ze er netjes uitzien voor de persoon die ermee moet werken. Een programma 'netjes eruit laten zien' houdt in dat u het scherm telkens wanneer het vol raakt of wanneer het programma aan een nieuw stadium begint, (bij voorbeeld overgaan van invoer op de uitvoer van uitkomsten), schoon wist.

U kunt woorden ook een nettere aanblik op het scherm geven door gebruik te maken van PRINT AT, zodat u kantlijnen verkrijgt. UCASE$ of LCASE$ kunnen gebruikt worden om strings te ordenen die ingevoerd zijn door de persoon die van het programma gebruik maakt. Geluidseffecten en kleuren kunt u als signalen aanwenden.

Wanneer u enkele werkelijk goed gestructureerde, nuttige en interessante programma's hebt geschreven, zult u niet alleen tevreden zijn over uw werk, maar zult u zich ook kundig en volleerd voelen!

U zult wel weten dat BASIC evenals de gesproken talen vele dialecten kent. In het verleden hebben computerbedrijven hun eigen BASIC met eigen woorden en ideeën ontworpen. Maar de kern van de taal alsook de toepassingsmethode, zijn hetzelfde gebleven.

De Enterprise maakt gebruik van het Standaard BASIC, zoals voorgelegd door de European Computer Manufacturers' Association en het American National Standards Institute.

Het Standaard BASIC is ontworpen voor de afhandeling van problemen die zich voordoen, zodra een niet-gestandaardiseerde taal gebruikt wordt. Daarom kunnen programma's overgebracht worden van de Enterprise naar andere machines die gebruik maken van het Standaard BASIC. Dit dialect van BASIC biedt ook enkele uiterst krachtige karakteristieken die niet beschikbaar zijn in niet-gestandaardiseerde BASIC-dialecten.

Misschien wilt u wel programma's in niet-gestandaardiseerd BASIC draaien, zonder ze meteen helemaal door te hoeven nemen en tot in detail en op gecompliceerde wijze te veranderen. Gelukkig volgen de meeste BASIC-dialecten het zogenaamde 'Minimal BASIC'; dit is enkele jaren geleden ontworpen en toegepast door verschillende fabrikanten. Waarschijnlijk zult u nog wel horen van Microsoft BASIC; in deze taal vinden we de karakteristieken terug van het Minimal BASIC.

Als bijdrage aan de uitwisselbaarheid van BASIC-dialecten biedt de Enterprise de belangrijkste karakteristieken van Minimal BASIC, maar ook die van het Standard BASIC. Dit hoofdstuk gaat nader in op deze karakteristieken.

VERTAKKINGEN

Dit gaat geen les worden over hoe u uw bekroonde appelbomen moet snoeien. Maar het is wel een grote steun, als u zich de structuur van een programma even probeert voor te stellen als een boom (min of meer) — met een stam en kleinere takken.

Sommige bomen hebben talrijke takken en andere moeten het met slechts een paar doen. Op soortgelijke manier kunt u uw programma's structureren: ofwel wordt de ene taak na de andere afgewerkt, of u stelt uw programma zo samen dat u een 'stam' heeft met

verschillende 'takken' die direct aan de stam ontspringen. Bovenal doet u er verstanding aan niet van tak naar tak te springen. Hiermee moet voorzichtig omgegaan worden.

Twee manieren van vertakking in een programma zijn GOTO en GOSUB. De functies bieden een andere belangrijke methode van vertakken op deze computer (zie voor functies het betreffende hoofdstuk).

GOTO en GOSUB bewerkstelligen dat de computer, zodra een van deze twee woorden aangetroffen wordt, naar een ander gedeelte van het programma springt.

GOTO

GOTO maakt gebruik van een regelnummer om de computer mee te delen naar welke regel hij toe moet. Voorbeeld:

```
100     CLEAR SCREEN
110     INPUT PROMPT "Houdt u van computers?
        ":SAY$
120     IF UCASE$(SAY$(1:1))="J" THEN
130           GOTO 220
135           !          GOTO 220 stuurt het
136           !          programma naar regel 220
140     ELSE IF UCASE$(SAY$(1:1))="N" THEN
150           GOTO 240
155           !          GOTO 240 stuurt het
156           !          programma naar reactie op
160           !          een nee antwoord
190     ELSE
200           GOTO 110
205           !          Hierdoor begint het
206           !          programma van voren af aan,
207           !          als u niet met ja of nee
208           !          antwoord.
210     END IF
220     PRINT "Fijn, dat doet me goed!"
230     END
240     PRINT "Maar ik ben anders. Let maar op!"
250     END
```

GOTO stuurt de computer dus naar een regelnummer. GOTO kan ook als lus gebruikt worden. Als u regel 250 verandert in.

```
25Ø     INPUT PROMPT "Zullen we dat nog eens
        herhalen?":A$
```

en regel 26Ø toevoegt;

```
26Ø     IF UCASE$(A$(1:1))="J" THEN GOTO 1ØØ
```

(alsook regel 27Ø — END), zal het programma teruggaan naar het begin als uw antwoord 'ja' luidt. Dit komt op hetzelfde neer als wanneer het programma in de vorm van een DO/LOOP was gegoten; de laatste regel zou dan luiden: (tweede regel 1Ø5 DO)

```
26Ø     LOOP WHILE UCASE$(A$(1:1)) = "J"
```

Ofschoon de computer in beide gevallen niet precies hetzelfde te werk gaat, zal de uitkomst hetzelfde zijn.

Lange programma's kunt u veel beter met behulp van functies dan met GOTO in afzonderlijke delen splitsen.

GOTO is vooral zo onhandig, omdat daardoor het normale programmaverloop onderbroken wordt zonder een alternatieve structuur aan te bieden. Om deze reden gaan programma's verloren!

GOSUB

GOSUB is in zoverre gelijk aan GOTO dat de computer naar een regelnummer in een ander gedeelte van het programma gestuurd wordt; maar in geval van GOSUB verwacht je terug te keren.

Wanneer u gebruik maakt van GOSUB moet u een gedeelte van het programma als 'eenheid' apart zetten — een gedeelte dat slechts een ding hoeft te doen. Daarna moet u aan het einde van die eenheid het woord RETURN toevoegen; zo informeert u de computer dat hij terug moet gaan naar de regel die direct volgt op de regel met het woord GOSUB.

Het gedeelte van een programma tussen de regel die door een GOSUB commando aangegeven wordt, en het RETURN commando wordt een subroutine genoemd. Door gebruik te maken van het woord CALL kan een functie ook als subroutine functioneren.

HOE ZIJN FOUTEN TE VERMIJDEN

Als u niet de mist in wilt gaan of in verwarring wilt raken, moet u altijd twee dingen onthouden als u

gebruik maakt van GOSUB/RETURN.

Ten eerste mag een GOSUB commando nooit gebruikt worden zonder RETURN. Wanneer dit gebeurt deelt de computer u mee dat u een fout heeft gemaakt.

Ten tweede heeft u weliswaar de subroutine als een aparte eenheid terzijde gezet, maar voor de computer geldt dat niet. Er bestaat geen speciaal commando om het begin van een subroutine te markeren. Hierdoor kan de computer de subroutine per ongeluk gaan uitvoeren.

Als dit zich voordoet en de computer het woord RETURN leest zonder dat hij eerst naar de subroutine is gestuurd, zal het programma stopgezet worden omdat de computer het niet begrijpt.

Wilt u dit probleem met GOSUB voorkomen, dan kunt u het best al uw subroutines aan het natuurlijke einde van een programma plaatsen. Plaats direct voorafgaande aan het begin van de subroutine de END opdracht of het STOP commando. Zo wordt het programma beeindigd, voordat de computer de kans krijgt de subroutines weer te lezen.

Het volgende programma illustreert deze methode.

```
 80    CLEAR SCREEN
100    PRINT AT 1,5:"MENSEN EN HUN
       LEEFRUIMTE.":!
110    PRINT AT 2,5:"PROGRAMMA DAT DE
       VERHOUDING MENS/"
115    PRINT AT 3,5: "LEEFRUIMTE UITREKENT."
120    PRINT AT 5,5:"Dit programma berekent het"
125    PRINT AT 6,5:"gemiddelde aantal vierkante
       meters"
130    PRINT AT 7,5:"dat voor een ieder"
140    PRINT AT 8,5:"in een stad beschikbaar is."
150    PRINT AT 9,5:"Bedenk dat het slechts een"
160    PRINT AT 10,5:"gemiddelde is, maar als u"
170    PRINT AT 11,5:"meerdere steden met elkaar"
180    PRINT AT 12,5:"vergelijkt weet u snel genoeg"
183    FOR X = 1 TO 5000
184    NEXT X
187    CLEAR SCREEN
190    PRINT AT 13,5:"Voer a.u.b. de volgende gegevens"
195    PRINT AT 14,5:"in."
200    PRINT AT 16,5:"A) Naam van de stad."
210    PRINT AT 17,5:"B) Bevolking."
```

```
220   PRINT AT 18,5:"C) Oppervlakte van stad in"
225   PRINT AT 19,5: "     vierkante kilometers."
230   FOR X = 1 TO 5000
232   NEXT X
235   PRINT
240   INPUT PROMPT "STAD?":CITY$
250   INPUT PROMPT "BEVOLKING?":POP
260   INPUT PROMPT "OPPERVLAKTE?":SIZE
270   GOSUB 1000
350   CLEAR SCREEN
360   PRINT AT 10,5:CITY$;", bij een oppervlakte van"
370   PRINT AT 11,5:SIZE;"  vierkante kilometer,"
380   PRINT AT 12,5:"en een bevolking van  ":POP
390   PRINT
400   PRINT AT 13,5:"zou elke inwoner gemiddeld"
410   PRINT AT 15,15:SPA
440   PRINT AT 17,5:"vierkante meter ter beschikking"
445   PRINT AT 18,5:"hebben."
450   PRINT
455   PRINT
460   INPUT PROMPT "Wilt u dit voor nog meer steden
      uitrekenen?":ANS$
470   IF UCASE$(ANS$(1:1))="j" THEN GOTO 187
480   !  ------------------------------------------
490   !     470 maakt gebruik VAN GOTO om u
495   !     terug te sturen voor meer gegevens, als u
500   !     met 'ja' antwoordt.
510   !  ------------------------------------------
520   END
530   !  ------------------------------------------
540   !     De END opdracht in 520 voorkomt dat de
550   !     subroutine weer uitgevoerd wordt en dat
560   !     fouten gemaakt worden. Bij regel 1000
600   !     begint de subroutine.
610   !  ------------------------------------------
1000  LET SQY=SIZE*10^2
1010  LET SPA=SQY/POP
1080  RETURN
```

Nu heeft u dus kunnen zien hoe u GOSUB moet gebruiken. Maar het zal u wel zijn opgevallen dat dit lang niet zo ordelijk is als een progamma dat een functie aanroept — merk bij voorbeeld op dat er nergens, afgezien van de GOSUB regel, aangegeven

wordt waar de subroutine begint.

Verwacht kan worden dat programma's die gebruik maken van GOTO en GOSUB, eerder fout gaan dan programma's die gestructureerde lussen en functies gebruiken.

DE 'DIM' INSTRUCTIE

Als u terugblikt op het hoofdstuk over de opslag van grotere hoeveelheden informatie, zult u zich herinneren hoe u geheugengebieden kunt reserveren voor reeksen. Er is ook nog een minder veelzijdige manier waarop u dat kunt doen.

DIM A (1Ø) reserveert een een-dimensionale reeks waarvan de elementen van Ø tot 1Ø zijn genummerd. DIM A(4,4) zal een twee-dimensionale reeks reserveren waarop hetzelfde systeem van elementnummering van toepassing is. U kunt, evenals bij NUMERIC of STRING, (X TO Y) gebruiken om een aantal nummers te specificeren die u aan de elementen van uw reeks wilt geven, maar u kunt niet bepalen wat de lengte van elk string-element zal zijn. U zult wel beseffen hoe nuttig dit kan zijn, wanneer u erg lange programma's schrijft waarin gebruik gemaakt wordt van meerdere string-reeksen; op die manier kunt u geheugenruimte besparen.

De DIM instructie wordt alleen maar geboden vanwege de uitwisselbaarheid met andere BASIC-dialecten. Hieronder volgt een kort illustratie-programma. Merk op dat DIM ARRAY$(13) netzelfde is als STRING ARRAY$(13) — het laagste element is altijd Ø, tenzij anders aangegeven.

```
1ØØ    DIM ARRAY$(13)   !  Een reeks met 14
                           elementen
11Ø    FOR N=Ø TO 13
12Ø      READ ARRAY$(N)
13Ø    NEXT N
14Ø    FOR N=Ø TO 13
15Ø      PRINT ARRAY$(N);" ";
16Ø    NEXT N
17Ø    DATA Dit,is,een,reeks,die,benoemd,is,
18Ø    DATA met,behulp,van,de,DIM,instructie,
19Ø    DATA "—eenvoudig,niet?"
```

Opdat u op flexibelere wijze de computer en de 'apparatuur' die daarop aangesloten is kunt bedienen, maakt de computer gebruik van zogenaamde kanalen.

Een kanaal is een speciale baan die geopend wordt tussen twee onderdelen van de computer. Wanneer de baan eenmaal opengesteld is, kan de communicatie tussen beide onderdelen plaatsvinden door alleen maar het kanaalnummer te specificeren — in instructies voorafgegaan door het symbool '£' (of in sommige tekensets door "#".

Het BASIC op de Enterprise probeert zo min mogelijk te speculeren op de vraag hoe u de computer zult gaan gebruiken. De kracht van de computer vloeit immers voort uit een flexibel reageren op uw wensen. Daarom bieden alle 'invoer-uitvoer' instructies u de mogelijkheid om, indien gewenst, kanalen te specificeren.

Maar daar waar mogelijk zal de Enterprise 'systeemgekozen' kanalen aanwenden — van overbodige commando's zult u immers niet gediend zijn. Normaal gesproken wilt u, wanneer u bij voorbeeld PRINT, de woorden of de getallen op het scherm zien verschijnen; dus PRINT "Hallo" zal de boodschap in beeld brengen.

Maar als u in plaats daarvan de boodschap via de printer wilt afdrukken (vooropgesteld dat er een angesloten is), kunt u het commando PRINT £1Ø4: "Hallo" geven.

Wanneer u de computer aanzet, wordt kanaal 1Ø4 automatisch opengesteld voor de printer; elke boodschap die naar kanaal 1Ø4 gestuurd wordt, wordt dus op de printer gezet.

U kunt ook naar kanaalnummers verwijzen door gebruik te maken van variabelen; bijgevolg kan het programma zelf bepalen waar een boodschap naar toe gestuurd moet worden.

```
12Ø  INPUT PROMPT "Voer a.u.b. een boodschap
     in:":A$
13Ø  PRINT
14Ø  PRINT "Waar wilt u de boodschap herhaald
     zien?"
15Ø  DO
16Ø     INPUT PROMPT "Voer a.u.b. Ø voor scherm in
        of 1Ø4 voor printer: ":Kanaal
```

```
170  LOOP UNTIL kanaal = Ø OR kanaal = 1Ø4
190  PRINT £kanaal: ! Blanco regel naar scherm of
     printer.
200  PRINT £kanaal:A$ ! Boodschap naar scherm of
     printer
220  END
```

Dit programma kan uiteraard alleen uitgevoerd worden als u een printer heeft aangesloten die aan staat. U kunt hetzelfde experiment ook uitvoeren met kanaal 1Ø1 in plaats van kanaal 1Ø4 — hierdoor wordt u verbonden met het gebruikelijke beeldscherm voor grafische gegevensverwerking. Neem het commando GRAPHICS in uw programma op en zie wat er dan gebeurt.

Wanneer u gebruik maakt van kanalen moet u onthouden, dat alle invoer-uitvoer op deze manier werkt — met inbegrip van dergelijke dingen als de toongenerator en de aansluiting met de bandrecorders. Als u op een slordige manier omgaat met de kanaalnummers, zouden wel eens vreemde dingen met uw computer kunnen gebeuren! Waarschijnlijk zal dan geen blijvende schade toegebracht worden, maar het is mogelijk dat u uw programma opnieuw moet starten of de computer opnieuw instellen.

Voor meer bijzonderheden over kanalen verwijzen we naar het hoofdstuk over commando's in het Naslaggedeelte (zie met name het trefwoord OPEN).

Het BASIC op de Enterprise voorziet in functies voor de afhandeling van "uitzonderingen" om fouten aan te kunnen pakken, om het netwerk of bepaalde andere apparaten te kunnen bedienen, maar ook om met enkele speciale omstandigheden uit de voeten te kunnen die los staan van het normale programmaverloop.

Functies voor de afhandeling van uitzonderingen lijken enigszins op functies. Maar in tegenstelling tot functies worden ze doorgaans niet geactiveerd door een oproep, als CALL in het programma.

Eerst enige tekst en uitleg. Een uitzondering is iets dat onafhankelijk van een lopend programma plaatsvindt; het kan wel van invloed zijn op het programma of juist bedoeld zijn om het programma tot op zekere hoogte te beïnvloeden. De 'stop' toets is een uitzondering — het programma hoeft immers niet te controleren of u deze toets heeft aangeslagen, maar het wordt er wel door beïnvloed. Programmeerfouten zijn eveneens uitzonderingen.

Wanneer u een programma draait dat een of andere fout bevat, zal de computer — indien de fout tot een type behoort waardoor het programma gestopt wordt — daarop reageren met een korte boodschap en een nummer. In dit geval kan het nummer als een 'uitzonderingstype' gebruikt worden, zodat de computer een aanwijzing kan geven wat er moet gebeuren.

BASIC biedt ook de mogelijkheid uw eigen uitzonderingen te maken. Deze moeten genummerd zijn van 1 tot 999 en kunnen toegepast worden in verband met verkeerd getikte invoer (bij voorbeeld een te groot of een te klein getal) of een andere voor een programma ongewone situatie.

Laten we eens kijken hoe uitzonderingen afgehandeld worden: eerst veroorzaken we een uitzondering in een programma en daarna pakken we deze aan door gebruik te maken van een functie voor de afhandeling van uitzonderingen.

```
50    WHEN EXCEPTION USE INPUT_ERROR
60    !
65    !    50 draagt de computer op de
70    !    'afhandelingsfunctie' te gebruiken (zie
75    !    verderop) als een uitzondering zich
```

```
 80    !     voordoet. Geldig totdat de computer END
 85    !     WHEN bereikt IN 145.
 90    !     ------------------------------------
100    INPUT PROMPT "Tik a.u.b. een woord: ":
       STRING$
110    IF VAL (STRING$) < > 0 THEN
115    CAUSE EXCEPTION 10
120    ELSE
130         PRINT "Uw woord is geaccepteerd."
140    END IF
145    END WHEN
150    END
160    !     ------------------------------------
170    !     260 tot 310 is het gedeelte van het
180    !     programma dat zich bezighoudt met een
190    !     uitzondering; in dit geval uitzondering
200    !     nummer 10. Deze wordt veroorzaakt
210    !     wanneer u een getal tikt in plaats van een
220    !     woord. Wellicht ziet u enkele parallellen
230    !     tussen de 'afhandelingsfunctie' en het
240    !     aanroepen van functies.
250    !     ------------------------------------
260    HANDLER INPUT_ERROR
270         IF EXTYPE = 10 THEN
280          PRINT "Dit is geen woord."
285         ELSE IF EXTYPE < > 10 THEN
290          EXIT HANDLER
300         END IF
310    END HANDLER
```

Waarschijnlijk kunt u wel zien dat u dit ook met andere methoden had kunnen doen. In overeenstemming met uw instructies laat regel 110 de computer een 'fout' registreren.

CAUSE EXCEPTION stuurt het programma naar een functie voor de afhandeling van uitzonderingen voor het geval zich een fout voordoet die normaliter niet door de computer herkend zou worden — zoals hierboven bij voorbeeld een woord dat met een getal begint. Normaal accepteert de computer getallen als string, maar doordat we VAL gebruiken in combinatie met de instructie CAUSE EXCEPTION, wordt een string die met een getal begint als fout gezien.

De VAL (waarde) van een string is 0 als er binnen de string geen geldig getal (behalve 0) aan de eerste

letter of het eerste niet-numerieke teken voorafgaat.

EXIT HANDLER komt op hetzelfde neer als EXIT DO of EXIT FOR, alleen hebben die woorden betrekking op het HANDLER-blok.

EXTYPE is het typenummer van de uitzondering — in het programma is dit nummer 1Ø. Naargelang datgene dat fout is gegaan en — in geval van een CAUSE EXCEPTION opdracht — het nummer dat u aan de uitzondering geeft die veroorzaakt gaat worden, verschilt het EXTYPE. Zie pagina 236 voor foutboodschappen. Weer een ander woord, EXLINE, geeft het regelnummer aan waar de uitzondering zich heeft voorgedaan.

Het laatste programma heeft een bepaalde manier voor de afhandeling van uitzonderingen geïllustreerd. In wezen lijkt dit veel op het aanroepen van een functie (die als subroutine of als moduul gebruikt wordt). Merk op dat deze methode gebruik maakte van de woorden WHEN EXCEPTION USE, wat zoveel betekent als 'wanneer ergens een fout optreedt moeten deze functies voor de afhandeling van uitzonderingen gebruikt worden. Dit commando moet met een END WHEN instructie afgerond worden.

De Enterprise is in staat via een gewone draadverbinding met andere computers te communiceren. De andere computers moeten natuurlijk over dezelfde faciliteit beschikken (bekend als het "Intelligent Netwerk") als ze hun deel van het gesprek voor hun rekening gaan nemen; verbindingen met andere Enterprise computers vormen geen probleem.

Het voordeel van een netwerk is dat tal van computers op elkaar aangesloten kunnen worden; maar wanneer een computer met een andere wil spreken kunnen de overige op het netwerk aangesloten computers het gesprek niet volgen — net zoals bij een telefoonnet.

Uw computer heeft niet zoals een telefoon een van tevoren vastgesteld nummer. Wanneer u in verbinding treedt met het netwerk moet u een nummer kiezen, bij voorbeeld:

```
SET NET NUMBER 5
```

Het netnummer kan alles zijn van 1 tot 32. U kunt ASK NET NUMBER gebrukken om uzelf te herinneren aan het nummer dat u gekozen hebt.

PROGRAMMA'S OVERBRENGEN

Als uw computer eenmaal een netnummer heeft, is het een eenvoudige zaak programma's van de ene naar de andere machine over te brengen.

Tik op de computer die het programma gaat ontvangen, LOAD "NET-Ø:". Hierdoor kunt u een programma ontvangen, verstuurd door welke computer ook op het net.

Maar als u een programma wilt ontvangen dat verstuurd is door een specifieke computer, moet u LOAD "NET-n:" tikken. De 'n' staat voor het hetnummer van de andere computer. Bij voorbeeld:

```
LOAD "NET-17:"
```

Deze instructie maakt het mogelijk een programma van de computer met netnummer 17 te laden.

Uiteraard moeten aan de andere computer instructies gegeven worden, zodat het programma verzonden wordt. Computer nummer 17 zou de volgende instructie moeten krijgen:

```
SAVE "NET-5:PROG1"
```

Hierdoor zou het lopende programma op computer 17 verzonden worden naar computer 5 en het programma de naam "PROG1" krijgen.

UITZENDEN

Netnummer Ø is een speciaal geval. Dit nummer kan aan geen enkele computer gegeven worden, maar wordt gebruikt voor algemene net-operaties.

Netnummer Ø wordt gebruikt voor algemene uitzendingen. Het gaat hier om boodschappen die niet aan een specifieke computer zijn gericht, maar uitgezonden worden (evenals een radiosignaal) naar elke computer die ingeschakeld is en luistert.

Deze faciliteit komt erg van pas, wanneer u een korte boodschap voor alle andere computers heeft of wanneer u niet weet welke netnummer andere computers hebben.

Problematisch is dat deze verzendingen een niet erg betrouwbare manier van communiceren zijn. In geval van een gericht signaal dat verstuurd wordt naar een specifieke computer, wordt de boodschap steeds opnieuw verzonden totdat de ontvangende computer bevestigt, dat de boodschap veilig en wel ontvangen is. Met algemene uitzendingen is dat niet mogelijk. Ook is het niet mogelijk de zendsnelheid automatisch terug te brengen tot een niveau, waarop de ontvangende computer met de informatie kan werken.

De instructie

```
SAVE "NET-Ø:PROG1"
```

brengt dus teweeg, dat een programma naar alle computers verstuurd wordt, maar er bestaat een grote kans dat het programma niet correct ontvangen wordt. Evenmin weet de computer die verstuurt, of de overbrenging al of niet is gelukt.

OPEN VOOR ALLE BOODSCHAPPEN

Netnummer Ø kan ook gebruikt worden voor selekte ontvangst van boodschappen via het net. Wanneer een computerkanaal (zie het hoofdstuk "Kanalen" en het OPEN commando in het Naslaggedeelte) opengesteld wordt voor "NET-Ø:", wordt het een 'algemeen' kanaal voor net-operaties.

Boodschappen die door welke computer ook verstuurd worden via het net, worden ontvangen als een algemeen kanaal open is. Ook een boodschap die specifiek aan uw computer is gericht, wordt deze boodschap door een speciaal kanaal te openen voor communicatie met de andere machine.

Dit betekent niet dat u prive boodschappen gaat ontvangen van andere machines die op het net aangesloten zijn. Een gerichte boodschap van de ene computer aan een andere kan niet door een derde computer afgeluisterd worden.

Let wel: wanneer u een speciaal kanaal hebt opengesteld voor een andere machine, ontvangt u boodschappen via dit kanaal (niet via het algemene kanaal), zelfs de algemene boodschappen.

COMMUNICATIE-KANALEN

Voor optimaal gebruik wordt een kanaal specifiek voor communicatie met een machine geopend. Hiervoor kunt u gebruik maken van de gewone procedures voor het openen van kanalen.

```
OPEN £11Ø:"NET-17:"
```

Met deze instructie wordt kanaal 11Ø geopend voor wederzijdse communicatie met computer 17.

Daarna kan dit kanaal gebruikt worden in combinatie met gewone invoer/uitvoer instructies. Bij voorbeeld:

```
PRINT £11Ø: "Dit is een boodschap voor computer 17"
LINE INPUT 11Ø:A$ ! A$ zal de regel van computer 17
ontvangen.
```

Omdat boodschappen die van de ene naar de andere computer verstuurd worden in een buffergeheugen opgeslagen worden (in het geheugen vastgehouden voor versturing of na ontvangst), is het vaak nodig een tweetal speciale instructies te gebruiken. FLUSH £kan forceert de verzending van elk gegeven dat in een buffergeheugen op verzending wacht. Normaliter worden boodschappen niet verstuurd zolang ze nog niet 256 tekens lang zijn, of wanneer het kanaal gesloten is; het FLUSH commando moet met andere woorden telkens dan gebruikt worden wanneer een kort bericht meteen verstuurd moet worden.

CLEAR £kan:NET stelt de invoer- en de uitvoerbuffers op nul. De computer ontvangt pas dan op een bepaald kanaal een boodschap van een andere computer wanneer het ontvangende buffergeheugen op nul staat. Zo wordt verminking van gegevens voorkomen, nog voordat ze gebruikt kunnen worden door de ontvangende computer.

Wanneer gegevens nog niet zijn verwijderd uit de ontvangende buffer (door een INPUT instructie bij voorbeeld), maar wel zonder meer verwijderd kunnen worden, moet u de CLEAR instructie gebruiken. In geval van gegevens die niet zijn verzonden moet voor CLEAR gebruik gemaakt worden van FLUSH.

```
PRINT £11Ø:"Bericht voor computer 17"
FLUSH £11Ø
CLEAR £11Ø:NET
LINE INPUT £11Ø:A$
```

Bij de keuze van kanaalnummers voor net-operaties is het raadzaam een kanaal te gebruiken waarvan het nummer boven de 1ØØ ligt (maar te voorkomen dat u een systeemgekozen kanaal kiest; zie OPEN, p. 193). Dit houdt hiermee verband dat he BASIC alle kanalen van 1-99 sluit, zodra variabelen op nul gesteld worden (wat nogal vaak verwacht mag worden in de rechstreekse modus).

NETCOMMUNICATIE ALS ACHTERGRONDTAAK

De meest interessante manier om van het net gebruik te maken doet zich voor, wanneer een programma zo is geschreven dat de communicatie via het net succesvol kan verlopen als een achtergrondtaak tijdens de hoofdbewerking van een programma.

Het besturingssysteem van de Enterprise maakt gebruik van een methode, bekend als 'onderbreken', om het net te besturen. Dit betekent dat computers in hun eigen tijd met elkaar kunnen communiceren, op een manier die voor de gebruikers van de computers niet waarneembaar is. het BASIC voorziet in een methode om met dit soort bewerkingen om te kunnen gaan, en wel via functies voor de afhandeling van uitzonderingen.

Uitzonderingen zijn fouten of andere dingen die gebeuren kunnen, die het normale verloop van een programma onderbreken (zie hoofdstuk 'Hoe om te

gaan met uitzonderingen').

Wanneer een BASIC programma loopt en onderbrekingen vanaf het net mogelijk zijn gemaakt met de instructie SET INTERRUPT NET ON, zal de ontvangst van een bericht op een netkanaal een uitzondering veroorzaken.

Een functie voor de afhandeling van uitzonderingen die er speciaal voor is om net-operaties af te handelen, moet altijd als eerste regel in het blok de instructie SET INTERRUPT NET OFF geven. Hiermee wordt voorkomen dat de computer in verwarring raakt, doordat een nieuwe onderbreking ontvangen wordt nog tijdens de afhandeling van de eerste onderbreking. Tijdens de afhandeling van de uitzondering worden de invoerbuffers van het net 'opgevraagd' (op hun beurt gecontroleerd) met behulp van de ASK NET CHANNEL instructie. Per keer kunnen verschillende kanalen berichten bevatten (gedurende de afhandeling van de uitzondering kunnen wel nog moet herhaald worden totdat de waarde 255 geretourneerd wordt) wat erop wijst dat geen berichten meer op ontvangst wachten. SET INTERRUPT NET ON moet de laatste instructie zijn van het blok.

Merk op dat functies voor de afhandeling van uitzonderingen in werking gesteld worden door 'onderbrekingen van programmatuur'. Deze komen niet voor in de rechstreekse modus van het BASIC; daarom moet er een voorgrond programma lopen om voor NET OPERATIES gebruik te kunnen maken van de techniek voor de afhandeling van uitzonderingen.

```
PRINT  12 "Gegroet, computer nummer 15"
```

Deze regel bewerkstelligt dat de boodschap naar de andere computer gestuurd wordt. Natuurlijk kan computer 15 besluiten deze boodschap te negeren, tenzij zulke commando's zijn gegeven als

```
INPUT  12:  1$
PRINT  A$
```

Deze instructies brengen teweeg dat de computer een string ontvangt die in A$ gestopt wordt (vooropgesteld dat kanaal 12 correct is geopend) en vervolgens afgedrukt.

Het 'brein' van de Enterprise is een Z80 microprocessor. De Z80 kan ongeveer 500 specifieke bewerkingen uitvoeren, waarvan elk een eigen codenummer heeft — een 'machinecode-nummer'. Wanneer u een verwerkingseenheid in machinecode programmeert, adresseert u deze eenheid direct en in haar eigen taal (niet via de BASIC-vertolker).

Er zijn twee belangrijke redenen die voor u wel eens aanleiding zouden kunnen zijn om machinecode routines in uw BASIC programma's op te nemen: enerzijds de grotere snelheid (die vooral van pas kan komen in geval van grafische gegevensverwerking en bij klank en geluid) en anderzijds de mogelijkheid gebruik te maken van een karakteristiek van de apparatuur waarin BASIC niet voorziet.

Progammeren in machinecode is een omvangrijk onderwerp dat in een enkel hoofdstuk slechts ontoereikend behandeld kan worden. Voor geinteresseerden zijn er vele programmeer-handleidingen voor de Z80 beschikbaar.

Het BASIC van de Enterprise kent verschillende commando's die u de mogelijkheid bieden machinecode routines (in hexadecimale codes) samen te stellen en deze uit te voeren vanuit een BASIC programma — ofschoon die commando's geen deel uitmaken van de ANSI standaard.

ALLOCATE

Eerst moet u een hoeveelheid geheugen reserveren voor de opslag van uw code; bepaal hoeveel bytes uw code zal omvatten en gebruik dan dit commando:

ALLOCATE aantal-bytes

Maar bedenk wel dat ALLOCATE (toewijzen) alle opgeslagen variabele waarden vernietigt; u doet er dus het verstandigst aan dit commando alleen aan het begin van een programma te gebruiken.

CODE EN HEX$

De routine in machinecode wordt — met benaming — opgeslagen door gebruik te maken van het CODE commando:

CODE naam = routine in hexadecimale code

De naam heeft dezelfde vorm als de benaming van een

variabele.

De CODE instructie kan alleen een string opslaan — uw hexadecimale codes moeten in de juiste vorm omgezet worden, en wel door gebruik te maken van HEX$. Voorbeeld:

HEX$ ("hex,hex,...")
(Vergeet niet de aanhalingstekens of de komma's waardoor de hex waarden van elkaar gescheiden worden!)

Een andere mogelijkheid is:

HEX$ (elke denkbare string)

Een routine bij voorbeeld die een nader aangegeven getal verdubbelt —

```
TEST:   29 ADD HL, HL    ! Hetzelfde getal bij een
                           getal optellen.
        C9 RET
```

— kan als volgt in een BASIC programma ingevoegd worden:

```
100     ALLOCATE 2
110     CODE TEST=HEX$("29,C9")
```

Zodra dit gedeelte van het programma uitgevoerd is, wordt de routine opgeslagen in de gereserveerde hoeveelheid geheugen.

DE ROUTINE UITVOEREN

In het algemeen gesproken creeërt uw code een nieuwe 'functie'. In het hoofdstuk 'Functies definieren' hebben we twee soorten functies onderscheiden. De 'ingebouwde' functies behoren tot de categorie voor het verkrijgen van uitkomsten. Een goed voorbeeld hiervan is SIN — dit commando berekent de sinus van een speciale hoek wanneer u iets dergelijks tikt als hieronder:

```
PRINT SIN (53)
```

De waarde die verwerkt moet worden (wordt tussen haakjes geplaatst) wordt het argument van de functie

genoemd. Sommige ingebouwde functies maken geen gebruik van een argument — bij voorbeeld RND — maar alle retourneren een waarde, een 'antwoord'.

Het andere type functie is meer te vergelijken met een 'commando'. Een commando bestaat uit een reeks bewerkingen die iets uitvoeren, zoals bij voorbeeld het scherm op blank stellen of wellicht een modus voor grafische gegevensverwerking instellen. Een commando retourneert geen waarde.

De manier waarop uw routine uitgevoerd wordt zal afhangen van de functiecategorie waartoe de routine behoort. Als er een waarde geretourneerd wordt, zoals in ons verdubbel-het-getal voorbeeld, moet u het volgende gebruiken:

USR (naam,argument)

Voorbeeld:

```
PRINT USR (TEST,2)
```

Hierdoor zal 4 op het scherm afgedrukt worden; en

```
LET A = 3*USR (TEST,2)
```

zal de waarde van 12 toewijzen aan de variabele A.

Onthoud dat het argument van USR aan het begin van de routine naar HL geschreven wordt; en de waarde die door USR geretourneerd wordt, is de inhoud van het HL register na afloop van de routine.

Als uw routine een bewerking is van het commando-type, moet u gebruik maken van

CALL USR

— door dit commando wordt geen waarde geretourneerd.
Voorbeeld:

```
CALL USR(CLEAR,Ø)+USR(GRAPH,Ø)+USR(PICTURE,Ø)
```

WORD$

Het commando WORD$ zet zijn argument om in een twee-bytes string — LSB MSB; dit commando komt dus van pas bij het regelen van sprongen achteruit vanaf

labels. Bijvoorbeeld,

```
WORD$(TEST)
```

Deze regel zal het begin adres van de TEST routine in de juiste rangschikking retourneren voor machinecode sprongen en calls.

Het naslaggedeelte biedt een overzicht van alle BASIC woorden die op de Enterprise voorhanden zijn, inclusief gebruik en toepassingsmethode. Sommige hiervan zijn slechts summier aan de orde gekomen, andere zijn helemaal nog niet genoemd.

Hopelijk gaat u met al deze woorden experimenteren en zelf ontdekken wat het volledige scala van mogelijkheden is op de Enterprise. Als u al vertrouwd bent met BASIC zult u dit gedeelte algemeen gesproken het beste richtsnoer vinden voor het Intelligent Standard BASIC (copyright Intelligent Software Ltd., 1984) dat op de Enterprise beschikbaar is.

ALGEMENE REGELS

Hoofdletters en kleine letters zijn onderling verwisselbaar in BASIC sleutelwoorden en identificatiesymbolen; FOR, For, for en fOr zijn alle hetzelfde woord.

Een programmaregel kan wel 25Ø tekens bevatten, de regelnummering loopt van 1 tot 9999. Een regel kan verschillende opdrachten bevatten die van elkaar gescheiden zijn door dubbele punten; alles dat na THEN is toegestaan (in geval van een IF/THEN opdracht) kan op een regel ondergebracht worden die uit meerdere opdrachten bestaat.

Een identificatiesymbool kan maximaal 31 tekens lang zijn en alle tekens zijn significant. Het identificatiesymbool kan letters, cijfers, punten en onderstrepingstekens bevatten; het eerste teken moet een letter zijn.

! wordt gebruikt om de rest van de regel voor opmerkingen te reserveren.

De vertolker schrapt de spaties die voor en achter een regelnummer en het eerste sleutelwoord staan, en de spaties aan het eind van de regel. Daarna laat hij het programma inspringen bij elk nieuw blok. FOR, DEF, DO, HANDLER SELECT en WHEN laten de volgende regel 2 spaties inspringen. ELSE en CASE binnen een ingesprongen blok worden twee tekens naar rechts verschoven. LOOP, END en NEXT vormen het eindpunt van een inspringing. De regelnummers worden zo afgedrukt dat de rechter kantlijn van de kolom met nummers keuring gelijkmatig loopt.

```
1    LET A = Ø
1Ø   DO WHILE A < 1Ø
```

```
100     LET A = A + 1
110     SELECT CASE A
120     CASE 1
130       PRINT "de eerste keer"
140     CASE ELSE
150       PRINT "niet de eerste keer"
160     END SELECT
170     PRINT A
180   LOOP
190   GOTO 1
1000  END
```

Zie voor meer gegevens over de syntaxis en de regels van het BASIC het Draft Proposal for Standard BASIC van de ANSI commissie X3J2/82-17.

De sleutelwoorden zijn in **VETTE HOOFDLETTERS** gedrukt in de linker kantlijn van de pagina. Het format van de commando's die van het sleutelwoord gebruik maken, is in gewoon schrift gedrukt, voorbeelden zijn cursief gedrukt.

MEERDERE PROGRAMMA'S

HET IS-BASIC op de Enterprise biedt de mogelijkheid dat er per keer veschillende programma's in de computer zitten. Elk programma heeft zijn eigen regelnummers en zijn eigen variabelen.

U kunt naar een programma verwijzen ofwel met een nummer ofwel met een naam die vermeld wordt op een PROGRAM regel. Zie met name de commando's **CHAIN, EDIT** en **PROGRAM**.

Per keer is een van de programma's (het systeemgekozen programma is programma Ø) het 'actuele' programma, waarop commando's als LIST en RENUMBER van toepassing zijn. Het nummer van het 'actuele' programma is te zien op de 'statusregel' boven in het beeld.

Programma Ø kan bij benadering gebruik maken van 42K aan geheugen. De overige programma's zijn beperkt tot ieder 32K.

UITBREIDINGEN

De mogelijkheid bestaat het BASIC uit te breiden. Deze uitbreidingen kunnen ofwel vanaf cassette of diskette geladen worden ofwel inbegrepen zijn in een aanvullende stapel eenheid voor de computer. Tekst en uitleg bij de extra commando's of functies wordt gegeven in de gebruiksaanwijzing die bij dergelijke

produkten zit.

DATA TYPEN

Twee data typen worden onderscheiden: numerieke en string variabelen. De namen van numerieke variabelen worden gegeven volgens de regels voor identificatie-symbolen (zie Algemene Regels). Identificaties-symbolen voor string variabelen eindigen altijd met een $ teken.

Numerieke variabelen worden in binair gecodeerde decimalen berekend en in maximaal 10 cijfers weergegeven. Getallen worden verwerkt tussen $1e^{-64}$ en $9.999999999e^{62}$.

Strings hebben een maximale lengte van 254 tekens, mits aangegeven (zie **STRING**). Naar substrings kan verwezen worden in de vorm string-id (x:y), waarmee een string wordt aangeduid die begint met tekennummer x en eindigt met tekennummer y. Worden x of y weggelaten, dan wordt als standaardwaarde respectievelijk het begin of het eind van de string genomen.

BEWERKINGS-TEKENS

Rekenkundige bewerkingstekens:

*	—	vermenigvuldigen
/	—	delen
^	—	machtsverheffen
+	—	optellen
-	—	aftrekken

String bewerkingstekens:

&	—	aaneenschakelen

Relationele berwerkingstekens:

>	—	groter dan
<	—	kleiner dan
=	—	is gelijk aan
>=	—	groter of gelijk aan
<=	—	*kleiner of gelijk aan*
<>	—	niet gelijk aan
AND	—	logisch AND (waar/nietwaar)
OR	—	logisch OR (waar/nietwaar)
BAND	—	binair logisch AND
BOR	—	binair logisch OR

AFKORTINGEN

In dit gedeelte worden de volgende afkortingen gebruikt:

kan — kanaalnummer
id — identificatiesymbool (b.v. de naam van een variabele)
str — string
var — variabele
uitdr — uitdrukking
verbe — vergelijkings bewerkings tekens (b.v. >,>=, etc.)
para — parameter

regelnummer

regelnummer tekst
regelnummer spatie
regelnummer

Toevoegen of vervangen van een programmaregel. Wordt het regelnummer slechts door een spatie gevolgd, dan wordt een regel met een '!' ingevoerd. Staat er niets achter het regelnummer, dan wordt de regel geschrapt. Alleen uitgevoerd in de rechstreekse modus. Stelt variabelen op nul.

Let wel: alle commando's of opdrachten die variabelen op nul stellen, bewerkstelligen eveneens dat alle open kanalen in het bereik van 1 tot en met 99 gesloten worden (zie **OPEN**).

ALLOCATE

ALLOCATE uitdr toewijzen

Toegepast in combinatie met machinecode subroutines. Schuift de broncode van het programma op om een bloktussenruimte te creeren ter grootte van het gespecificeerde aantal bytes. Hier gaat de machinecode van de gebruiker naar toe. De adresteller wordt ingesteld op de eerste vrije byte in de tussenruimte. Onthoud dat hierdoor alle variabelen vernietigd worden; gebruik ALLOCATE dus alleen aan het begin van het programma.

ASK

ASK machine-optie var vragen

Informeert naar een bepaalde optie (bij voorbeeld de aanslagsnelheid); zie de paragrafen 'Machine-opties', 'Video-opties' en 'Geluids-opties'. Zie ook **SET** en **TOGGLE**.

De variabele, indien aanwezig, zal de actuele waarde van de machine-optie overnemen.

Voorbeeld: *ASK KEY RATE A*

Hierdoor wordt de actuele herhalingssnelheid van het toetsenbord aan de variabele A toegewezen.

auto (-matisch)

Speciaal opmaakcommando dat automatisch de regelnummers afdrukt. Functioneert alleen in de

rechtstreekse modus.

AUTO
AUTO AT 1Ø0 STEP 1Ø
AUTO STEP 1Ø0

Het systeemgekozen eerste regelnummer is 1Ø0. De systeemgekozen STEP is 1Ø. Oude regels worden vervangen door nieuwe regels met hetzelfde nummer.

AUTO kan herroepen worden door de 'stop' toets aan te slaan.

CALL aanroepen

CALL functie
CALL functie (para-lijst)

Wordt gebruikt om een functie aan te roepen (ofwel ingebouwd, ofwel gedefinieerd door DEF), wanneer de functie geen resultaat hoeft op te leveren.

Elke uitdrukking die op CALL volgt wordt berekend, maar de uitkomst wordt genegeerd. *CALL USR(A,B)+USR(C,D)* roept daarom twee USR programma's in machinecode aan.

Kan uitgevoerd worden in de rechstreekse modus.

CAPTURE CAPTURE FROM £kan TO £kan vangen

Houdt invoer vanaf eerste kanaal vast en vervangt deze door invoer die vanaf het tweede kanaal verwacht wordt, totdat een einde-bestand situatie op het eerste kanaal ontstaat of een fout optreedt.

CAPTURE FROM een specifiek kanaal kan ook beeindigd worden door £255 (normaliter ongeldig) als het TO kanaal te geven, en wel in een latere opdracht.

CASE Zie **SELECT** blok.

CAUSE EXCEPTION CAUSE EXCEPTION uitdr

Definieert een fout en brengt deze onder in de door de uitdrukking aangeduide categorie; de waarden van de gebruiker moeten tussen 1 en 999 liggen; deze waarden worden immers nooit door het BASIC gebruikt.

CHAIN Koppelen

CHAIN kan :programma nummer
CHAIN kan: "naam"

Gebruikt on BASIC programma's uit te voeren van uit het lopende programma, alsook voor het aanroepen van Read Only geheugens in de machine.
Paramaters kunnen wat hun waarde betreft van het ene aan het andere programma doorgegeven worden.

CHAIN "MIJN_programma" (1, "Fred")

Zie ook **PROGRAM**.

CLEAR terugstellen, op nul stellen, op blank stellen, schoonmaken

CLEAR £kan
CLEAR ENVELOPE
CLEAR FKEYS
CLEAR FONT
CLEAR £kan: NET
CLEAR GRAPHICS
CLEAR QUEUE nummer toon generator
CLEAR SCREEN
CLEAR SOUND
CLEAR TEXT

Stelt verschillende opties op blank. Kan uitgevoerd worden in de rechstreekse modus.

CLOSE (af-) sluiten

CLOSE £kan

Sluit het kanaal en bevriest elke denkbare buffer.

CODE CODE naam variabele=string

Wordt gebruikt in combinatie met machinecode subroutines. Schrijft een kopie van een string naar de positie die aangegeven wordt door de actuele adresteller. Is er een variabele gegeven, dan neemt deze de waarde aan van de adresteller. De adresteller

wordt zo achtergelaten dat deze naar de byte wijst die op de rij volgt; deze byte dient de machinecode te bevatten. De naam van de variabele kan later gebruikt worden om de routine aan te roepen of het bestemmingsadres van sprongen etc. te bepalen.

CONTINUE continueren, verder gaan

Als commando in de rechstreekse modus wordt hierdoor het programma opnieuw gestart, en wel op de volgende regel na een 'stop' commando of een aanslag van de 'stop' toets.

Bij toepassing in een programma als uitweg uit een functie voor de afhandeling van uitzonderingen wordt met de opdracht verder gegaan, die direct volgt op de regel waar de uitzondering veroorzaakt werd.

COPY kopieren

COPY FROM £kan: TO £kan:

Kopieert de inhoud van het ene kanaal naar een tweede kanaal (beide kanalen moeten open zijn). De kopie houdt op te bestaan i.g.v. een einde bestand, een fout of de 'stop' toets. De systeemgekozen invoer is kanaal Ø, de systeemgekozen uitvoer is kanaal 1Ø4.

COPY

Kopieert van £Ø — naar £1Ø4

COPY FROM £5

Kopieert van £5 naar £1Ø4. Kanaal 5 dient geopend te zijn.

DATA gegevens

DATA lijst met gegevens.

Maakt de implicatie mogelijk van een lijst met constanten, cijfers en/of strings die achtereen-volgens gelezen worden. Zie **READ**.

DATE DATE datum-string

Specificeert de datum die de computer bijhoudt. De datum wordt in het internationale standaardformaat YYYYMMDD aangegeven.

DATE "1985Ø727"

is gelijk aan 27 juli 1985.

Kan gebruikt worden in de rechstreekse modus. Zie **DATE$** functie.

DEF definitie

DEF numeriek-id=uitdrukking
DEF numeriek-id (parameterlijst)=uitdrukking
DEF string-id=tekstuele uitdrukking
DEF string-id (parameterlijst)=tekstuele uitdrukking

De definitie van een functie die uit een regel bestaat:

DEF AVERAGE (X,Y)=(X+Y)/2

DEF blok: een blok is een groep opdrachten die ofwel als procedure-opdracht aangeroepen kunnen worden ofwel als functie die in een uitdrukking een waarde retourneert. Er zijn een paar kleine afwijkingen van de ANSI definitie. Zie ook **CALL, EXIT DEF, NUMERIC** en **STRING**.

def-regel
    een willekeurig aantal opdrachten of blokken
einde-def-regel

def-regel:
DEF numeriek-id
DEF numeriek-id (parameterlijst)
DEF string-id
DEF string-id (parameterlijst)

einde-def-regel:
END DEF

Als de functie een waarde hoort terug te geven, dan moet deze waarde met het DEF blok aan de functienaam worden toegeschreven.

```
DEF ANSWER(A$)
  IF UCASE$(A$(1:1))="Y" THEN
  ANSWER R=1
  ELSE IF UCASE$(A$(1:1))="N" THEN
    ANSWER=Ø
  ELSE
    ANSWER=1
  END IF
END DEF
```

Het aantal variabelen in welk gedeelte van het programma ook is dynamisch — het aantal is met andere woorden afhankelijk van de loop der regels die al zijn uitgevoerd en niet van de statistische opmaak van het programma.

```
1ØØ     NUMERIC FRED
11Ø     LET FRED = 1
12Ø     CALL Q
13Ø     PRINT FRED
14Ø     END
2ØØ     DEF P
21Ø       LET FRED = 123! Deze FRED is een algemene
          variabele.
22Ø     END DEF
3ØØ     DEF Q
31Ø       NUMERIC FRED! Een bijzondere FRED.
32Ø       LET FRED = Ø
33Ø     CALL P
34Ø     PRINT FRED
35Ø     END DEF
```

In dit voorbeeld wordt FRED als algemene en als bijzondere variabele gebruikt. Wanneer regel 21Ø uitgevoerd wordt, verandert de FRED van regel 31Ø in 123 en niet die van regel 1ØØ. Het programma zal 123 en 1 afdrukken. In een taal met een statische bereik zou het programma 1 en 123 drukken; dit kan ook gebeuren als hetzelfde programma gedraaid wordt bij toepassing van een BASIC vertaalprogramma.

Alles dat binnen een DEF blok benoemd wordt is specifiek ('bijzonder') voor dat blok en wordt toegewezen bij elke eerste uitvoering van de benoeming na de CALL. Wat niet benoemd wordt kan afhankelijk van het verloop bijzonder of algemeen zijn.

U doet er het verstandigst aan alle variabelen aan het begin van een programma of een functie te benoemen; zo voorkomt u onverwachte resultaten.

```
100     CALL P ! Deze CALL van P heeft I als
        bijzondere variabele van P.
110     LET I = 9
120     CALL P ! Deze CALL van P verandert de
        algement I.
130     END
140     DEF P
210       LET I = 6
220     END DEF
```

Teneinde kloppende resultaten te krijgen zou een regel als

```
90      NUMERIC I
```

aan het programma toegevoegd moeten worden; hierdoor wordt I — beide keren dat P aangeroepen wordt — algemeen.

De hoeveelheid geheugen die voor de opslag van bijzondere variabelen wordt gebruikt, wordt vrijgegeven zodra de functie verlaten wordt. Als deze karakteristiek uitgebuit wordt kan er op efficiente wijze gebruik gemaakt worden van het computergeheugen — zo kan bij voorbeeld een tijdelijke reeks met gegevens binnen een functie gesitueerd worden.

Vrijwel alles kan verplaatst worden als referentie parameter. Doorgaans worden parameters wat hun waarde betreft doorgegeven, wat wil zeggen dat een kopie aan de functie doorgegeven wordt, en dat elke denkbare bewerking binnen die functie de externe variabelen niet zal veranderen.

Referentie parameters ontlenen hun vorm aan de actuele parameter; elke verandering binnen de functie zal ook een verandering van de externe variabelen met zich meebrengen.

```
100     DEF SWAP (REF A,REF B)
110       NUMERIC T
120       LET T = A
130       LET A = B
140       LET B = T
```

```
15Ø     END DEF
2ØØ     LET X = 99
21Ø     LET Y = 23
22Ø     CALL SWAP(X,Y)
23Ø     PRINT X,Y
```

Afgedrukt worden 23 en 99

Reeksen en functies moeten altijd via referentie doorgegeven worden.

```
1ØØ     NUMERIC A(1Ø)
11Ø     OPTION ANGLE DEGREES
12Ø     DEF P (REF FN,X)
13Ø       PRINT FN(X),
14Ø     END DEF
15Ø     LET A(2) = 66
16Ø     CALL P(A,2)
17Ø     CALL P(SIN,3Ø)
```

Afgedrukt worden 66 en .5

Het doorgeven van ingebouwde en door de gebruiker gedefinieerde functies kan goed van pas komen bij bibliotheek-programmatuur. Een functie voor het tekenen van grafieken kan datgene dat getekend moet worden als parameter doorgeven, een sorteerfunctie kan dit voor haar rekening nemen en de routines vergelijken die als functie doorgegeven kan worden.

Functies kunnen zichzelf bij herhaling aanroepen.

DELETE schrappen, wissen

DELETE beschrijving regel TO beschrijving regel,...
DELETE beschrijving regel — beschrijving regel,...
DELETE naam blok.

Wist programmaregels. Wordt alleen uitgevoerd in de recht streekse modus. Stelt variabelen op nul.

```
DELETE LAST
DELETE FIRST TO 1ØØ
DELETE 1 TO 199, 3ØØ, 5ØØ TO 9999
```

In plaats van TO wordt ook '–' geaccepteerd. Als het

eerste (of laatste) getal uit een reeks weggelaten wordt, gaat het systeem automatisch naar de eerste (of laatste) regel van het programma.

Voorbeeld: : *DELETE FIRST-1ØØ, 5ØØ-LAST*
of : *DELETE TO 1ØØ, 5ØØ-*
in plaats van: *DELETE FIRST TO 1ØØ, 5ØØ TO LAST*

Regels die een functie P definiëren kunnen met DELETE P worden gewist. DELETE alleen wist alle programmaregels; is te onderbreken met de 'stop' toets.

DIM DIM lijst met reeksen

Benoemt numerieke of tekstuele reeksen; systeem-gekozen waarden aflopend tot Ø, indien niet gespecificeerd. Een of twee dimensies zijn mogelijk. De maximale lente van een string kan niet bepaald worden met DIM; dus wordt de systeemgekozen waarde van 132 tekens gebruikt. (Vergelijk **STRING**).

DIM A(1 TO 1Ø), FRED$(9), B(-7899 TO 7890)

Merk op: alle reeksen hierboven bestaan uit 1Ø elementen.

DISPLAY (visueel) weergeven

DISPLAY £kan:AT a FROM b TO c

Definieert een venster voor de weergave van een segment van een tekstuele of grafische video-pagina. Scherm-rij 'a' is de positie waar de eerste regel van het segment zal komen. De beide parameters 'b' en 'c' zijn tekenrijen op de pagina die weergegeven moet worden; 'b' en 'c' geven de eerste en de laatste regel van het segment aan. De nummering van de tekenrijen stemt overeen met de conventies die voor teksten gelden — het maakt geen verschil uit of de pagina die weergegeven wordt een tekstuele of een grafische pagina is. Zie **PRINT**.

DISPLAY GRAPHICS

Stelt de eerste 20 regels beschikbaar voor grafische gegevensverwerking en brengt de laatste grafische pagina — indien open — in beeld.

DISPLAY TEXT

Stelt het volledige scherm beschikbaar in de tekstmodus en brengt een hele tekstpagina in beeld — vooropgesteld dat die pagina voorheen al open was. Stelt het grafische scherm niet op blank.

Als voorheen slechts een kleine pagina voor tekst open was, wordt deze op nul gesteld en wordt er een nieuwe pagina van volle omvang geopend.

DO

do-regel
    ieder willekeuring aantal opdrachten of blokken
loop-regel

do-regel:
DO
DO WHILE uitdrukking met vergelijkingen
DO UNTIL uitdrukking met vergelijkingen

loop-regel:
LOOP
LOOP WHILE uitdrukking met vergelijkingen
LOOP UNTIL uitdrukking met vergelijkingen

De structuur van een lus (loop) wordt gedefinieerd als een blok, bestaande uit een DO-regel, het lichaam van de lus en een LOOP-regel. DO of LOOP kan niet geplaatst worden op een voorwaardelijke regel.

```
DO WHILE A > 3 AND A < 10
  LET A = A + 1
  PRINT A
LOOP
```

De besturing kan niet vanuit het binnenste van een lus plaatsvinden. Zie **EXIT DO**.

EDIT

bewerken, opmaken

EDIT programma-nummer
EDIT "naam"

Bewerkstelligt dat het gespecificeerde programma actueel wordt, zodat LIST, RENUMBER, RUN etc. erop van toepassing kunnen zijn. Wordt alleen uitgevoerd in de rechtstreekse modus. Zie: **CHAIN, INFO** en **PROGRAM**.

ELSE Zie **IF**.

END Stopt de uitvoering, markeert het einde van een programma. In geval van **END DEF, END HANDLER, END IF, END SELECT** en **END WHEN** wordt het einde van het desbetreffende blok gemarkeerd.

ENVELOPE envelop

ENVELOPE £kan:NUMBER a;b,c,d,e;f,g,h,i;...;
RELEASE j,k,l,m;...

Definieert een muziekenvelop voor toepassing in combinatie met een SOUND opdracht die een en ander bestuurt. De 'a' die de envelop identificeert moet tussen Ø en 254 liggen.

De parameters 'b', 'c', 'd' en 'e' definiëren de eerste fase van de envelop; 'b' geeft de verandering in toonhoogte aan, uitgedrukt in halve tonen (decimalen toegestaan), 'c' en 'd' specificeren de verandering in geluidssterkte voor achtereenvolgens de linker en de rechter luidspreker; en 'e' geeft de duur aan van de fase, uitgedrukt in 'tellen' (een tel is 1/5Ø seconde).

De waarden voor 'c' en 'd' liggen binnen het bereik van Ø-63; hierdoor worden de veranderingen in geluidssterkte gespecificeerd, uitgedrukt in percentages van het totale maximumvolume dat door de SOUND opdracht is toegestaan. Een waarde van -63 zal het geluid uitschakelen (elke overschrijding van de hoogste of de laagste waarde wordt genegeerd); aan het begin van de envelop wordt het geluid verondersteld 'uit' te zijn. Als geen gebruik gemaakt wordt van stereo apparatuur, wordt het volume telkens bepaald door de waarden voor de linker en voor de rechter luidspreker (afhankelijk van SOUND en ENVELOPE opdrachten) bij elkaar op te tellen.

De volgende fase wordt gedefinieerd door 'f', 'g', 'h' en 'i' ... Zie voor het mogelijke aantal fasen de 'Geluidsopties', onder **SOUND BUFFER**.

RELEASE is facultatief; dit woord kan gevolgd worden door een onbepaald aantal fasen met hun afzonderlijke parameters. De 'release' fasen worden na uitvoering van de voorafgaande fasen of na afloop van de SOUND duur uitgevoerd, als er op hetzelfde kanaal geen muziekje op de wachtlijst staat.

EXIT DO
EXIT FOR
EXIT DEF

Een FOR, DO of DEF blok verlaten. Niet geldig, tenzij binnen het juiste blok gebruikt.

EXIT HANDLER

Het verlaten van een functie voor de afhandeling van uitzonderingen die de uitzondering aan de hem omringende omgeving doorgeeft. Hieroor wordt weer een andere functie voor de afhandeling van uitzonderingen geactiveerd:ofwel een door de gebruiker gedefinieerde functie ofwel de systeemgekozen functie.

EXT

EXT parameter-string

Geeft een string door aan het besturingssysteem. Wordt vervolgens doorgegeven aan geldige externe programma's in het geheugen (ROM of RAM). De string wordt dan door deze programma's naar behoren vertolkt.

Het eerste woord van een parameter-string geeft meestal een uit te voeren commando aan, of een nieuw programma waar naartoe moet worden gesprongen.

Bijvoorbeeld:

EXT"WP"

springt naar de ingebouwde wordprocessor van de computer.

Het woord "HELP" is van speciale betekenis omdat alle externe programma's er met hun naam op moeten antwoorden. Antwoorden van de externe programma's worden naar het standaard systeemkanaal gestuurd. (Zie **DEFAULT CHANNEL** machine-optie.)

Aanvullende toepassings- en service programma's bepalen hun eigen commando-namen en parameter instellingen. Deze programma's antwoorden vaak op de instructie

.

EXT"HELP NAME"

("NAME" is dan de eigenlijke naam van het programma) door een lijst te geven van de beschikbare string commando's.

Hetzelfde effect als met EXT kan in de rechtstreekse modus verkregen worden door een regel met een dobbele punt te beginnen. In dat geval hoeft de parameter string niet tussen aanhalinstekens te staan.

:HELP NAME

FLUSH

FLUSH £kan

Dwingt tot verzending van gegevens die nog in een kanaalbuffer zitten, zonder het kanaal af te sluiten of het eindebestand label te geven. Deze bewerking is slechts op bepaalde apparaten van toepassing (b.v. NET:). Kan in rechtstreekse modus worden gebruikt.

FOR

for-regel
elk willekeurig aantal opdrachten of blokken next-regel

for-regel:
FOR enkelvoudige variabele=uitdrukking TO uitdr
STEP uitdr

STEP kan weggelaten worden — de systeemgekozen STEP waarde is 1.

next-regel:
NEXT
NEXT variabele

De structuur van een FOR lus wordt gedefinieerd als een blok, bestaande uit een FOR regel, het lichaam van de lus en een NEXT regel. FOR en NEXT kunnen niet op een voorwaardelijke regel geplaatst worden. Toegestaan in Minimal BASIC.

```
FOR Y = Ø TO 1Ø STEP 2
  PRINT Y
NEXT Y
```

De eind-waarde van de 'stuur'-variabele na afloop van de lus omvat ook de STEP uitdrukking.

Geneste FOR lussen kunnen niet dezelfde 'stuur'-variabele gebruiken. Bij uitvoering van de FOR regel worden de begrenzings- en vermeerderings-uitdrukkingen gekopieerd en naar een verscholen geheugendeel geschreven; deze waarden kunnen niet veranderd worden door het lichaam van de lus. De besturing kan niet van buiten overgebracht worden naar het binnenste van een lus. Zie ook **EXIT FOR**.

GET

GET £kan: string-id

Haalt een enkel teken uit een kanaal en geeft een lege string (" "), wanneer er geen teken beschikbaar is.

Geeft standaard kanaal 1Ø5 (KEYBOARD:) en is voor eenvoudige toepassingen vergelijkbaar met de functie INKEY$.

GOSUB

GOSUB regelnummer

Roept subroutine aan die op de aangegeven regel begint.

GOTO

GOTO regelnummer

De uitvoering van het programma wordt voortgezet op de gespecificeerde regel. Kan toegepast worden om FOR, DO, HANDLER of DEF blokken, etc. te verlaten, maar dit wordt niet aangeraden.

GRAPHICS

grafische gegevensverwerking

GRAPHICS
GRAPHICS HIRES/LORES nummer aantal kleuren
GRAPHICS ATTRIBUTE

Het commando GRAPHICS komt neer op het sluiten en opnieuw openen van de systeemgekozen grafische en tekstuele pagina (£1Ø1 en £1Ø2); de systeemgekozen grafische pagina wordt weergegeven. Onder in het beeld blijven vier regels open vor tekst.

GRAPHICS stelt ook het systeemgekozen kanaal in (1Ø1) voor video machine-opties, zoals bij voorbeeld PALETTE.

Geldige nummers die het aantal kleuren aangeven zijn 2, 4, 16 en 256. Wanneer het aantal kleuren of de HIRES/LORES optie niet gespecificeerd wordt, worden de waarden die van toepassing waren bij het laatste GRAPHICS commando, opnieuw gebruikt. Zie voor de betekenis van deze waarden de 'Video-modus' onder 'video-opties'. In eerste instantie selecteert u met GRAPHICS een grafische pagina met een hoog scheidingsvermogen en met 4 kleuren.

Met GRAPHICS ATTRIBUTE selecteert u een 'attribute' modus van de grafische gegevensverwerking, waarin elke kleurcel (8 puntjes breed en 1 puntje diep) een 'verf'-kleur en een 'papier'-kleur kan bevatten. Deze modus combineert een palet van 16 kleuren met het scheidingsvermogen van GRAPHICS 4 (het scheidingsvermogen en het aantal kleuren kunnen niet door de gebruiker gspecificeerd worden). Er kunnen voor zowel afdrukken als teken commando's gegeven worden, ofschoon de kleuren elkaar wederzijds kunnen beinvloeden. De lijnmodi 4-7 (zie 'Video-opties') worden gebruikt om in de 'papier'-kleur in plaats van de 'verf'-kleur te tekenen.

HANDLER

funtie

HANDLER naam functie
  opdrachten voor de afhandeling van uitzondering
END HANDLER

Het HANDLER blok wordt gebruikt om uitzonderingen aan te pakken die veroorzaaakt zijn door fouten, het CAUSE EXCEPTION commando of onderbrekingen van de machine.

De funtie die gebruikt moet worden, wordt gespecificeerd door de naam die gegeven wordt in het WHEN blok dat uitgevoerd gaat worden.

Zie **CONTINUE, RETRY, EXIT HANDLER**, en de functies **EXLINE** en **EXTYPE**.

De besturing kan alleen als gevolge van een uitzondering (niet door GOTO of GOSUB) naar een functie voor de afhandeling van uitzonderingen overgebracht worden.

Als zich een uitzondering voordoet binnen deze functie, is de uitwerking hiervan hetzelfde als EXIT HANDLER: de besturing gaat nu immers naar het

volgende hogere niveau van de functie (zoals aangegeven door het volgende hogere niveau van het WHEN blok). Maar de waarden van EXTYPE en EXLINE zullen nu vervangen worden door nieuwe waarden.

IF als, indien

IF vergelijkende uitdrukking THEN regelnummer
IF vergelijkende uitdrukking THEN enkelvoudige opdracht.

Opdrachten die niet op een IF regel zijn toegestaan zijn DATA, DEF, END, DIM, MUMERIC, STRING, nog een IF, of welke opdracht ook die een blok introduceert.

IF A > = 3 AND A < = 9 THEN 1ØØ
IF A > = 3 THEN GOTO 1ØØ

if-regel
  een onbepaald aantal opdrachten of blokken
else-if-regels
  een onbepaald aantal opdrachten of blokken
else-regel
  een onbepaald aantal opdrachten of blokken
end-if-regel

if-regel:
IF vergelijkende uitdrukking THEN

else-if-regel:
ELSE IF vergelijkende uitdrukking THEN

Het aantal ELSE IF regels is onbeperkt.

else-regel:
ELSE

end-if-regel
END IF

IF blokken kunnen elke opdracht bevatten die niet beperkt is tot de rechstreekse modus.

IF A < 1Ø THEN

```
  PRINT A
ELSE IF A > 3Ø AND A < = 4Ø OR A > 5Ø THEN
  PRINT A + 1ØØ
ELSE
  PRINT B
END IF
```

De ELSE en de ELSE IF regels kunnen gebruikt worden om het blok in sub-blokken te verdelen met de gebruikelijke betekenissen.

ELSE mag slechts een keer gebruikt worden, ELSE IF zo vaak als nodig is. De programma-besturing kan niet van buiten een IF blok naar binnen overgebracht worden.

IMAGE

opmaakkenmerk

IMAGE: opmaakspecificatie

Toegepast in combinatie met PRINT commando's om de opmaak van de uitvoer te regelen. De opmaak-specificatie is een aantal tekens met — in deze context — de volgende betekenis.

Numerieke opmaaktekens:

, — drukt een komma af in het getal;
$ — drukt een drijvend dollarteken af dat aan het teken voorafgaat;
- — drukt een zwevende spatie of een '-' teken af;
+ — drukt een zwevend '+' of '-' teken af;
% — drukt een cijfer af, inclusief de belang-rijkste nullen.
* — drukt een cijfer af of een voorafgaand '*';
£ — drukt een cijfer of spatie af, en de nullen achter een drijvende komma;
. — drukt een drijvende punt af;
^ — drukt exponent af; minimaal vier tekens.

Als het getal niet in de spatie past, wordt er een fout veroorzakt.

Tekstuele opmaaktekens:

< — de string links uitvullen, in het gegevensveld aangegeven door '£' tekens;
£ — drukt een teken af;
> — de string rechts uitvullen.

Het 'uitvul'-teken moet het begin van het gegevensveld vormen; wordt dit teken niet gebruikt, dan wordt de string gecentreerd.

De opmaakspecificatie in de IMAGE regel begint direct na de ':' en eindigt met het laatst afgedrukte teken op deze regel.

INFO

Drukt de hoeveelheid geheugen in het systeem af en het aantal ongebruikte bytes. Ook wordt er een tabel met informatie over de programma's in het geheugen afgedrukt, en wel als volgt:

Nummer programma	aantal bytes in programma	eerste regel van programma

INFO stelt alleen variabelen op nul. Alleen uitgevoerd in de rechstreekse modus.

INPUT

invoeren

INPUT £kan, IF MISSING actie, AT rij-nummer, kolom nummer PROMPT string; lijst met variabelen.

Leest gegevens van een kanaal naar een lijst met variabelen. Het systeemgekozen kanaal is de opmaakroutine (kanaal Ø). Gegevensbestanddelen die gelezen worden om ze met de variabelen binnen de lijst te vergelijken, moeten door komma's van elkaar gescheiden zijn.

INPUT PROMPT K$&"Voer a.u.b. het volgende getal in ":N
INPUT A(1), B$

De gedeeltes IF MISSING en PROMPT kunnen in elke volgorde staan of helemaal afwezig zijn. De systeemgekozen INPUT PROMPT is "?". PROMPT vervangt de systeemgekozen 'prompt' door een string.

De AT optie (met rij- en kolomnummer) staat los van de PROMPT optie.

IF MISSING wordt gebruikt, indien minder gegevens van het kanaal zijn ontvangen dan vereist door de lijst met variabelen. In dit geval wordt dezelfde actie ondernomen als met READ. Zie ook **LINE INPUT**.

LET

gegeven...

LET lijst met variabelen=uitdr

Eenvoudige toewijzing; LET is facultatief, tenzij de naam van de variabele identiek is aan een sleutelwoord. Het uitlijsten of vastleggen van het programma brengt teweeg dat de LET zo ingevoerd wordt dat het programma zich aan de standaard conformeert. Kan in de rechstreekse modus uitgevoerd worden.

Een en dezelfde waarde kan aan verschillende variabelen toegewezen worden:

LET A, B(4), C = Ø
A_VAR = A_VAR + 1
A$, FRED$ = "Hij zei"&"Doe dat niet"&". "&FRED$(I:J)
LET INPUT=3

LINE INPUT

regelinvoer

Te vergelijken met INPUT, alleen wordt nu een hele regel gelezen (inclusief komma's, enz.) voor elk bestanddeel in de lijst met variabelen (die alleen string-variabelen mag bevatten).

LIST

uitlijsten

LIST £kan:beschrijving regel To beschrijving regel
LIST £kan:beschrijving regel — beschrijving regel
LIST naam blok

Lijst het volledige programma of een gedeelte ervan uit. Kan stopgezet worden met de 'stop; toets en stilgezet met de 'hold' toets. Alleen uitgevoerd in de rechstreekse modus. Het systeemgekozen kanaal is £Ø.

LIST 3ØØ
LIST 3ØØ TO 4ØØ
LIST FIRST TO 9ØØ, 1ØØØ, 2ØØØ TO LAST
LIST TO 5ØØ, 7ØØ TO

LIST MY_FUNCTION
LIST LAST

Evenals bij DELETE mag TO vervangen worden door '-'.
Voorbeeld: *LIST FIRST-1ØØ, 5ØØ-LAST*
LIST FIRST TO 1ØØ, 5ØØ TO LAST

LLIST

LLIST dat wat uitgelijst moet worden.

Identiek aan LIST, maar het systeemgekozen kanaal is £1Ø4, het 'uitlijst' — kanaal van de printer.

LOAD

Laden

LOAD £kan:bestandsnaam
LOAD naam apparaat

Laadt een bestand vanaf een bepaald kanaal, of vanaf kanaal 1Ø6, wanneer er geen kanaal is gespecificeerd (cassette of schijven). Wordt LOAD zonder parameters ingevoerd, dan wordt standaard het "zelfstart" bestand gekozen (op cassette, het eerste bestand).

LOAD
LOAD "Mijn_Programma"
LOAD "NET-Ø:"

Bevat het bestand een BASIC programma dan vervangt dit het programma dat op het moment in het geheugen zit. Als meerdere BASIC programma's in één bestand zijn opgeslagen, dan worden alle programma's in het geheugen erdoor vervangen en wordt teruggekeerd naar programma Ø.

In het bestand kunnen andere data typen en programma's zitten (zoals uitbreidingen of andere toepassingsprogramma's) die automatisch door het besturingssysteem worden verwerkt.

Zie **OPEN** voor de definitie van een apparaat- en bestandsnaam.

Variabelen worden op nul gesteld. Kan alleen worden uitgevoerd in de rechtstreekse modus. **RUN** kan echter ook in een programma opdracht worden gebruikt (zie **RUN**).

LOOK kijken

LOOK £kan AT x,y:v

Wijst aan de variabele 'v' de paletkleur toe ter hoogte van de coordinaten (x,y) op de grafische standaard-pagina of welke pagina ook, zoals gespecificeerd door het kanaalnummer. Zowel het kanaalnummer als het AT gedeelte zijn facultatief. Als het AT gedeelte weggelaten wordt, wordt de actuele positie van de positie-wijzer gebruikt.

Let wel: het gebruik van AT leidt ertoe dat het videosignaal uitgeschakeld wordt en verplaatst naar (X,Y).

LOOP Zie **DO**.

LPRINT LPRINT print-uitdrukking

Identiek aan PRINT, maar het systeem-gekozen kanaal is 1Ø4, het uitlijstkanaal van de printer.

MERGE Samenvoegen

MERGE £kan:bestandsnaam

Voegt het bestand op de schijf, band of ander kanaal samen met het actuele bestand. Regels van het nieuwe programma vervangen regels met hetzelfde nummer in het actuele bestand. Alleen uitgevoerd in de rechstreekse modus. Variabelen worden op nul gesteld.

NEW nieuw

Wist het actuele programma volledig. Alleen uitgevoerd in de rechstreekse modus. Stelt variabelen op nul.

NEW ALL Wist alle programma's uit het programmageheugen en gaat terug naar programma Ø.

NEXT Zie **FOR**.

NUMERIC numeriek

NUMERIC lijst met variabelen/reeksen

Benoemt numerieke variabelen of reeksen. De systeemgekozen ondergrens is Ø. Vergelijk **DIM.**

NUMERIC 1,A(1Ø),B(-1Ø TO 2Ø, 2 TO 4)

ON in geval van

ON uitdr GOTO lijst met regelnummers
ON uitdr GOSUB lijst met regelnummers

De waarde van de uitdrukking wordt vastgesteld, de uitkomst in een geheel getal omgezet en het resultaat N hiervan gebruikt om het N-de regelnummer uit de lijst te kiezen (er wordt geteld vanaf 1). De programma-uitvoering wordt dan hervat vanaf die regel. Als er geen N-de regelnummer is wordt geen actie ondernomen. Gebruik SELECT of een IF blok om een leesbaarder programma te krijgen.

ON A + 2 GOTO 1ØØ, 2ØØ, 3ØØ, 4ØØ, 99, 7ØØØ

OPEN openen

OPEN £kan:NAME apparaat/bestandsnaam ACCES modus
OPEN £kan:apparaat/bestandsnaam

De toegangsmodus is ofwel INPUT ofwel OUTPUT. ACCESS OUTPUT zal een nieuw bestand proberen te maken (indien op band of schijf); ACCESS input zal een bestaand bestand proberen te gebruiken. Op apparaten als VIDEO: zijn beide mogelijkheden van toepassing. De systeem-gekozen mogelijkheid is INPUT.

Verbindt een apparaat of — in geval van tape en schijf — een bestand met een kanaal. Commando's kunnen nu door naar het kanaalnummer te verwijzen gegevens van en naar het apparaat (of het bestand) lezen, schrijven of anderszins manipuleren.

OPEN £8:"DISK-1:TEST_PROGRAM" ACCESS OUTPUT

Slechts een apparaat (of bestand) kan per keer met een gegeven kanaal verbonden worden, maar een

enkel kanaal kan gebruikt worden om toegang te verkrijgen tot meerdere apparaten (of bestanden), de een na de ander.

Gebruik het CLOSE commando om een verbinding tussen een kanaal en een apparaat (of bestand) ongedaan te maken. De kanaalnummers liggen in het bereik van Ø tot 254 (255 is een ongeldig kanaalnummer dat voor speciale doeleinden gebruikt wordt).

Het BASIC systeem maakt gebruik van verschillende systeemgekozen kanalen, wanneer de kanalen niet gespecificeerd worden in de opdrachten. Het gaat om de volgende kanalen:

Ø — gebruikt voor de invoer van commando's en de uitvoer van gewone tekst (b.v. voor LIST en PRINT). Dit kanaal wordt in geval van 'reset' (of inschakelen) met het apparaat "EDITOR:" (opmaakroutine) verbonden.

De "EDITOR:" zelf maakt weer gebruik van de apparaten "KEYBOARD:" (toetsenbord) en "VIDEO:"; van toepassing zijn videomodus Ø een paginagrootte van 24,4Ø.

Dit kanaal is ook het systeem-gekozen kanaal voor COPY FROM en REDIRECT FROM.

In geval van een 'reset' wordt kanaal Ø automatisch geopend en blijft open, totdat een expliciet CLOSE commando gegeven wordt.

Merk op dat kanaal Ø het systeemgekozen kanaal voor commando's is om een en ander met het ANSI overeen te laten stemmen. De overige systeemgekozen kanalen zijn vanaf 1Ø1 genummerd, zodat u de lagere nummers kunt gebruiken voor eigen definities.

1Ø1 — gebruikt voor grafische invoer- en uitvoeropdrachten. Wanneer u het GRAPHICS commando voor het eerst gebruikt, wordt dit kanaal verbonden met het apparaat "VIDEO:" dat als volgt wordt ingesteld: videomodus 1, kleur-modus 1 en een paginagrootte van 2Ø,4Ø. Kanaal 1Ø1 blijft open totdat het expliciet gesloten wordt, bij voorbeeld door een TEXT commando.

1Ø2 — de standaardpagina voor tekst. Wordt

automatisch geopend in geval van 'reset'; paginagrootte 24,40.

1Ø3 — gebruikt voor de standaarduitvoer van geluid. Het kanaal wordt bij 'reset' met "SOUND:" verbonden.

Kanaal 1Ø3 wordt bij 'reset' automatisch geopend en pas weer gesloten wanneer daartoe expliciet opdracht gegeven wordt.

1Ø4 — gebruikt voor te verwachten 'hard-copy' (afdruk) bewerkingen. In geval van 'reset' wordt dit kanaal verbonden met het rand-apparaat "PRINTER:". Het is het systeem-gekozen kanaal voor COPY TO en REDIRECT TO. Kanaal 1Ø4 is automatisch geopend bij 'reset' en pas gesloten na een uitdrukkelijk bevel daarteo.

1Ø5 — gebruikt voor toetsenbordbewerkingen (in geval van 'reset' verbonden met "KEYBOARD:"). Blijft open zolang er geen expliciet sluitingscommando volgt.

1Ø6 — gebruikt voor invoer- en uitvoerbewerkingen die op een bestand van toepassing zijn. Telkens wanneer dat nodig is wordt het kanaal met "DISK(1):" verbonden — indien geladen; zijn er geen schijven geladen, dan wordt het met "TAPE:" verbonden.

Tot de standaard bestandsbewerkingen behoren LOAD, MERGE en VERIFY.

Kanaal 1Ø6 wordt alleen geopend wanneer dat nodig is, en gesloten na voltooiing van elke afzonderlijke bewerking — tenzij er expliciet een OPEN commando is gegeven.

1Ø7 — gebruikt voor netwerkbewerkingen. Dit is het systeemgekozen kanaal bedoeld voor SET CAPTURE FROM.

Kanaal 1Ø7 wordt alleen automatisch geopend door een commando, dat ervan uitgaat dat dit kanaal het systeemgekozen kanaal is; het wordt gesloten na voltooiing van de bewerking.

De kanalen 1ØØ-254 blijven open tenzij ze speciaal gesloten worden; maar de kanalen 1-99 worden altijd gesloten, wanneer RUN getikt wordt of wanneer een of andere bewerking plaatsvindt die alle variabelen op nul stelt. Als het BASIC een systeem-gekozen kanaal gesloten aantreft, worden alle kanalen (Ø-254) gesloten en wordt geprobeerd de systeem-gekozen kanalen weer te openen. Als dit niet mogelijk is, gaat het BASIC ervan uit dat een onherstelbare fout is gemaakt; de rand van het scherm zal nu blijven knipperen totdat de computer 'reset' wordt.

Namen van apparatuur die naar het besturingssysteem doorgestuurd worden, hebben vanwege de herkenbaarheid een dubbele punt als laatste teken. Als meerdere apparaten met dezelfde naam worden aangeduid, wordt een getal aan de naam toegevoegd; bij voorbeeld "DISK2:" of "DISK-2:".

Geldig zijn de volgende namen:

"DISK-n:"	Schijfeenheden.
"EDITOR:"	Scherm-opmaakroutine. Deze op haar beurt maakt gebruik van de apparaten "VIDEO:" en "KEYBOARD:".
"KEYBOARD:"	Toetsenbord. Omvat de externe spelpookjes.
"NET-n:"	Het ingebouwde lokale net. Het getal 'n' is het netwerkadres (tussen 1 en 32) van de computer waarmee communicatie wordt geopend. Wanneer 'n' Ø is, wordt een "algemeen" kanaal gekozen — voor het zenden van gegevens naar alle aangesloten apparatuur en voor het ontvangen van gegevens zonder eerst de "bron" op te geven.
"PRINTER:"	'Centronics-achtige' printerpoort.
"SERIAL:"	Seriegewijze RS423 I/O.
"SOUND:"	Toon generator.

"TAPE-n:" Tape decks.

"VIDEO:" Videopagina's.

Naar gelang er andere apparaten op de computer aangesloten worden, zullen er al of niet nieuwe namen in het besturingssyteem gedefinieerd worden.

In de meeste gevallen is slechts de naam van een apparaat vereist om een kanaal te openen. Wanneer er sprake is van invoer/uitvoerbewerkingen die van toepassing zijn op een bestand, moet een bestandsnaam gegeven worden.

De volledige specificate van een bestandsnaam luidt:

"apparaat-n:naam"

"Apparaat" is facultatief; wordt dit weggelaten, dan wordt het systeemgekozen massageheugen gebruikt — bij een minder uitgebreid systeem is dit "TAPE:".

"n" is het nummer van het apparaat; wordt dit nummer weggelaten, dat is het systeemgekozen nummer 1. "DISK:" zou dan verwijzen naar "DISK-1:".

"Naam" is de beschrijving van het bestand in het apparaat. Hierop zijn dezelfde opmaakregels van toepassing als op een BASIC identificatiesymbool, hoewel alleen de eerste 28 tekens belangrijk zijn.

Als de bestandsnaam geen dubbele punt omvat, wordt ervan uitgegaan dat de naam van het apparaat weggelaten is. Bij voorbeeld "SOUND" is een bestand op "TAPE:", maar "SOUND:" is het apparaat voor de geluidsvoorbrenging.

Het "naam" gedeelte van een bestandsnaam wordt door alle nieuw gedefinieerde apparaten genegeerd behalve door "TAPE:", en "DISK:". Zo is "PRINTER:PRETTY-LISTING" gelijk aan "PRINTER:".

Er zijn enkele commando's die u de mogelijkheid bieden binnen een opdracht zowel kanalen als bestandsnamen te specificeren; b.v. LOAD en SAVE.

In deze gevallen ziet de volledige specificatie er als volgt uit:

£kan:bestandsnaam

Wanneer er geen kanaalnummer wordt vermeld, wordt

gebruik gemaakt van een systeemgekozen kanaal.

OPTION optie

OPTION ANGLE DEGREES/RADIANS

Selecteert de basiseenheid voor opeenvolgende bewerkingen die gebruik maken van hoeken. De systeem-gekozen optie is radianten.

OUT OUT n,a

Schrijft byte 'a' naar de I/O poort 'n'.

PING Brengt een kort rinkelend geluid voort.

PLOT tekenen

PLOT £kan:lijst met puntjes
PLOT £kan:ANGLE uitdr
PLOT £kan:FORWARD/BACK uitdr
PLOT £kan:LEFT/RIGHT uitdr
PLOT £kan:ELLIPSE uitdr, uitdr
PLOT £kan:PAINT

PLOT gevolgd door een lijst met puntjes tekent puntjes en/of lijnen. Wanneer een PLOT commando met een puntkomma eindigt, blijft het videosignaal 'aan' nadat het commando uitgevoerd is, anders wordt het videosignaal 'uitgeschakeld'.
Met andere woorden:

PLOT x,y

zal het videosignaal — die een lijn zou trekken als hij 'aan' stond — naar positie (x,y) brengen en hem vervolgens uitschakelen.

PLOT x,y;

zal het videosignaal 'aan' laten.

De laatste twee opdrachten tekenen beide een punt ter hoogte van (x,y). Als deze coordinaten gevolgd worden door een komma, wordt het videosignaal naar de gespecificeerde positie gebracht zonder daar een

punt te tekenen (en blijft uitgeschakeld).

PLOT x1,y1;x2,y2;...

zal met het videosignaal 'aan' lijnen trekken tussen de gespecificeerde posities en het signaal 'aan' laten, als het commando met een puntkomma eindigt. Als het videosignaal al voor de uitvoering van het commando 'aan' was, zal er ook een lijn getrokken worden van de laatste signaal-positie naar positie (x1,y1).

Er wordt getekend in de actuele kleur en in overeenstemming met de actuele kleur en in overeenstemming met de actuele vorm en modus de lijnen (zie de 'Video-opties').

De coordinaten die gebruikt worden in PLOT opdrachten houden de conventies aan die voor grafisch tekenen gelden. De linker benedenhoek van de videopagina is (Ø,Ø). Van de coordinaten (x,y) is x de — vanaf links gerekend — horizontale positie en y — van beneden naar boven — de verticale positie.

ELLIPSE tekent een ellips waarvan het middelpunt de actuele videosignaal-positie is. De beide parameters die op het sleutelwoord volgen geven de horizontale en verticale afstand aan tussen het middelpunt en de omtrek, uitgedrukt in grafische schermposities. De ellips moet met het videosignaal 'uit' getekend worden, als er in het middelpunt geen punt dient te verschijnen.

Voorbeeld: *PLOT 3ØØ,35Ø, ELLIPSE 2ØØ,3ØØ*

Zo wordt voorkomen dat een punt getekend wordt.

PAINT vult een omsloten gebied (waarin zich het signaal bevindt) op met de actuele kleurmenging. Het op te vullen gebied wordt omgeven door een ononderbroken lijn die qua kleur verschilt van de oorspronkelijke kleur van de positie waar het videosignaal heeft gestaan.

Wanneer het videosignaal zich op een plaats bevindt waar een punt is getekend in de actuele 'verf'-kleur, zal PAINT geen effect hebben; dan wordt immers onmiddellijk een grensvoorwaarde ontdekt. Evenals met ELLIPSE moet u maatregelen treffen om te voorkomen dat er een punt getekend wordt.

PLOT 4ØØ,3ØØ, PAINT

In geval van een PLOT commando gaat het systeem naar kanaal 1Ø1.

POKE poken, porren

POKE adres,waarde

Bepaalt de waarde van de gespecificeerde Z80 geheugenpositie.

PRINT (AF) drukken £

PRINT £kan, AT rij-nummer,kolomnummer:uitvoerlijst
PRINT £kan, USING regelnummer:uitvoer-lijst
PRINT £kan, USING string:uitvoer-lijst

Een bestanddeel uit de uitvoer-lijst kan ofwel een uitdrukking zijn ofwel het woord TAB gevolgd door een kolomnummer tussen haakjes. De bestanddelen kunnen door komma's of door puntkomma's van elkaar gescheiden worden. Een puntkomma genereert een lege string; een komma voert spaties in, totdat de volgende printzone begint. TAB voert spaties in tot aan de gespecificeerde kolom. Een uitvoerlijst die met een komma of een puntkomma eindigt genereert niet een einde-regel opdracht. Kan uitgevoerd worden in de rechstreekse modus.

De AT optie brengt de positiewijzer naar de gespecificeerde rij en kolom, voordat de lijst afgedrukt wordt. Het facultatieve kanaalnummer stuurt de uitvoer naar een andere bestemming (het systeemgekozen kanaal is de standaardpagina voor tekst).

De specificatie van de positie na AT houdt de conventies aan die voor het plaatsen van tekst gelden. De linker bovenhoek van de videopagina heeft de tekstcoordinaten (1,1). De vijftiende kolom op de tweede regel heeft de tekstcoordinaten (2,15).

```
PRINT "VALUE=";A
PRINT AT x,y:"o";
```

De USING optie regelt de opmaak van de uitvoer. Het regelnummer moet het nummer zijn van een IMAGE opdracht. Zie IMAGE voor details van de opmaakspecificatie.

PROGRAM programma

PROGRAM naam (lijst met variabelen)

Definieert de naam van het lopende programma, zodat dit in CHAIN opdrachten gebruikt kan worden. De naam van het programma moet de standaardvorm hebben van een identificatiesymbool.

PROGRAM "Mijn_programma" (A,B$)

De lijst met variabelen (indien ingesloten) maakt het mogelijk dat gespecificeerde parameters uit een ander programma wat hun waarde betreft doorgegeven worden. Zie **CHAIN** en **EDIT**.

RANDOMIZE toevallig verdelen, willekeurig maken

Normaal begint elke gang van een programma met dezelfde willekeurige getallenrekks. RANDOMIZE zorgt voor een nieuwe reeks van willekeurige getallen.

READ lezen

READ lijst met variabelen
READ IF MISSING regelnummer:lijst met variabelen
READ IF MISSING EXIT DO:lijst met variabelen

Leest gegevens van de DATA opdrachten; de IF MISSING actie wordt uitgevoerd zodra een poging ondernomen wordt verder te lezen, terwijl de gegevens op zijn.

READ, A,B$(i)

REDIRECT naar een andere bestemming sturen

REDIRECT FROM £kan TO £kan

leest invoer vanaf eerste kanaal en stuurt deze naar het tweede kanaal, totdat het einde van een bestand bereikt wordt, det 'stop' toets aangeslagen wordt of een fout optreedt op een van de kanalen. De versturing van gegevens van het ene naar het andere kanaal kan ook gestopt worden door gebruik te maken van het

ongeldige worden door gebruik te maken van het ongeldige kanaalnummer £255 als het FROM kanaal in een latere REDIRECT opdracht.

REM

opmerking (REMark)
Regel met opmerkingen.

REM moet aan het begin van de regel staan. Vanwege een grotere flexibiliteit wordt '!' aanbevolen.

RENUMBER

opnieuw nummeren

RENUMBER beschrijving regel TO beschrijving regel AT uitdr STEP uitdr.

Opnieuw nummeren van het volledige programma of een gedeelte ervan. Alleen uitgevoerd in de rechstreekse modus.

RENUMBER FIRST TO 1ØØ
RENUMBER 1Ø TO 1ØØ AT 3ØØ STEP 1Ø
RENUMBER STEP 1ØØ
RENUMBER MY_FUNCTION AT 5ØØØ

STEP en AT kunnen in elke volgorde staan of helemaal weggelaten worden. Als STEP niet nader gespecificeerd wordt is de systeemgekozen STEP 1Ø. Als AT wordt weggelaten, wordt het eerste regelnummer gebruikt van het segment dat opnieuw genummerd moet worden. Worden er geen regelnummers gegeven, dan wordt het hele programma opnieuw genummerd; de systeemgekozen AT wordt nu 1ØØ. Vergelijk **DELETE** voor de syntaxis van de regels.

Alle verwijzingen in het programmablok naar opnieuw genummerde regels worden veranderd.

RENUMBER kan niet de volgorde van regels in een programma veranderen. Dus als de opnieuw genummerde regels regels overlappen of omgeven, of wanneer de opnieuw genummerde regels een nieuwe plaats in de volgorde zouden innemen of een te hoog regelnummer zouden creeren, dan wordt het RENUMBER commando niet uitgevoerd en blijft de tekst van het programma ongewijzigd.

RESTORE terugzetten

RESTORE
RESTORE regelnummer

Opnieuw gebruiken van DATA (toegepast in combinatie met READ opdrachten; de computer gaat terug naar het begin van het programma of naar de aangegeven regel).

RETRY opnieuw proberen

Wordt gebruikt als uitweg uit een functie voor de afhandeling van uitzonderingen de besturing wordt teruggegeven aan de regel of opdracht die de uitzondering heeft veroorzaakt. Vergelijk **CONTINUE**. Wanneer een functie voor de afhandeling van uitzonderingen gebruikt wordt om de 'stop' — toets op te vangen, moet RETRY gebruikt worden om het programma voort te zetten.

RETURN terugkeren

Van een subroutine terugkeren die door GOSUB aangeroepen is.

RUN draaien, uitvoeren

RUN
RUN regelnummer
RUN £kan:bestandsnaam
RUN naam:apparaat

RUN alleen voert het actuele programma vanaf de eerste regel uit. Als er een regelnummer gegeven wordt, begint de uitvoering vanaf die regel.

Wordt een bestandsnaam gegeven (met een kanaalnummer naar keuze), dan wordt het programma geladen en gedraaid. Stelt variabelen op nul. Met RUN kunnen parameters aan programma's worden doorgegeven. Deze moeten wel overeen stemmen met de voor het programma gebruikte parameters. Zie parameters. Zie **PROGRAM**.

SAVE opslaan, vastleggen

SAVE £kan:bestandsnaam
SAVE naam apparaat
SAVE ALL £kan: bestandsnaam

Legt het actuele programma vast. Dit gebeurt via kanaal 1Ø6 (systeemgekozen).

Als er geen **PROGRAMMA** naam is moeten er dubbele aanhalingstekens worden gebruikt. Voorbeeld — (SAVE " ").

SAVE ALL zal alle programma's bewaren die op dat moment in de computer aanwezig zijn.

SELECT kiezen

select-regel
case-regel
    een willekeurig aantal opdrachten of blokken
case-regel optie
    een willekeurig aantal opdrachten of blokken
end-select-regel

select-regel:
SELECT CASE uitdr

case-regel:
CASE uitdr
CASE uitdr TO uitdr
CASE IS uitdr met verbe
CASE ELSE

end-select-regel:
END SELECT

Het SELECT blok bestaat uit een aantal opdrachten om de variabele of de uitdrukking te toetsen aan een aantal wisselende voorwaarden.

Het woord CASE op de SELECT regel is facultatief tenzij de uitdrukking met een identificatiesymbool CASE begint.

Voorbeeld: *SELECT CASE CASE+23*

Het aantal CASE regels is onbeperkt. De cases worden getest in volgorde van regelnummers. Het maakt niets

uit wanneer u nog een aantal extra case-regels op CASE ELSE laat volgen; deze kunnen normaal gesproken toch niet bereikt worden.

Meerdere cases kunnen op een regel gecombineerd worden door ze met komma's van elkaar te scheiden.

Voorbeeld: *CASE 1,2,3 TO 6,99*

```
SELECT CASE N
CASE 1
  PRINT "eerste case"
CASE 2 TO 9,11,21
  PRINT "nog enkele cases"
CASE IS<=A+2Ø
  PRINT "nog meer cases"
CASE ELSE
  PRINT "de rest van de cases"
END SELECT
```

De CASE ELSE regel kan slechts een keer gebruikt worden en moet volgen op alle andere CASE regels. De overige CASE regels kunnen in elke gewenste volgorde gebruikt worden, waarbij de regels tussen twee CASE regels een blok vormen. De besturing kan niet van buiten een SELECT blok naar binnen overgebracht worden.

SELECT blokken voor strings zijn eveneens beschikbaar.

SET instellen

Stelt de actuele waarden in van een machine-optie. Zie 'Machine-opties', 'Video-opties' en 'geluidsopties'. Vergelijk **ASK** en **TOGGLE**.

SOUND geluid

SOUND £kan:PITCH uitdr, DURATION uitdr, LEFT uitdr, RIGHT uitdr, SOURCE uitdr, STYLE uitdr, ENVELOPE uitdr, SYNC uitdr, INTERRUPT

Met dit commando kan het geluid volledig geregeld worden. De parameters kunnen in elke volgorde staan.

Het getal achter PITCH kan elk getal zijn tussen Ø en 127, ofschoon goede resultaten doorgaans alleen bereikt worden in het bereik tot ongeveer 83. Binnen dit bereik zal elke verhoging met 1 resulteren in een verhoging van de toonhoogte met een halve toon. Toonwaarde 37 is gelijk aan de eengestreepte C; dit is tevens de systeemgekozen waarde.

DURATION geeft de duur van het geluid aan (niet de duur van de RELEASE fasen van de envelop), uitgedrukt in 'tellen' (een tel is 1/5Ø seconde). De systeemgekozen waarde bedraagt 5Ø tellen.

De beide parameters LEFT en RIGHT specificeren de totale geluidssterkte die naar de beide luidsprekers gaat. De waarden liggen tussen Ø (geen geluid) en 255 (het maximumvolume van de machine — tevens de systeemgekozen waarde). Als er geen gebruik gemaakt wordt van stereo-apparatuur, wordt het volume bepaald door de waarden voor het linker en het rechter kanaal bij elkaar op te tellen.

SOURCE specificeert de toongenerator die gebruikt wordt; de waarden liggen tyssen Ø en 3 (systeemgekozen waarde: Ø). Nummer 3 is de 'ruisgenerator' (die de toonwaarden negeert).

De STYLE parameter ligt binnen het bereik van Ø tot 255 (systeemgekozen waarde:Ø); zie voor het effect hiervan de 'Geluidsopties'.

ENVELOPE specificeert het nummer van de envelop die van toepassing is op de SOUND opdracht. Zie **ENVELOPE**. 255 is een ingebouwde envelop (systeem-gekozen).

SYNC biedt u de mogelijkheid 1, 2 of 3 geluiden van verschilledne 'bronnen' synchroon te laten beginnen. Wanneer u bij voorbeeld 3 geluiden tegelijkertijd wilt laten beginnen, kunt u elk hiervan de instructie SYNC 2 geven, wat resulteert in een gelijktijdig begin met de beide andere geluiden. (De systeemgekozen waarde is Ø).

Indien inbegrepen, bewerkstelligt INTERRUPT dat eventuele muziekjes vervangen worden door een nieuw muziekje dat van dezelfde bron afkomstig is.

SPOKE

SPOKE segment,adres,waarde

Als POKE, maar de waarde wordt naar het systeemadres geschreven in het gespecificeerde segment.

START beginnen (te draaien)

Als er geen programma is geladen, wordt door dit commando het eerste bestand op kanaal 1Ø6 geladen en gedraaid. Is er wel een programma geladen, dan fungeert START als RUN.

STOP STOP

Stopt de uitvoering (drukt STOP af).

Merk op dat u CONTINUE mag gebruiken na een STOP instructie.

STRING STRING

STRING lijst n met variabelen/reeksen.

Geeft de maximale lengte aan van tekstuele variabelen en reeksen. De systeemgekozen lengte bedraagt 132. Wanneer n achter het woord STRING of achter de benoeming van de variabele staat, wordt de lengte bepaald op n. De systeemgekozen ondergrens voor een reeks is Ø.

*STRING*8 LAST__NAME$*2Ø,FIRST__NAME$, MIDDLE__NAME$*

In dit voorbeeld heeft LAST__NAME$ een maximale lengte van 2Ø; FIRST__NAME$ en MIDDLE__NAME$ zijn beide maximaal 9 tekens lang.

STRING NAME$

In dit geval heeft NAME$ een maximale lengte van 132 tekens.

*STRING NAME$ (4 TO 99)*1Ø*

Deze reeks bestaat uit 96 elementen die elk uit 1Ø tekens bestaan.

Merk op: met een DIM opdracht kan de lengte van een tekstuele variabele niet bepaald worden.

TEXT tekst

TEXT
TEXT 4Ø
TEXT 8Ø

Opent een pagina voor tekstuele doeleinden die het volledige scherm beslaat, behalve het gebied van de statusregel. De grafische standaardpagina wordt gesloten als deze open is.

4Ø en 8Ø specificeren het aantal kolommen op het scherm. Als het aantal kolommen niet gespecificeerd wordt, is waarde 4Ø van toepassing.

THEN Zie **IF**.

TIME TIME tijd-string

Specificeert de tijd zoals door de computer bijgehouden. De tijd heeft de volgende indeling:

TIME "UU:MM:SS"

Kan gebruikt worden in de rechstreekse modus. Zie de TIME$ funtie; zie ook het commando **WAIT DELAY** en de machine-optie **TIMER**.

TOGGLE synoniem met flip-flop

Is van invloed op machine-opties die slechts twee mogelijke waarden hebben (b.v. 'aan' en 'uit'); van de actuele waarde overschakelen naar de andere waarde. Zie 'Machine-opties', 'Video-opties' en 'Geluidsopties'; vergelijk SET en ASK.

TRACE volgen

TRACE ON TO £kan
TRACE OFF

Na TRACE ON volgt het nummer van de regel die op dat moment uitgevoerd wordt. De uitvoer wordt naar het kanaal Ø gedirigeerd tenzij er een kanaalnummer gespecificeerd wordt.

TYPE

TYPE

Speciale instructie om BASIC te verlaten en toegang te krijgen tot de ingebouwde tekstverwerker. Bij uitvoering worden alle BASIC programma's en variabelen gewist. Daarom wordt de gebruiker gevraagd de instructie met 'Enter' te bevestigen.

VERIFY

verifieren

VERIFY £kan:bestandsnaam

Verifieert of een programma correct is opgeslagen; vergelijkt het lopende programma-bestand met het gespecificeerde bestand en geeft een foutboodschap als beide bestanden niet identiek zijn. Kanaal 1Ø6 wordt als het systeem-gekozen kanaal gebruikt. Alleen uitgevoerd in de rechstreekse modus.

WAIT DELAY

Wachten

WAIT DELAY uitdr

Zorgt dat de uitvoering van het programma voor een bepaalde, in secondes opgegeven tijdsduur wordt onderbroken.

WAIT DELAY 6Ø

Onderbreekt de programma-uitvoering één minuut.
De maximale onderbrekingstijd is 32,767 seconden. Zie de machine optie **TIMER** voor automatische blokkeertijden bij ononderbroken programma-uitvoering.
Het woord DELAY is facultatief.

WHEN

wanneer

WHEN EXCEPTION USE naam functie
opdrachten
END WHEN

Specificeert de functie voor de afhandeling van uitzonderingen die gebruikt moet worden, wanneer zich binnen het WHEN blok een uitzondering voordoet die veroorzaakt wordt door het programmaverloop.

De programma-instructies kunnen ook nog geneste WHEN blokken omvatten.

Zie **HANDLER**.

Bepaalde systeemvariabelen en machinefuncties kunnen direct via BASIC bestuurd worden; we hebben hier te maken met de zogenaamde machine-opties. Aan een optie kan een waarde toegewezen worden door gebruik te maken van het commando SET. Indien aangegeven, kunnen de onderstaande opties ook in combinatie met ASK of TOGGLE gehanteerd worden.

DEFAULT CHANNEL Systeemgekozen kanaal

SET DEFAULT CHANNEL uitdr

Specificeert het standaard systeemkanaal. Dit kanaal kan vervolgens worden gebruikt door functie-programma's die gericht zijn op communicatie met de gebruiker, maar de betekenis van de momenteel gebruikte kanalen niet kennen.

Dit kanaal wordt met name gebruikt door programma's die reageren op het via het besturings-systeem gestuurde HELP commando.

Voor de instelling met SET heeft dit kanaal het nummer Ø.

EDITOR BUFFER SET EDITOR BUFFER uitdr

Definieert de afmeting van het buffergeheugen van de opmaakroutine, en wel in blokken van 256 bytes. Kan toegepast worden in combinatie met ASK.

EDITOR KEY SET EDITOR KEY kanaalnummer

Maakt het mogelijk dat het gespecificeerde kanaal gebruikt wordt voor de invoer vanaf het toetsenbord voor de opmaakroutine. Kan gebruikt worden in combinatie met ASK.

EDITOR VIDEO SET EDITOR VIDEO kanaalnummer

Hierdoor kan het gespecificeerde kanaal gebruikt worden als tekstuele pagina voor de opmaakroutine. Kan toegepast worden in combinatie met ASK.

FAST SAVE Snel bewaren

SET FAST SAVE ON/OFF

Geeft de transmissiesnelheid voor snelle opslag op cassette. De standaard transmissiesnelheid is in dit geval ongeveer 24ØØ baud. Is FAST SAVE op OFF gezet, dan wordt de transmissiesnelheid gehalveerd.

Bij het laden van cassette wordt automatisch rekening gehouden met de verschillende transmissiesnelheden bij de opslag.

Kan met TOGGLE worden gebruikt.

FKEY

SET £kan:FKEY nummer toets string

Stelt de functietoets zo in dat bij elke aanslag de gespecificeerde string geproduceerd wordt (een lege string heeft een uitzondering tot gevolt). Het systeemgekozen kanaal is 1Ø5.

De functietoetsen zijn van 1 tot 16 genummerd. De nummers 1 tot en met 8 zijn de niet verschoven functietoetsen; de nummers 9 tot en met 16 zijn de verschoven equivalenten van de eerste acht toetsen.

De functietoetsen zijn van tevoren door het systeem gedefinieerd; bij herdefiniering van de toetsen zullen de systeemgekozen definities wegvallen.

Gebruik voor automatische wagenterugloop &CHR$(13).

INTERRUPT

ASK INTERRUPT CODE

Vraagt naar de onderbrekingscode van het programma voor de laatste onderbreking. Kan gebruikt worden in combinatie met ASK.

SET INTERRUPT KEY ON/OFF

Indien 'AAN' wordt er een onderbreking van het programma veroorzaakt, wanneer u ook maar een toets aanslaat. Kan gebruikt worden in combinatie met TOGGLE.

SET INTERRUPT NET ON/OFF

De programmatuur wordt al of niet onderbroken bij ontvangst van gegevens van het netwerk.

SET INTERRUPT STOP ON/OFF

De programmatuur wordt al dan niet onderbroken via de 'STOP' toets. Kan gebruikt worden in combinatie met TOGGLE.

KEY CLICK SET KEY CLICK ON/OFF

Bepaalt of er bij elke aanslag van een toets al of niet een klik wordt gehoord. Kan gebruikt worden in combinatie met TOGGLE.

KEY DELAY SET KEY DELAY uitdr

Stelt de beginvertraging van het toetsenbord in — in eenheden van 1/5Ø seconde — voordat de automatische herhaling start. Kan gebruikt worden in combinatie met ASK.

KEY RATE SET KEY RATE uitdr

Specificeert de automatische herhalingssnelheid van het toetsenbord, uitgedrukt in eenheden van 1/5Ø seconde — voordat de automatische herhaling start. Kan gebruikt worden in combinatie met ASK.

NET CHANNEL ASK NET CHANNEL var

Geeft het nummer van het kanaal dat aan de beurt is voor het lezen van gegevens in een netwerk buffer.

Als de eerste byte via dit kanaal gelezen is, wijst NET CHANNEL het volgende te bedienen kanaal aan. Wanneer geen van de kanalen meer gegevens hebben, wordt de waarde 255 teruggeven.

NET MACHINE ASK NET MACHINE var

Deze wordt tegelijk met NET CHANNEL bijgewerkt en geeft het netwerknummer van het apparaat op afstand. Dit is met name van belang, wanneer de gegevens op het "algemene" kanaal (NET-Ø) worden ontvangen.

NET NUMBER SET NET NUMBER uitdr

Stelt het netwerk adres nummer van de computer in. Dit

nummer moet liggen tussen 1 en 32. Begint met Ø, welk getal niet als netwerkadres gebruikt kan worden.

Een netwerknummer moet worden opgegeven voordat het netwerk gebruikt wordt en mag niet gelijk zijn aan netwerknummers van andere op het netwerk aangesloten computers.

SERIAL BAUD

SET SERIAL BAUD uitdr

De parameter (in het breik Ø-15) bepaalt het aantal baud voor de RS232 poort, en wel volgens de onderstaande code. Kan toegepast worden in combinatie met ASK.

Ø = > 50 baud	11 = > 36ØØ baud
1 = > 75 baud	12 = > 48ØØ baud
2 = > 11Ø baud	13 = > 72ØØ baud
3 = > 134,5 baud	14 = > 96ØØ baud
4 = > 15Ø baud	15 = > 96ØØ baud
5 = > 2ØØ baud	
6 = > 3ØØ baud	
7 = > 6ØØ baud	
8 = > 6ØØ baud	
9 = > 18ØØ baud	
1Ø = > 24ØØ baud	

SERIAL FORMAT

SET SERIAL FORMAT uitdr

Definieert de woodrangschikking voor het besturingsprogramma van het seriele apparaat. Als hieronder wordt de rangschikking geregeld door de binaire cijfers in het getal:

bit	waarde	resultaat	
Ø	Ø	8 bits	
	1	7 bits	
1	Ø	geen pariteit	
2	Ø	even pariteit	wordt
	1	oneven-pariteit	genegeerd, indien bit 1 nul is.
3	Ø	twee stopbits	
	1	een stopbit	

Bit 4 en verder moeten Ø zijn.

STATUS SET STATUS ON/OFF

Zet de 'statusregel' (boven in het beeld) aan en uit. Kan gebruikt worden in combinatie met TOGGLE.

REM1 SET REM1 ON/OFF

Bestuurt schakelaar 1 afstandsbediening. (Wordt ook bestuurd door magneetband-bewerkingen.)

REM2 SET REM2 ON/OFF

Als hierboven, maar nu voor schakelaar 2 afstandsbediening.

TAPE LEVEL SET TAPE LEVEL uitdr

Regelt het volumeniveau bij gegevensopslag op cassette. Aanvaardbaar zijn niveau's tussen 1 en 6, waarbij het volume per niveau verdubbelt. Niveau 1 is gelijk aan ongeveer 40 mV topwaarde. De standaard instelling is 2.

TAPE SOUND SET TAPE SOUND ON/OFF

Regelt de transmissie van geluid vanaf de invoer tot de uitvoer. Hierdoor kunt u muziek of gesproken woorden direct vanaf de band nar de interne luidspreker of de hi-fit uitvoer leiden. Kan gebruikt worden in combinatie met TOGGLE.

TIMER SET TIMER uitdr

Start een timer die bij nul een programma-onderbreking veroorzaakt. De waarde wordt in seconden gegeven, tot 255 maximaal. Door de timer de waarde Ø te geven wordt deze stopgezet zonder een onderbreking te veroorzaken.

De timer stopt altijd bij Ø en moet dan steeds opnieuw worden gestart.

De uitzonderingsonderbreking (EXTYPE 9229) moet door een onderbrekingsafhandelingsprogramma worden behandeld, anders wordt de programma-uitvoering stopgezet. Na de uitzonderings-onderbreking, kent ASK INTERRUPT CODE A aan A de

waarde 64 toe, wanneer de onderbreking door de timer werd veroorzaakt.

VARIABLE

SET variabele-nummer, uitdr
ASK variabele-nummer var
TOGGLE variabele-nummer

De opgegeven besturingssysteem variabele instellen, opvragen of omschakelen (zie technisch handboek Enterprise).

Deze zijn van invloed op het ingebouwdde video-apparaat dat vele videopagina's met telkens verschillende parameters kan bevatten. De commando's die van toepassing zijn op afzonder-lijke pagina's kunnen vergezeld gaan van een kanaal-specificatie, maar als dit weggelaten wordt nemen sommige commando's de standaardpagina voor tekst aan (1Ø2) en andere de standaardpagina voor grafische afbeeldingen (1Ø1) — zie hieronder.

Merk op dat COLOR altijd een acceptabele vervanging is van COLOUR (kleur).

ATTRIBUTES

SET ATTRIBUTES uitdr

Stelt stuurcodes in voor bewerkingen in de speciale video modus (nummer 15). De codes hebben de volgende betekenis:

1 — teken in Ø'len (kleur papier)
2 — teken zonder pixels te veranderen.
4 — teken zonder inkt kenmerken te veranderen.
8 — tekenen zonder papierkenmerken te veranderen.
16 — afdrukken in Ø'len.
32 — afdrukken zonder pixels te veranderen.
64 — afdrukken zonder inkt kenmerken te veranderen.
128 — afdrukken zonder papierkenmerken te veranderen.

Voor gecombineerde effecten dienen de getallen opgeteld te worden. De standaard instelling is Ø.

BEAM

SET £kan:BEAM ON/OFF

De actuele plaats waar grafisch getekend wordt, wordt de 'beam' positie genoemd. Telkens wanneer het videosignaal verplaatst wordt, kan deze al of niet (afhankelijk van de vraag of hij 'aan' of 'uit' is) een lijn achter zich laten.

BIAS

SET £kan:BIAS nummer kleur

Bepaalt welke groep kleuren de nummers 8-15 zullen

zijn binnen het palet. Het nummer dat is het commando aangegeven wordt is het standaard codenummer van welke kleur ook binnen de gewenste groep; er zijn 32 effectieve waarden. De bias kan ook nader gespecificeerd worden door gebruik te maken van de COLOUR functie.

SET BIAS COLOUR (Ø.,6,.4)

Het systeemgekozen kanaalnummer is £1Ø1. Maar de bias is van toepassing op elk palet dat voor de weergave gebruikt wordt.

BORDER

SET £kan:BORDER nummer kleur

Wijzigt de kleur van de paginarand in overeenstemming met het gespecificeerde standaard codenummer. Het systeemgekozen kanaalnummer is £1Ø1.

CHARACTER

SET £kan:CHARACTER n,a,b,c,d,e,f,g,h,i

Definieert het patroon van het teken met de ASCII-code 'n'. Elk van de parameters a-i definieert een rij van het patroon, te beginnen met de bovenste. Bij de aanmaak van tekens kan de BIN functie gebruikt worden om elke pixel in een rij als een Ø of een 1 te specificeren.

Hoewel een kanaalnummer gespecificeerd wordt, is het commando van invloed op alle videopagina's. Het systeemgekozen kanaalnummer is £1Ø2.

COLOUR

SET £kan: COLOUR paletnummer kleurcode

Geeft de waarde van één kleur in het palet (zie **PALETTE**). Paletnummers liggen tussen Ø en 7. De kleurcode ligt in het standaard bereik van Ø tot 255 (of gespecificeerd met de RGB functie).

CURSOR

SET £kan:CURSOR CHARACTER code
SET £kan:CURSOR COLOUR paletnummer

Specificeert de ASCII-code van het teken en/of het paletnummer van de kleur die gebruikt moet worden voor de positiewijzer. Het systeemgekozen kanaalnummer is £1Ø2.

INK

SET £kan:INK nummer kleur

Stelt de actuele tekenkleur in. Het nummer van de kleur is een paletnummer behalve in kleurmodus 3 (256 kleuren) waar het een standaard codenummer is. Het systeemgekozen kanaalnummer is £1Ø1.

LINE MODE

SET £kan:LINE MODE parameter

Bepaalt de interactie tussen de kleuren op het scherm en de nieuwe lijnen die getekend worden. In modus Ø (de systeemgekozen modus) overschrijft een nieuwe lijn alles dat al getekend is. In de modi 1-3 wordt de kleur voor ieder mogelijk gedeelte van de nieuwe lijn bepaald door paletnummers van de oude en de nieuwe kleurmengingen te combineren; en wel als volgt:

modus 1 — 'of'
modus 2 — 'en'
modus 3 — 'non-equivalentie'.
het systeem gekozen kanaalnummer is £1Ø1

LINE STYLE

SET £kan:LINE STYLE parameter

De actuele vorm van de lijn kan ingesteld worden met de waarden in het bereik van 1-14, waardoor verschillende soorten gestippelde lijnen mogelijk worden gemaakt. Het systeemgekozen kanaalnummer is £1Ø1.

Het systeemgekozen kanaal is £1Ø1.

PALETTE

SET £kan:PALETTE a,b,c,d,e,f,g,h

Stelt de waarde in van de eerste 8 kleuren in het palet; deze worden dan gebruikt door video-opties als SET PAPER en SET INK. Het systeemgekozen kanaalnummer is £1Ø1.

Alleen de eerste vier kleuren kunnen gebruikt worden in kleurmodus Ø. Wanneer alleen de eerste 2 of 4 kleuren gespecificeerd worden, neemt de rest automatisch de waarde van kleur Ø aan.

De kleuren die in het palet opgenomen worden, worden ofwel door standaard codenummers in het bereik van Ø-255 gespecificeerd, ofwel door middel van

de COLOUR functie (zie 'Ingebouwde functies en variabelen').

Het palet bestaat in totaliteit uit 16 kleuren, maar alleen de eerste 8 kunnen helemaal vrij gekozen worden. Zie de BIAS optie voor bijzonderheden over de resterende 8 kleuren.

PAPER

SET £kan:PAPER: nummer kleur

Selecteert de kleur die gebruikt zal worden als achtergrond bij het printen of tekenen. In geval van kleurmodus 3 wordt de kleur van het papier gedefinieerd door een standaard codenummer, in andere modi door een paletnummer. Het systeemgekozen kanaalnummer is £1Ø1.

Voor de grafische pagina geldt (modus 1 en 5 zie VIDEO MODE optie) dat het PAPER commando alleen effect zal hebben, wanneer de pagina op blank is gesteld — wanneer een nieuwe achtergrond is geselecteerd voor de grafische weergave.

De geldige papierkleuren voor een 8Ø-kolommen pagina voor tekst (videomodus 2) zijn de paletnummers Ø, 2, 4 en 6. Deze worden gekoppeld aan de respectieve kleurmengingen 1, 3, 5 en 7; een teken dat in een specifieke kleurmenging gedrukt wordt, krijgt vanzelf zijn eigen bijbehorende achtergrondkleur. Een kleurenpaar voor menging en achtergrond selecteert u door SET PAPER of SET INK te tikken, gevolgd door een van beide relevante paletnummers.

Voor een 4Ø-kolommen pagina voor tekst geldt hetzelfde, alleen zijn er nu slechts twee kleurenparen beschikbaar.

SCROLL

SET £kan:SCROLL ON/OFF

In- of uitschakelen van 'automatisch opschuiven' van de tekst. Het systeemgekozen kanaalnummer is £1Ø2.

SET £kan:SCROLL UP/DOWN n,m

Schuift het beeld omhoog en omlaag van regel (n-32) naar (m-32). Het systeemgekozen kanaalnummer is £1Ø2.

VIDEO COLOUR

SET VIDEO COLOUR uitdr

Stelt de kleurmodus in voor videopagina's die daarna geopend worden. (Het kanaalnummer wordt genegeerd.)

Wanneer u een tekstpagina definieert moet altijd kleurmodus Ø gekozen worden. Wat de grafische pagina's met een hoog scheidingsvermogen betreft, hebben de kleurmodi de volgende betekenis.

modus Ø — 2 kleuren; horizontaal scheidingsvermogen 64Ø
modus 1 — 4 kleuren; horizontaal scheidingsvermogen 32Ø
modus 2 — 16 kleuren; horizontaal scheidingsvermogen 16Ø
modus 3 — 256 kleuren; horizontaal scheidingsvermogen 8Ø

Op een LORES grafsiche pagina (die maar de helft van het geheugen gebruikt in vergelijking met HIRES) blijft de hoeveelheid kleuren hetzelfde als hierboven, maar het horizontale scheidingsvermogen wordt gehalveerd.

VIDEO MODE

SET VIDEO MODE uitdr

Stelt de videomodus in voor pagina's die daarna geopend worden. (Het kanaalnummer wordt genegeerd).

De parameter waarden zijn als volgt:

modus Ø — tekstpagina met 4Ø kolommen (2 kleurenparen)
modus 1 — grafische pagina met hoog scheidingsvermogen.
modus 2 — tekstpagina met 8Ø kolommen (4 kleurenparen).
modus 5 — grafische pagina met laag schedingsvermogen.
modus 15 — 'attribuut' grafisch scherm.

VIDEO X

SET VIDEO X uitdr

Definieert de horizontale afmeting van videopagina's die daarna geopend worden. (Het kanaalnummer

wordt genegeerd.)

De afmeting wordt uitgedrukt in tekenposities in het bereik van 2-42, uitgaande van het coordinatenstelsel dat voor tekstpagina's geldt.

VIDEO Y

SET VIDEO Y uitdr

Als hierboven, alleen wordt de verticale afmeting van de pagina gedefinieerd (uitgedrukt in een aantal tekenrijen in het bereik van 1-255).

Deze zijn van invloed op de ingebouwde toongenerator.

SOUND BUFFER

SET SOUND BUFFER uitdr

Stelt de hoeveelheid geheugenruimte in van de envelop, en wel door een daarna te openen signaal-generator. De uitdrukking is het aantal fasen. Mogelijke waarden liggen tussen 1 en 255; de systeem-gekozen waarde is 2Ø. Kan toegepast worden in combinatie met ASK.

SOUND STYLE

De waarden van een STYLE parameter in een SOUND opdracht (zie 'Commando's en opdrachten') hebben het volgende effect:

op toonkanaal Ø:

16 — lage vervorming
32 — middelmatige vervorming
48 — hoge vervorming
64 — gebruik high pass filter. Toonkanaal 1 is klok.
128 — ringmodulatie met kanaal 2.

Op toonkanaal 1:

als kanaal Ø, alleen maakt het high pass filter gebruik van toonkanaal 2; de ringmodulator benut het ruiskanaal (kanaal 3);

Op toonkanaal 2:

als kanaal Ø, alleen benut het high pass filter het ruiskanaal (kanaal 3); de ringmodulator maakt gebruik van kanaal Ø.

Op kanaal 3 (ruiskanaal)

1,2,3 — gebruik toonkanaal Ø, 1 of 2 klokfrequentie in plaats van de standaard 31.25 KHz frequentie;

4,8,12 — kies ruisfrequentie van 15, 11 of 9-bits polynome tellers in plaats van de standaard 17-bits teller;

16 — vervang een 7-bits polynome door de 17-bits teller;

32 — gebruik low pass filter op ruiskanaal en gebruik kanaal 2 als de klok;

64 — gebruik high pass filter op ruiskanaal en gebruik kanaal Ø als de klok;

128 — gebruik ringmodulator met toonkanaal 1.

SPEAKER

Wilt u een combinatie van SOUND STYLE opties kiezen, voeg dan de waarden van de afzonderlijke opties samen en specificeer het hieruit resulterende getal als de STYLE parameter.

SET SPEAKER ON/OFF

Regelt de geluidsuitvoer vanaf de interne luidspreker; SET SPEAKER OFF wordt gebruikt om de machine op snelle wijze tot zwijgen te brengen.

Trigonometrische functies werken met graden of met radianten. Minimal BASIC functies zijn; ABS, ATN, COS, EXP, INT, LOG, RND, SGN, SIN, TAB en TAN.

ABS(X) De absolute waarde van een getal. Dit komt neer op het verwijderen van het plus- of het minteken. *ABS(-9)* is dus 9.

ACOS(X) De hoek die in verband gebracht wordt met cosinus X, d.w.z. het tegenovergestelde van COS. Zo is *ACOS(COS(X))* X.

ANGLE(X,Y) De hoek tussen de positieve x-aslijn en de lijn die punt (Ø,Ø) verbindt met punt (X,Y).

ASIN(X) De hoek waarvan X de inus is.

ATN(X) De hoek waarvan X de tangens is.

BIN(X) Retourneert het getal dat met het gegeven binaire getal correspondeert. Voorbeeld: *BIN(11ØØ1)* is 25.

BLACK De kleur 'zwart', gelijk aan COLOUR (Ø,Ø,Ø).

BLUE De kleur 'blauw', gelijk aan COLOUR (Ø,Ø,1).

CEIL(X) Geeft het kleinste hele getal dat niet minder is dan X. X wordt met andere woorden naar boven afgerond: *CEIL(3,45)* is 4 en *CEIL(-3,45)* is -3.

CHR$(X) Retourneert het teken waarvan X het ASCII code-nummer is.

COLOR(R,G,B) Identiek met COLOUR(R,G,B).

COLOUR(R,G,B) Retourneet het machine-afhankelijke kleurnummer dat equivalent is aan de gespecificeerde menging rood, groen en blauw. R specificeert het percentage rood (Ø tot 1), G het percent groen (Ø tot 1) en B het percentage blauw (Ø tot 1)

Bij voorbeeld: *SET INK COLOUR(1/2,1/3,1/4)*

COS(X) De cosinus van X.

LEN(A$) Het aantal tekens (de lengte) van A$.

LOG(X) De ondergrens van dimensie N van een reeks A.

LOG1Ø(X) De logaritme van X met grondtal 1Ø.

LOG2(X) Logaritme van X met grondtal 2.

LTRIM$(A$) Verwijdert ale spaties die zich aan het begin van string A$ bevinden. *LTRIM$(" Hallo")* wordt dus "Hallo".

MAGENTA De kleur 'magenta', gelijk aan COLOUR (1,Ø,1).

MAX(X,Y) Retourneert het grootste getal van X en Y. *MAX(6,99)* is dus 99.

MAXLEN(A$) Geeft de maximale lengte, zoals gespecificeerd voor een string-variabele of reeks.

MIN(X,Y) Als MAX(X,y),maar nu wordt het kleinste getal geretourneerd.

MOD(X,Y) X modulo Y. Of eenvoudiger uitgedrukt, de rest van X gedeeld door Y.

ORD(A$) Geeft de ASCII-code van het teken tussen aanhalingstekens, of de ASCII-code van het eerste teken van een string-variabele. ORD staat voor 'ordinal' (rangtelwoord) en betekent het getal dat in verband gebracht wordt met het teken, dat in de tekenset door de computer gebruikt wordt. Omdat de Enterprise gebruik maakt van ASCII, wordt de ASCII waarde geretourneerd.

PEEK(N) Retourneert de byte op Z80 adres N.

PI Het zogenaamde getal pi. Op de Enterprise afgerond op 3,141592654. Retourneert de waarde.

POS(A$,B$,M) Alternatief voor POS. Zie hierondel.

POS(X$,Y$) Geeft de positie in X$ (de tekens van links naar rechts tellend) waar Y$ voor het eerst voorkomt. Als Y$ niet gevonden kan worden in X$, is het resultaat Ø. Door een getal toe te voegen achter de tweede string (bij

voorbeeld POS(X$,Y$,X)) kunt u de machine laten weten, dat ze vanaf een speciale plaats in X$ naar Y$ moet gaan zoeken. Als X$ "LONDON" is en Y$ "ON", dan is POS(X$,Y$) 2. Mar in geval van POS(X$,Y$,4) zou de computer vanaf de "D" in "LONDON" naar "ON" beginnen te zoeken, met als uitkomst 5.

RAD(X) Converteert X van graden naar radiarlen. RAD(X)=X*PI/18Ø.

RED De kleur 'rood;, gelijk aan COLOUR (1,Ø,Ø).

REM(X,Y) De rest van X, gedeeld door Y. N.B.: REM(-1,3) = -1. Zie MOD(X,Y).

RGB Retourneert het machine-afhankelijke kleurnummer dat gelijk is aan de opgegeven combinatie van rood, groen en blauw. R specificeert de hoeveelheid rood (Ø tot 1), G de hoeveelheid groen (Ø tot 1) en B de hoeveelheid blauw (Ø tot 1).

RND Brengt een willekeurig getal voort tussen Ø en 1. Om praktische redenen worden willekeurige getallen vermenigvuldigd om er grotere van te maken. *INT(RND 1ØØ)* geeft een willekeurig geheel getal tussen Ø en 99 (inclusief 99). (RND is nooit 1.)

RND(X) Genereert een willekeurig geheel getal, kleiner dan X. De maximum toegestane waarde voor X is 32767.

ROUND(X,N) X afronden op N decimalen. *ROUND(1.7668,2)* is 1.77; *ROUND(-1.7668,2)* is -1.76.

RTRIM$(A$) Haalt spaties weg aan het einde van de strong. Als LTRIM$, maar nu worden de spaties rechts verwijderd.

SEC(X) De secans van X.

SGN(X) Retourneert het teken van X. Retourneert -1 als X negatief is, 1 als X Ø is, en 1 als X groter dan o is.

SIN(X) De sinus van X.

SINH(X) De sinus hyperbolicus van X.

SIZE(A) Het aantal elementen in de reeks A.

SIZE(A,N) Het aantal elementen dat toegestaan is in dimensie N van de reeks.

SPEEK(S,N) Als PEEK, maar retourneert de byte op systeemadres N in segment S.

SQR(X) De vierkantswortel uit X. X moet positief zijn.

STR$(X) Zet de waarde X om in een numerieke string zonder voorafgaande of achteraankomende spaties, maar met een '-' teken als X negatief is.

TAB(X) Alleen toegestaan in PRINT opdrachten. Verplaatst de positiewijzer naar kolom X op de actuele rij.

TAN(X) De tangens van X.

TANH(X) De tangens hyperbolicus van X.

TIME$ Geeft de tijd in standaardformaat (UU:MM:SS). Zie **TIME** commando.

TRUNCATE(X,N) Kapt N decimalen af van X.

UBOUND(A) De bovengrens van de dimensie van een ééendimensionale reeks A.

UBOUND(A,N) Bovengrens van dimensie N van reeks A.

UCASE$(A$) Verandert alle letters in string A$ in hoofdletters.

USR(N,X) Roept een adres N aan (dat waarschijnlijk met behulp van CODE is gedefinieerd) en overhandigt het gehele getal X in HL aan de machinecode routine. De waarde die in HL achterblijft wordt door USR geretourneerd.

VAL(A$) Zet een string om in een getal (het tegenovergesteld van STR$). VAL begint met het omzettingsproces bij het eerste cijfer in de string en stopt ermee, zodra het eerste niet-numerieke teken aangetroffen wordt.

WHITE De kleur 'wit', gelijk aan COLOUR (1,1,1).

WORD$(N)

Retourneert een twee-byte string die de hoog- en de laagwaarde byte van N bevat — n wordt verondersteld een adres te zijn. N zal meestal een adres zijn dat gedefinieerd is door de CODE opdracht; sprongen achteruit end, met behulp van labels zijn toegestaan. De eerste byte van de string zal de minst significante bit zijn.

YELLOW

De kleur 'geel', gelijk aan COLOUR (1,1,Ø).

EXOS is de afkorting van Enterprise eXpandable Operating System (het besturingssysteem van de Enterprise met uitbreidingsmogelijkheden). Een besturingssysteem is een programma dat de bediening van een computer met voorzieningen zo soepel mogelijk probeert te laten zijn. Het besturingssysteem vormt een verbinding tussen programma's in een hogere programmeertaal (zoals bij voorbeeld het ingebouwde BASIC) en de computer.

De belangrijkste voorzieningen van de computer zijn de verschillende (rand-)apparaten. Hiertoe behoren bij voorbeeld het beeldscherm, de magneetband-koppeling en de printer. Het besturingssyteem houdt zich hoofdzakelijk bezig met deze apparaten: het invoer/uitvoer systeem. Tot de overige voorzieningen waarvoor het besturingssysteem verantwoordelijk is, behoort o.a. de verdeling van geheugenruimte.

INVOER/UITVOER

De Enterprise microcomputer is uiterst complex; zelfs eenvoudige functies, zoals bij voorbeeld het afdrukken van een string op het scherm, vereisen duizenden instructies op machine niveau. En wilt u dezelfde string op een printer zetten, dan komen er nog eens een paar honderd instructies bij. Het besturingssysteem van de Enterprise zorgt voor een efficienter verbinding tussen een programma en de microcomputer, waardoor een string net zo gemakkelijk op een printer als op het scherm gezet kan worden. Dit wordt bereikt door programma's alle invoer en uitvoerapparaten op dezelfde manier te laten behandelen. Alle in-en uitvoer wordt geregeld via 'kanalen' (het enige dat een kanaal bewerkstelligt, is het programma met een apparaat verbinden). De kanalen zijn van Ø tot 254 genummerd. Het besturingssysteem voorziet in de volgende functies op kanalen:

Code-Nummer	Functie
Ø	systeem opnieuw ingesteld
1	een kanaal openen (een apparaat aansluiten)
2	een kanaal creeren en openen
3	een kanaal sluiten (loskoppelen)
4	een kanaal sluiten en laten vervallen
5	een teken van een kanaal lezen

6	een blok lezen
7	een teken naar een kanaal schrijven
8	een blok schrijven
9	retourneer de toestand van het kanaal
10	de toestand van het kanaal instellen en die lezen
11	een speciale functie uitvoeren
16	een systeemvariabele lezen, schrijven of schakelen
17	invoer van kanaal tot kanaal vangen
18	een kanaal een nieuwe richting geven
19	de naam instellen van het systeemgekozen apparaat
20	de toestand van het systeem retourneren
21	koppelen apparaat
22	de systeemgrens lezen
23	de gebruikersgrens instellen
24	een segment toewijzen
25	een segment vrijmaken
26	de plaats van Read Only geheugens nagaan
27	een kanaalbuffer toewijzen.
28	retourneer foutmelding
29	laad module
30	laad verplaatsbare module
31	set tijd
32	lees tijd
33	set datum
34	lees datum

Om in invoer/uitvoer faciliteiten te voorzien worden deze functies benut door het BASIC. Ze kunnen met alle talen gebruikt worden en bieden zo een uniforme methode om met randapparaten te communiceren. Ze staan eveneens ter beschikking van de machinecode programmeur, voor wie het nu een makkie wordt om programma's in machinecode te schrijven.

Wanneer u het besturingssysteem vanuit een programma in machinecode wilt aanroepen, is een enkele instructie vereist gevolgd door de code van de functie. Voor het openen van een kanaal is bij voorbeeld de volgende code nodig:

Machinecode	Assembleercode
F7	RST 3ØH
Ø1	D9 1

Het besturingssysteem van de Enterprise biedt veel meer functies dan de hierboven opgesomde. Een volledig overzicht van de functies en de regels voor het aanroepen vindt u terug in de Technische Handleiding bij de Enterprise.

GEBRUIK VAN HET GEHEUGEN

Het besturingssysteem heeft als basis het Read Only geheugen. Dit houdt in dat het programma in het ROM opgeslagen wordt, maar niettemin RAM ruimte nodig heeft voor de opslag van zijn gegevens. De Enterprise kan een enorme hoeveelheid RAM en ROM aan.

Met de geheugens kan gewerkt worden door deze in 'pagina's' te verdelen (niet te verwarren met video-pagina's); elke pagina is 16K bytes lang. In totaliteit zijn er 256 pagina's met een maximale opslagcapaciteit van 4M bytes. De Z80 in de Enterprise kan slechts vier van deze pagina's tegelijkertijd gebruiken (tot op zekere hoogte met het lezen van een boek te vergelijken, waarbij u slechts twee pagina's tegelijk kunt zien en lezen).

Af en toe kunt u niet verhinderen dat u een fout maakt in een programma. Het kan een probleem zijn te achterhalen waar de fout zit — of zelfs wat er fout is gegaan.

Hier krijgt u hulp van de computer die u via boodschappen zo goed mogelijk informeert over wat er fout is gegaan. Wanneer u een programma draait dat een BASIC fout bevat, zal de computer daar stoppen waar hij het programma niet langer begrijpt, en een korte instructie in beeld brengen met informatie over de oorzaak van het probleem.

Bedenk wel dat de computer u niet op dezelfde manier kan informeren over fouten van een ander type. Wanneer u bij voorbeeld 'prioriteit van bewerkingstekens' over het hoofd hebt gezien of wanneer u een andere uitkomst verwachtte dan de werkelijke uitkomst, zal het programma toch tot het einde toe gedraaid worden — alleen zal het programma dan iets anders uitvoeren dan u had verwacht. De computer kan alleen fouten ontdekken in de syntaxis of organisatie van uw BASIC, of problemen die veroorzaakt zijn, omdat het programma om een onmogelijke actie heeft gevraagd.

De boodschap die door de computer in beeld wordt gebracht, wanneer u iets hebt getikt dat niet begrepen wordt, bevat een codenummer dat naar de categorie verwijst van uw fout (deze categorieen komen hierna aan de orde).
Voorbeeld:

```
*** Not understood (Niet begrepen)
```

Als het programma al draait, wanneer een probleem zich voordoet dat verdergaan onmogelijk maakt, zal de foutboodschap het relevante regelnummer bevatten; bij voorbeeld:

```
*** Invalid argument to SQR
300 PRINT SQR (Y)
```

Nu kunt u de positiewijzer verplaatsen en regel 300 wijzigen.

De functies EXLINE, EXTYPE en EXSTRING$ dienen om fouten en andere uitzonderingssituaties te helpen oplossen. Elke fout heeft zijn eigen nummer, dat

met EXTYPE kan worden opgeroepen. Bij de meeste fouten wordt een speciale melding afgedrukt, als deze niet wordt onderdrukt. Doet zich een uitzonderingssituatie voor, waarvoor in de standaard meldingen niet is voorzien, dan wordt het algemene foutnummer afgedrukt, samen met het nummer van de uitzondering. Bijvoorbeeld

```
* Overloop fouttype 1234
```

Hieronder volgen de algemene typen foutmeldingen.

```
    Ø —   999 Gebruiker
 1ØØØ —  1999 Overloop
 2ØØØ —  2999 Subscript
 3ØØØ —  3999 Wiskundig
 4ØØØ —  4999 Parameter
 5ØØØ —  5999 Geheugen vol
 6ØØØ —  6999 Matrix
 7ØØØ —  7999 Bestandsgebruik
 8ØØØ —  8999 Invoer/uitvoer
 9ØØØ —  9999 Exos
1ØØØØ — 10999 Besturing
11ØØØ — 11999 Grafisch
12ØØØ — 12999 Real-time
2ØØØØ — 2Ø99 Syntaxis
3ØØØØ —      Systeem
```

De specifieke foutmeldingen zijn:

```
1ØØØ — Unexpected value given (onverwachte waarde
       gegeven)
1ØØ1 — Overflow in numeric constant (overloop bij
       numerieke constante)
1ØØ2 — Overflow in numeric expression (overloop bij
       numerieke expressie)
1Ø51 — Overflow in string expression (overloop bij
       string-uitdrukking)
11Ø6 — Overflow in string assignment (string te lang)

2ØØ1 — Array subscript out of bounds (suffix buiten
       grenzen)
3ØØ1 — Division by zero (deling door nul)
3ØØ4 — Invalid argument to LOG (ongeldig argument
       bij LOG)
```

3005 — Invalid argument to SQR (ongeldig argument bij SQR)
3007 — Invalid argument to ASIN or ACOS (ongeldig argument bij ASIN of ACOS)

4000 — Error in DEF parameters (fout in de berekening van DEF parameters)
4002 — Argument to CHR$ out of range (argument van CHR$(X) buiten bereik)
4003 — Invalid argument to ORD (argument van ORD ongeldig)
4004 — Index to SIZE out of range (index van SIZE buiten bereik)
4005 — Argument to TAB out of range (argument van TAB buiten bereik)
4008 — Index to LBOUND out of range (index van LBOUND(X) buiten bereik)
4009 — Index to UBOUND out of range (index van UBOUND(X) buiten bereik)
4301 — Error in CHAIN parameters (fout in berekening van CHAIN parameters)

5000 — Insufficient memory (onvoldoende geheugen)
5100 — Insufficient stack space (onvoldoende ruimte)
5110 — Insufficient extension space (onvoldoende uitbreidingsruimte)
5120 — Insufficient ALLOCATE space (onvoldoende ALLOCATE ruimte)
7001 — Invalid channel no. (ongeldig kanaalnummer)
7003 — Channel already open (OPEN opdracht terwijl kanaal al open is)
7004 — Channel not open (kanaal niet open)
7401 — TRACE channel not open (TRACE kanaal niet open)

8001 — Out of date in READ/INPUT (READ/INPUT)
8101 — Numeric data expected (numerieke gegevens verwacht)
8201 — Invalid USING string (ongeldige USING string)
8202 — No format item in USING string (geen opmaakbestanddeel in USING string)
8203 — USING format item too short (opmaakbestanddeel in opmaakstring te kort voor uitvoer)

9208 — Cassette CRC error (cassette CRC fout)
9209 — Editor — load file too big (geladen bestand te groot)
9210 — Editor — keyboard channel error (verkeerd kanaal voor toetsenbord)
9211 — Editor — keyboard channel error (verkeerd kanaal voor toetsenbord)
9212 — Editor — video channel error (verkeerd kanaal voor vide)
9213 — Network link already exists (netwerkverbinding bestaat al)
9214 — Network address not set (netwerkverbinding bestaat al)
9215 — Cannot use both serial and network (seriele verwerking en netwerk kunnen niet tegelijk worden gebruikt)
9216 — Invalid beam position (ongeldige positie videokanaal)
9217 — Invalid cursor coordinates (ongeldige cursor-coordinaten)
9218 — Invalid row number to scroll (ongeldig rijnummer voor verschuiving omhoog of omlaag)
9219 — Invalid video page file (ongeldig bestand videopagina)
9220 — Invalid display parameters (ongeldige weergave-parameters)
9221 — Invalid video mode (ongeldige videomodus)
9222 — Invalid video page size (ongeldige afmeting videopagina)
9223 — Sound queue full (wachtrij signaalgenerator vol)
9224 — Envelope storage full (opslagruimte voor envelop benut)
9225 — Envelope too big (envelop te groot)
9226 — Function key string too long (functietoets-string te lang)
9227 — Protection violation (inbreuk op systeembescherming)
9228 — Unexpected end of file (onverwacht eind bestand)
9229 — STOP key pressed (STOP-toets ingedrukt)
9230 — Invalid escape sequence (ongeldig 'esc'-sequentie)

9231 — Call not supported by this device (oproep niet ondersteund door dit apparaat)
9232 — Invalid unit number (ongeldig nummer eenheid)
9233 — Device already in use (apparaat al in gebruik)
9234 — Invalid special function call (ongeldige oproep speciale functie)
9235 — Invalid date or time value (ongeldige waarde voor datum of tijd)
9236 — End of file module (einde bestand moduul)
9237 — Unknown module type (onbekend moduultype)
9239 — Invalid Enterprise file header (ongeldige koptekst bestand op Enterprise)
9240 — Unrecognised command string (commandostring niet herkend)
9241 — Invalid device descriptor (ongeldige beschrijving apparaat)
9242 — Unknown EXOS variable number (onbekend EXOS variabel getal)
9243 — Invalid user boundary (ongeldige gebruikersgrens)
9244 — Cannot free segment (segment kan niet vrijgemaakt worden)
9245 — No free segment (geen vrij segment)
9246 — Insufficient video memory (onvoldoende geheugen voor video)
9247 — Insufficient memory (onvoldoende geheugen)
9248 — Channel open error (kanaal open fout)
9249 — Channel already exists (kanaal bestaat reeds)
9250 — Device does not exist (apparaat bestaat niet)
9251 — Channel does not exist (kanaal bestaat niet)
9252 — EXOS stack overflow (overloop EXOS stapel)
9253 — Invalid EXOS string (ongeldige EXOS string)
9254 — EXOS function call not allowed (oproep EXOS functie niet toegestaan)
9255 — Invalid EXOS function code (ongeldige code EXOS functie)

10002 — Return with out GOSUB (Return' zonder GOSUB)
10004 — No CASE selected (geen CASE gekozen)
10005 — Program does not exist (programma bestaat niet)

20000 — Not understood (niet begrepen)
20001 — Invalid line number (ongeldig aantal regelnummers)
20002 — Invalid line number range (ongeldig aantal regelnummers)
20004 — Line number does not exist (regelnummer bestaat niet)
20010 — Cannot do specified RENUMBER (gespecificeerde RENUMBER kan niet uitgevoerd worden)
20020 — Continue not possible (verdergaan niet mogelijk)
20030 — Identifier expected (identificatiesymbool verwacht)
20031 — String identifier expected (identificatiesymbool voor string verwacht)
20032 — Array identifier expected (identificatiesymbool voor array verwacht)
20034 — Type mismatch (tik fout)
20040 — Variable not initialised (variabele niet geinitialiseerd)
20041 — Identifier declared twice (identificatiesymbool twee keer benoemd)
20042 — Identifier too long (identificatiesymbool te lang)
20043 — Missing closing quotes (ontbrekende aanhalingstekens sluiten)
20050 — Missing end of block (einde van blok ontbreekt)
20051 — Invalid end of block (einde van blok ongeldig)
20052 — Too many nested blocks (teveel geneste blokken)
20060 — Invalid option use (ongeldig gebruik van machine-optie)
20071 — Statement in immediate mode (opdracht in rechstreekse modus)
20072 — Command in program (commando in programma)
20073 — Statement not allowed after THEN (opdracht niet toegestaan na THEN)
20074 — Invalid multi-statement line (ongeldige meeropdrachtregel)
20075 — Line too long (regel te lang)
20080 — Invalid file format (ongeldige indeling bestand)

20081 — Programs do not VERIFY (programma's verifieren niet)

30000 — BASIC data has been corrupted (BASIC gegevens zijn beschadigd)

Foutmeldingen kunnen onder bepaalde omstandigheden opgevangen worden door gebruik te maken van WHEN EXCEPTION en een functie voor de afhandeling van uitzonderingen. Een functie voor de afhandeling van uitzonderingen is weliswaar handig, maar kan niet voorkommen dat er syntacticale fouten gemaakt worden (bij voorbeeld een verkeerd gespeld sleutelwoord) of dat het geheugen overloopt. Dingen als delen door nul of een negatief SQR argument kunnen opgevangen worden zonder het programma te 'crashen'.

Deze woordenlijst zal u helpen vertrouwd te raken met het computerjargon, waarmee u te maken krijgt als uw belangstelling voor computers groter wordt. De meeste woorden bent u al ergens in de handleiding tegengekomen, een aantal andere niet — voorzover mogelijk hebben we in de handleiding jargon proberen te vermijden en in plaats daarvan tekst en uitleg gegeven. Deze woordenlijst zal van pas komen wanneer u boeken of tijdschriften over computers leest waarin geen woordenlijst is opgenomen, maar die wel bol staan van moeilijke termen.

AANEENSCHAKELEN
CONCATENATION

Twee strings samenvoegen met gebruikmaking van het & teken. A$&B$ is een lange string die bestaat uit de inhoud van A$ en die van B$.

AANROEPEN
CALL

Een sprong binnen een programma naar een een subprogramma of een subroutine. Door gebruik te maken van het sleutelwoord CALL (aanroepen) kunnen in BASIC programma's op de Enterprise functies benut worden die door de gebruiker zijn gedefinieerd.

ADRES
ADDRESS

Een 'plek' in het geheugen van een computer. Deze positie wordt nader aangeduid met een getal, hetzij in decimale notatie (grondtal 1Ø) hetzij in hexadecimale notatie (grondtal 16) of in binaire notatie (grondtal 2). Afhankelijk van datgene dat de computer uitvoert of van dat wat het programma met het adres doet kan een adres een van meerdere getallen bevatten. De adressen bij voorbeeld die het beeldscherm besturen, bevatten afhankelijk van wat in beeld gebracht moet worden steeds andere getallen. We kennen ook het werkwoord adresseren: de inhoud van een geheugenpositie onderzoeken.

ALFANUMERIEK
ALPHANUMERIC

Letters of cijfers. De benaming voor de tekenset, exclusief speciale of grafische symbolen.

ALGORITME
ALGORITHM

De gedachtes achter een programma en de taken die er deel van uitmaken. Eerst moet u uw algoritme uitwerken en daarna pas uw programma schrijven. Een algoritme is de weg waarlangs een probleem opgelost wordt.

ANSI

American National Standards Institute. De Amerikaanse

tegenhanger van het British Standards Institute. Een gemeenschappelijke commissie van de ANSI en de European Computer Manufacturers' Association gaf nader inhoud aan het Standaard BASIC.

APPARAAT
DEVICE

Een toestel, bij voorbeeld de TV of een cassette-recorder, dat op de computer aangesloten wordt en door de computer tot op zekere hoogte bestuurd. De signaalgenerator en de grafische gegevensverwerking worden beide door de Enterprise als apparaten gezien. Elk randapparaat dat op de Enterprise aangesloten is, wordt gezien als een extra apparaat dat met behulp van BASIC via het besturingssysteem bediend wordt.

ARGUMENT

Zie OPERAND en PARAMETER.

ASCII

American Standard Code for Information Interchange. (Amerikaanse nationale standaardcode voor informatie-uitwisseling). Een systeem van codetekens voor gebruik in de computer en voor transmissie van gegevens van de computer naar andere machines en terug. Elk teken heeft een nummer.

ASSEMBLEERTAAL
ASSEMBLER
LANGUAGE

Een programmeertaal die op haar eigen voorwaarden met de computer praat. Een assembleertaal kent eenvoudiger instructies dan BASIC en biedt u de mogelijkheid uw computer veel gedetailleerder te programmeren dan met BASIC mogelijk is — het gebruik van een assembleertaal is niet zozeer moeilijk, als wel vervelend. Wanneer u programmeert in assembleertaal, programmeert u op een laag niveau. U maakt dan gebruik van kleine, uiterst eenvoudige sleutelwoorden en — meer dan in BASIC het geval is — past u codes toe als de ASCII-code.

BASIC

De programmeertaal die bij de Enterprise (en de meeste andere kleine computers) geleverd wordt. Er worden verschillende soorten BASIC-dialecten onderscheiden. Het woord BASIC is een afkorting van Beginners' All-purpose Symbolic Instruction Code. Oorspronkelijk ontworpen op kolossale computers als steun bij het leren programmeren.

BAUD

De snelheid waarmee gegevens van en naar de computer gestuurd kunnen worden. In grote lijnen komt

baud overeen met 'bits per seconde'. Doorgaans laadt de Enterprise een programma vanaf cassette met een snelheid van 2400 baud.

BENOEMEN
DECLARATION

De computer informeren dat u een speciale variabele of reeks gaat gebruiken.

BESTAND
FILE

Een georganiseerde verzameling gegevens. Een bestand kan ofwel een programmabestand zijn ofwel een gegevensbestand waarvan een programma gebruik zal maken. Opgeslagen op schijf of cassette (of uitgelijst op een printer). Een bestand wordt in records verdeeld.

BESTURINGSCODE
CONTROL CODE

Een teken dat niet zichtbaar is op het scherm, maar in plaats daarvan een of andere actie teweegbrengt van de computer. Voorbeelden daarvan zijn CHR$(8), wat staat voor 'een spatie teruggaan', en CHR$(13), wat staat voor 'wagenterugloop'.

BESTURINGSSYSTEEM
OPERATING SYSTEM

Een programma dat alle in- en uitvoer regelt. Meestal gebruikt in combinatie met schijfsystemen om de schijf zelf en de organisatie van de gegevens op de schijf te besturen. CP/M is een besturingssyteem, waarmee alle aspecten van de apparatuur van de Enterprise en eventueel aangesloten apparaten bestuurd kunnen worden.

BEWERKINGSTEKEN
OPERATOR

Een wiskundig begrip waarmee tekens aangeduid worden die voor bewerkingen staan, zoals bij voorbeeld *,/,+ en - die respectievelijk staan voor vermenigvuldigen, delen, optellen en aftrekken. Dit zijn aalemaal wiskundige bewerkingen.

BINAIR
BINARY

2 Als grondtal in plaats van 10. Daarom zijn binaire getallen uitsluitend uit enen en nullen samengesteld. Zo is negen 1001. Zo goed als we bij gewone getallen eenheden onderscheiden (tientallen, honderdtallen, etc.) net zo onderscheiden we eenheden in het binaire talstelsel (tween, vieren, achten, etc.). De computer denkt binair.

BIT

Een binair cijfer (0 of 1). De kleinste eenheid informatie die de computer kan herkennen. Kan ook weergegeven worden als hoog/laag voltage (zoals in de computer) of

als positieve/negatieve magnetische impulsen, zoals op cassette.

BYTE

Acht bits. Wordt beschouwd als een eenheid in de computer. Het geheugen wordt gemeten in bytes of in kilobytes (1Ø24 bytes, ook wel K genoemd) of in megabytes (1Ø24K bytes). Normaal is er een byte nodig om een teken op te slaan.

CARTRIDGE

Een cassette met een of twee chips die een programma bevatten. De cartridge kan in de gleuf aan de zijkant van de Enterprise gestoken worden en levert zo een programma dat meteen gedraaid wordt zonder laden of tikken. Video-bedieningspanelen maken gebruik van cartridges voor spelletjes. U hoeft alleen maar de cartridge te verwisselen wanneer u een ander spel wilt.

CHIP

Een klein plaatje met microscopische schakelingen in de computer. De chip zelf is ongeveer 2 mm2 (de afmetingen kunnen varieren) en is gemaakt van silicium, ook wel bekend als een halfgeleider of metalloide. De uiterst verfijnde schakelingen van de chip worden beschermd door een celluloid laagje; de chip is voorzien van metalen pinnen om contact te maken met de andere chips in de computer. Siliciumchips stan ook bekend als IC's.

COAXKABEL
COAXIAL CABLE

De kabel die de computer met de TV verbindt. Technisch gezien wordt de kabelkern volledig omgeven door nog een geleidende laag, teneinde interferentie te voorkomen.

CODE

Zie CONTROL CODE. 'Code' is ook een omschrijving van de tekst van een programma (broncode) of van het programma, zoals uitvoerd door de computer (werkcode). Zie ook MACHINECODE.

CODEREN
CODING

De technische term voor het schrijven van programmaregels; tegenstelling: het ontwerpen van programma's.

COMMANDO
COMMAND

Een sleutelwoord, zoals bij voorbeeld GOTO, CALL, etc., dat de computer directe instructies geeft. Het eerste sleutelwoord op een programmaregel is meestal een commando.

COMPILEER-PROGRAMMA

Een speciaal programma dat van de ene computertaal naar een andere vertaalt — doorgaans van een broncode in een hogere programmeertaal naar machinecode. U kunt niet direct veranderingen aanbrengen in een programma dat gebruik maakt van een compileerprogramma — hetgeen wel het geval is met een VERTOLKPROGRAMMA — maar in het algemeen wordt het programma sneller uitgevoerd als het gecompileerd is.

CONSTANTE
CONSTANT

Een grootheid die niet in waarde verandert. Het woord PI bij voorbeeld staat voor een constante (3,14...) die gebruikt wordt bij berekeningen met cirkels. 2 is een constante; "A" is een constante. Het komt wel eens voor dat een 'variabele' het hele programma door in feite gebruikt wordt als een constante.

COORDINAAT
CO-ORDINATE

Een getal dat de positie van iets nader aangeeft — met name de positie op het scherm. PRINT AT maakt gebruik van twee coordinaten — een vertical en een horizontale coordinaatom aan te geven waar een teken afgedrukt moet worden. PLOT maakt eveneens gebruik van coordinaten; zo wordt de positie van puntjes aangegeven of het begin of het einde van een grafische lijn.

CRASH

Een noodstop die veroorzaakt kan worden door tal van dingen.

CTRL

Afkorting van 'control' (besturing). Een toets die gebruikt wordt om direct vanaf het toetsenbord besturingscodes te genereren.

CVE
CPU

Cenrale verwerkingseenheid (Central Processing Unit). Een grote chip in de computer die alle andere chips bestuurt.

De assembleertaal is geschreven om rechtstreeks met de CVE te kunnen 'praten'; een assembleertaal verschilt naargelange het type CVE. De Enterprise benut een Z80 CVE.

DATA

(1) Een BASIC opdracht waarmee u de computer laat weten dat u constanten — woorden of getallen — in een programma gaat gebruiken; deze worden benut in combinatie met het READ commando.

(2) Gegevens. Letters of cijfers waarvan een programma gebruik makt om zinvolle informatie samen te stellen of een taak uit te voeren. De cijfers die u gebruikt voor de definiering van uw eigen tekens, zijn gegevens.

DECIMAAL
DECIMAL

Het stelsel met als grondtal 1Ø, waaraan we allemaal gewend zijn.

DEFINIEREN
DEFINE

De details van iets specificeren, met name voor later gebruik. Meestal van toepassing op een door u ontworpen teken of op een functie.

DIMENSIES
DIMENSIONS

Worden toegepast wanneer reeksen beschreven worden. ARRAY(X) is een eendimensionale reeks, ARRAY(X,Y) is een tweedimensionale reeks.

DISPLAY

Weergave. Alles dat op het TV scherm verschijnt.

DOORGEVEN
PASS

De verplaatsing van variabelen of waarden van het ene naar het andere programma, of van het ene naar het andere gedeelte van hetzelfde programma.

EENHEID

Ofwel een afzonderlijk onderdeel — een teken is een gegevenseenheid — ofwel een apparaat. De computer is een eenheid, een schijfaandrijving eveneens.

ELEMENT

Een onderdeel van een reeks. Het aantal elementen in een reeks geeft u zelf aan wanneer u de reeks benoemt.

ESC

Afkorting van 'escape' (wisselen). Wordt ofwel gebruikt om naar een hoger programmaniveau te gaan, ofwel om aan te geven dat de volgende codes een speciale betekenis hebben.

EXOS

Enterprise Expandable Operating System (besturingssysteem van de Enterprise met uitbreidingsmogelijkheden). Dit is een besturingsprogramma in de Enterprise dat gebruikt wordt door het BASIC en dat de communicatie van de computer met andere apparaten regelt.

EXPONENT

De macht waarmee een grondtal wordt verheven. 1Ø^3

is 1Ø*1Ø is 1Ø tot de macht 3; 3 is de exponent. Zie MACHTSVERHEFFING.

FIRMWARE

Een programma dat altijd in de computer zit. De Exos van de Enterprise is firmware. De BASIC voor de Enterprise bevindt zich op een uitneembare cartridge. Op andere thuiscomputer kan dit programma, ook wel INTERPRETER (vertolker) genaamd, firmware zijn.

FOUT
BUG

Een fout in een programma. Soms gemakkelijk te achterhalen — vooral wanneer het bij voorbeeld om een foutief gespeld BASIC woord gaat. En soms zijn vlooien niet zo gemakkelijk terug te vinden — althans de gevolgen komt u al snel tegen, maar de oorzaken in het programma zijn slechts met moeite te vinden.

FOUTEN OPSPOREN
DEBUG

Naar fouten in een programma zoeken en deze vervolgens corrigeren. Dit is een cruciaal stadium in de ontwikkeling van omvangrijke of complexe programma's. Fouten zijn niet altijd even opvallend en elk programma moet zorgvuldig getest worden op allerlei mogelijkheden.

FUNCTIE
FUNCTION

Een gedeelte van een programma dat een specifieke taak of berekening uitvoert. Een functie wordt beschouwd als een 'programma in een programma'. De computer maakt er alleen volgens instructie gebruik van. Een functie kan ofwel vooraf gedefinieerd zijn en deel uitmaken van het BASIC (bij voorbeeld INT, CHR$, LOG), ofwel nader ingevuld worden door u. De acht speciale programmeerbare toetsen op de Enterprise worden functietoetsen genoemd, omdat de functie ervan opnieuw gedefinieerd kan worden.

GANG
PASS

Een uitvoeringscyclus van een lus.

GEBRUIKER
USER

De persoon die een computer bedient of van een programma gebruik maakt. De term 'gebruikersvriendelijk' verwijst naar het gemak waarmee sommige programma's benut kunnen worden.

GEGEVENSBANK
DATABASE

Opgeslagen gegevens. Vaak ook gebruikt als korte omschrijving van een programma dat uitsluitend

ontworpen is om gegevens op te slaan en te manipuleren — bij voorbeeld postadressen. Gegevensbanken lopen qua complexiteit uiteen van heel eenvoudige — de mogelijkheid bieden een volledig computergestuurd opbergsysteem te ontwerpen en/of sorteer- of extractiebewerkingen uit te voeren.

GERESERVEERD WOORD
RESERVED WORD

Een BASIC woord dat in het BASIC niet voor andere doeleinden gebruikt kan worden. De meeste BASIC woorden kunnen aangewend worden als benaming voor variabelen; de BASIC woorden waarvoor dit niet geldt, zijn gereserveerde woorden.

GETALLEN KRAKEN
NUMBER CRUNCHING

Uiterst snel en ingewikkelde berekeningen uitvoeren met getallen. Sommige computer zijn speciaal ontworpen voor het kraken van getallen — ze zijn snel en bedreven in het rap uitvoeren van hele reeksen berekeningen.

GRAFISCHE GEGEVENS-VERWERKING
GRAPHICS

Het vermogen van een computer om tekeningen te maken of lijnen en speciale symbolen te gebruiken voor de opmaak van informatie.

HARDWARE

Apparatuur; het materiele gedeelte van een informatieverwerkend systeem. Zowel apparatuur die van de computer zelf deel uitmaakt als losse apparaten die met de computer verbonden zijn.

HEXADECIMAAL
HEXADECIMAL

Een talstelsel dat 16 in plaats van 1Ø als grondtal heeft. Hex (afkorting) maakt gebruik van de cijfers Ø-9 en de letters A-F (F is 15). Hexadecimale getallen worden soms voorafgegaan door & of gevolgd door een H om het hexadecimale karakter aan te geven (2ðH is 38 decimaal). hex is een handige manier om binaire getallen weer te geven.

HOOFDCOMPUTER
MAINFRAME

Een echt grote computer. Sommige hiervan zijn zo groot dat ze meerdere vertrekken beslaan.

IC

Integrated Circuit. Een andere benaming voor siliciumchip.

IDENTIFICATIE-SYMBOOL
IDENTIFIER

Een naam waarmee een variabele, een functie, een apparaat of ieder ander gedeelte van de computer of van het progamma geïdentificeerd kan worden. Hoe duidelijker de identificatiesymbolen des te gemakkelijker zijn de programma's te begrijpen.

INFORMATIE
INFORMATION

Informatie wordt soms gezien als de uitkomst van gegevensverwerking, d.w.z. als de resultaten van sorteren en het doorzoeken van bestanden en/of rubrieken. Gegevens die op zo'n manier bij elkaar zijn gebracht dat er betekenis aan toegekend kan worden, noemen we informatie. Onder informatie verstaan we ook alles waarmee het programma werkt, zoals bij voorbeeld getallen, strings, variabelen.

INTELLIGENTE SOFTWARE
INTELLIGENT SOFTWARE

Intelligente programmatuur. Intelligent Software Ltd. is de geestelijke vader van de Enterprise.

INVOER
INPUT

Alle gegevens of elk programma dat de computer ingaat, ofwel vanaf een geheugeneenheid (schijf, etc.) ofwel via het toetsenbord.

I/O

Invoer/Uitvoer (Input/Output). Hiermee wordt naar elk gedeelte van de computer verwezen dat te maken heeft met een gegevensstroom van en naar de computer.

KANAAL
CHANNEL

Een 'route' waarlangs gegevens de computer kunnen binnenkomen of verlaten, of waarlangs ze van het ene naar het andere gedeelte van de computer kunnen gaan. Doorgaans kan de toepassing van een kanaal opnieuw gedefinieerd worden door een programma.

KILOBYTE

1Ø24 bytes (een rond binair getal dat het dichtst bij 1ØØØ ligt).

KOPPELING
INTERFACE

Een stuk apparatuur dat een verbinding vormt tussen de computer en iets daarbuiten. De aansluitingen van de TV en de cassette zijn aangesloten op koppelingen in de Enterprise.

LADEN
LOAD

Iets uit een geheugen halen en naar het computer-geheugen brengen.

LEEG
NULL

De waarde nul hebbend. Een lege string bevat geen tekens, een leeg teken heeft een codewaarde van nul.

LEZEN
READ

Iets uit een geheugen halen en in gebruik nemen. Kan synoniem zijn met LADEN, ofschoon LEZEN als BASIC commando een iets andere betekenis heeft.

LOGICA
LOGIC

De wetenschap die zich met de wetten van het logische denken bezighoudt. Computers zijn zo ontworpen dat ze zich aan de wetten van het logische denken houden. Logische sleutelwoorden zijn bij voorbeeld AND en OR.

LUS
LOOP

Herhalingen binnen een programma door terug te gaan naar een regel waar de lus begint. Een gedeelte van een programma waarvan het einde terugleidt naar het begin, tenzij er aan een 'exit' voorwaarde voldaan wordt. Een oneindige lus is een lus zonder 'exit' voorwaarde die de computer doorgaans aanleiding geeft er de brui aan te geven en niets zinvols meer uit te voeren.

MACHINECODE

De echte machinecode is helemaal uit binaire cijfers samengesteld. Zo functioneert de computer in werkelijkheid; wat u gebruikt — BASIC — is een programma dat door de machinecode instructies in de computer uitgevoerd wordt. Assembleertaal is een directe omschrijving van machinecode; met letters en symbolen worden de binaire cijfers weergegeven.

MACHTSVERHEFFING
INVOLUTION

Een getal met een macht verheffen, bij voorbeeld 2^3; Zie EXPONENT.

MEGABYTE

Circa 1 miljoen bytes. Is gelijk aan 1024 kilobytes.

MICROSOFT BASIC

Een algemene omschrijving van BASIC vertolk-programma's die ontworpen zijn door Microsoft Inc., VS. De meeste karakteristieken van het Microsoft BASIC kunnen benut worden als deelverzameling van de eigenschappen van het BASIC op de Enterprise.

MODULAIR
MODULAR

Opgebouwd uit kleinere, onderling samenhangende delen.

NET, NETWERK
NETWORK

Een aantal computers die alle op elkaar aangesloten zijn en met elkaar kunnen communiceren. De

Enterprise maakt gebruik van een systeem dat bekend staat als het 'intelligente net'.

ONDERBREKEN
BREAK

Om een programma halverwege stop te zetten, doorgaans met als gevolg dat RUN ingetikt moet worden om het programma opnieuw te starten, of CONTINUE om het programma te hervatten waar het verlaten werd.

OPDRACHT
STATEMENT

Een op zichzelf staand, zinvol gedeelte van een programma. IF/THEN regels vormen opdrachten. IF A=2 op zich zou niet erg veel zin hebben, maar IF A=2 THEN PRINT "A=2" wel.

OPERAND

Een getal of een variabele waarop een bewerking van toepassing is. In de uitdrukking 6*2 zijn 6 en 2 operands. Soms ook wel ARGUMENTEN genoemd.

OPSLAAN
STORE

Informatie of gegevens voor later gebruik opbergen, ofwel tijdelijk in een RAM ofwel permanent op cassette of op schijf.

OPZOEKEN
SEARCH

Een bestand of een lijst doornemen op de aanwezigheid van iets speciaals.

OVERDRAAG-
BAARHEID
PORTABILITY

De mogelijkheid (heel zelden) om een programma op meer dan een systeem te gebruiken.

PARAMETER

Een variabele die aan een functie is toegewezen of een waarde toegewezen heeft gekregen. In TAN(X) is X de parameter (of het ARGUMENT).

PIEK
SPIKE

Een plotselinge toe-of afname van de netspanning. Kan heel incidenteel problemen met schijfsystemen opleveren en de computer tijdelijk uitschakelen, waardoor lopende programma's verloren gaan (dit komt maar zelden voor).

PIXEL

Een enkele beeldpunt op het TV scherm.

POORT
PORT

Een stekerbuis om de computer (via een koppeling) met een randapparaat te verbinden.

PROCEDURE

Een ander woord voor subprogramma. Wat het BASIC

op de Enterprise betreft, voorzien functies in alle gangbare mogelijkheden van procedures.

PROGRAMMA
PROGRAM

Een aantal geordende instructies die te samen een taak vormen die door de computer uitgevoerd moet worden.

PSG

Acroniem van Programmable Sound Generator (programmeerbare signaalgenerator). Een chip in de Enterprise die signalen naar een luidspreker stuurt en aldus geluid voortbrengt.

RAM

Random Access Memory (direct toegankelijk geheugen). Een geheugen voor de tijdelijke opslag van programma's of gegevens. De inhoud ervan vervluchtigt wanneer u het woord NEW tikt of wanneer de computer uitgeschakeld wordt.

RANDAPPARAAT
PERIPHERAL

Deze term verwijst naar elk apparaat dat via de computer bediend wordt — bij voorbeeld een TV, een cassette, een printer, enz. Een randapparaat wordt weliswaar door de computer gebruikt maar maakt er geen deel van uit.

REAL TIME

Onmiddelijke verwerking. Tijd die in verband staat met de echte wereld en niet met het specifieke functioneren van de computer. Wordt vaak gebruikt om de verwerking van gegevens te beschrijven, zodra ze zijn ingevoerd. INKEY$ is nagenoeg een 'real time' bewerking. Op bepaalde spelletjes is onmiddellijke verwerking van toepassing om de indruk te wekken dat echte dingen heel snel plaatsvinden. Onmiddellijke verwerking is alleen voorhanden op bepaalde computers of in bepaalde programma's.

REEKS
ARRAY

Een variabele die zelf uit meerdere verschillende variabelen bestaat. Kan gezien worden als een lijst (eendimensionaal) of als een raster (tweedimensionaal).

REFERENTIE
REFERENCE

Doorgaans gebruikt wanneer een parameter beschreven wordt die een functie is, maar eigenlijk een variabele bevat en niet de actuele waarde van de variabele.

ROBUUST
ROBUST

Deze beschrijving is van toepassing op een programma of een stukje programmatuur dat heel weinig fouten bevat en prima functioneert, ongeacht wat gebruikers ermee doen.

ROM

Read Only Memory (permanet geheugen). Een geheugen dat harde programmatuur bevat. De inhoud kunt u niet veranderen, maar wel gebruiken of raadplegen. Wanneer u de computer uitschakelt blijft de inhoud van het geheugen onaangetast.

RUBRIEK
FIELD

Een gedeelte van een record in een bestand. Een postadres is bij voorbeeld een record. De naam van de persoon, de straatnaam, etc. zouden dan rubrieken zijn. Dankzij de rubrieken kunnen stukjes informatie veel gemakkelijker gevonden worden.

SCHEIDINGS-
VERMOGEN
RESOLUTION

Het aantal puntjes die beschikbaar zijn voor tekeningen op een scherm. Dit bepaalt de kwaliteit van de grafische gegevensverwerking op een computer. Het scheidingsvermogen kan hoog zijn of uiterst laag. Hoe hoger het scheidingsvermogen is des te beter zijn de lijnen en figuren op het scherm. Het maximale scheidingsvermogen op de Enterprise is 672 puntjes horizontaal en 512 puntjes verticaal.

SCHIJF
DISK

Een ronde schijf met een magnetiseerbare laag. Wordt gebruikt in combinatie met een schijfaandrijving om programma's en gegevens op te slaan die later weer gelezen kunnen worden. Gegevens kunnen veel sneller op schijf vastgelegd en weer opgehaald worden dan op cassette. Schijven zitten in een permanent plastic hoesje, maar als dit verwijderd wordt (dit kunt u alleen doen met schijven die op de een of andere manier onherstelbaar beschadigd zijn!) lijken de schijven heel veel op een flexibele plaat. De gebruikelijke schijven voor de Enterprise zijn de 3½ inch 'microfloppies' (diskettes).

SCHRIJVEN
WRITE

Gegevens of informatie ergens in stoppen. U kunt bij voorbeeld gegevens naar een bestand schrijven.

SLEUTELWOORD
KEYWORD

Ieder BASIC woord. Elk woord heeft zijn eigen betekenis.

SOFTWARE

Programmatuur; een programma is programmatuur. Instructies voor de computer die veranderd kunnen worden.

SORTEREN
SORT

Een bestand of een lijst opnieuw ordenen, met name alfabetisch.

SPRONG
JUMP

Zie VERTAKKING (BRANCH).

STRING

Een reeks tekens die niet begrepen maar wel verwerkt worden door de computer. In een uitgelijst programma in BASIC verschijnt een string ofwel als een variabele met als laatste teken $ ofwel als een serie tekens tussen aanhalingstekens.

STRUCTUUR
STRUCTURE

De manier waarop de delen van een programma zijn georganiseerd; dit in tegenstelling tot een algoritme — de weg waarlangs een probleem opgelost wordt.

SUBROUTINE

Nagenoeg hetzelfde als functie. Een gedeelte van een programma dat onder controle van de rest van het programma een specifieke taak uitvoert. Een functie heeft een specifiek begin en einde. Een subroutine wordt op basis van regelnummers bestuurd met de woorden GOSUB en RETURN. Een functie heeft een naam.

SUFFIX
SUBSCRIPT

Een toegevoegde aanduiding wanneer u toegang hebt tot een reeks. Voorbeeld: in geval van ARRAY(X) is X het suffix.

TAAL
LANGUAGE

Een manier om een computer te programmeren (een systeem van communicatie met de computer). BASIC is een taal. Andere programmeertalen zijn Pascal, FORTRAN, Forth, LISP en LOGO. Elk hiervan is ontworpen voor een speciale manier van programmeren.

TEKEN
CHARACTER

Een symbool dat gebruikt wordt voor weergave van informatie — letters, cijfers, bewerkingstekens en interpunctiesymbolen zijn alle tekens.

TESTEN
TEST

Een programma proberen om te zien of het werkt. In programmeertermen betekent testen: nagaan of iets

aan een voorwaarde voldoet. De computer test variabelen in IF of CASE opdrachten om te zien of ze overeenstemmen met een of andere opdracht.

TOEGANG HEBBEN TOT
ACCESS

Het ophalen van gegevens van een randapparaat — bij voorbeeld een printer of een cassètte. Ook het onttrekken van gegevens aan een programma zoals bij voorbeeld een gegevensbank.

UITDRUKKING
EXPRESSION

Een aantal cijfers of woorden die een waarde hebben, nadat een en ander uitgerekend is. Een voorbeeld van een numerieke uitdrukking is X*6; BASIC kent ook tekstuele uitdrukkingen.

UITVOER
OUTPUT

Alles dat de computer verlaat: beelden, muziek, afgedrukte lijsten, uitkomsten, etc.

VARIABELE
VARIABLE

Een getal dat of een string die veranderen kan; daarom heeft een variabele een naam.

VERTAKKING
BRANCH

Een plek in een programma waar de uitvoering van de opeenvolgende regelnummers gestopt wordt en de computer een ander gedeelte van het programma draait — bij voorbeeld een functie of een subroutine. Synoniem met JUMP.

VERTOLK-
PROGRAMMA
INTERPRETER

Een speciaal computerprogramma dat van de ene computertaal naar de andere vertaalt. Het BASIC op de Enterprise is een vertolkprogramma. Vertolk-programma's zijn niet te vergelijken met programma's die voor een definitieve vertaling zorgen tussen twee computertalen; deze programma's kennen we als compileerprogramma's.

VERWERKEN
PROCESSING

Bewerkingen op gegevens uitvoeren. Dit omvat berekeningen met getallen, grafische gegevens-verwerking (met gebruikmaking van coordinaten, etc.) en in feite al het andere dat een computer uitvoert.

VOORWAARDELIJK
CONDITIONAL

Omschrijving van een gedeelte van een programma dat gebruik maakt van voorwaarden voor het nemen van beslissingen. IF/THEN is een voorwaardelijke opdracht; de ruwe vertaling hiervan luidt als volgt: 'Gesteld dat A>1Ø, dan (THEN)...'.

WISSEN
WIPE

Leegmaken, ofwel een geheugenmedium (bij voorbeeld een schijf of cassette) ofwel het computergeheugen.

ABS	225
ACOS	225
ALLOCATE	161, 172
ANGLE	225
arrays, numeric	85
arrays, one-dimensional	85
arrays, string	87
arrays, two-dimensional	86
ASCII	124
ASK	172
ASIN	225
ATN	225
ATTRIBUTES	217
AUTO	40, 172
Basic	18
BEAM	106, 217
BIAS	217
BIN	225
BLACK	225
BLUE	225
BORDER	119–218
branches	145
broadcasting	157
CALL	173
calling functions	93
CAPTURE	173
CASE	173
cassette handling	56
CAUSE EXCEPTION	153, 173
CEIL	225
CHAIN	174
channels, introduction to	151–152
CHARACTER	218
character matrix	122
character set	22, 121
characters, what are	22
CHR$	121, 225
circles, ellipses	108
CLEAR	174
clearing screen	7
CLOSE	174
CODE	174
COLOUR	109, 218
colour modes	110
colour options	218, 107

colour selection 111
comment lines 25
computer languages 18
concatenation (aaneenschakeling) 69
CONTINUE 7, 175
COPY 175
correcting mistakes 5
COS 225
COSH 226
COT 226
crash 141
CSC 226
'ctrl' key 52
cursor 5, 34, 218
CYAN 226
DATA 175
DATE/DATE$ 175, 225
decisions 76–84
declaring variables 28, 88
DEF 176
DEFAULT CHANNEL 211
DEG 226
DELETE 19, 41, 179
device names 196–97
DIM 150, 180
disks 59
DISPLAY 180
DO/LOOPS 26, 70, 181
dummy variables 101
EDIT 181
editing 46–51
EDITOR BUFFER 211
EDITOR KEY 211
EDITOR VIDEO 211
eighty columns 208
ELSE 78, 182
END 182
'enter' key 5, 8
ENVELOPE 126, 182
EPS 226
'erase' key 5
error messages 235–241
'esc' key 50
exception handling 153–155, 182
EXIT DEF 183

EXIT DO	183
EXIT FOR	183
EXIT HANDLER	183
exiting from word processing	51
exiting loops and blocks	183
EXLINE	226
EXOS	232-34
EXP	226
expressions	31-35
EXSTRING$	226
EXT	183
EXTYPE	226
FAST SAVE	211
files	59
FKEY	212
FLUSH	184
FOR	184
FOR/NEXT loops	72
FP	226
FREE	226
function key operations	54
function keys, setting	52-53
functions and variables	94
functions, calling	93
GET	185
global variables	94, 178
GOSUB	147, 185
GOTO	146, 185
graphic origin	105
GRAPHICS	104-120, 185
graphics page	104, 116
GREEN	227
HANDLER	186
HEX$	161, 227
'hold' key	16
IF	187
IF blocks	77
IF/THEN	76, 187
IMAGE	188
immediate mode	21
IN	227
INF	227
INFO	189
INK	111-115, 219
INKEY$	65, 227

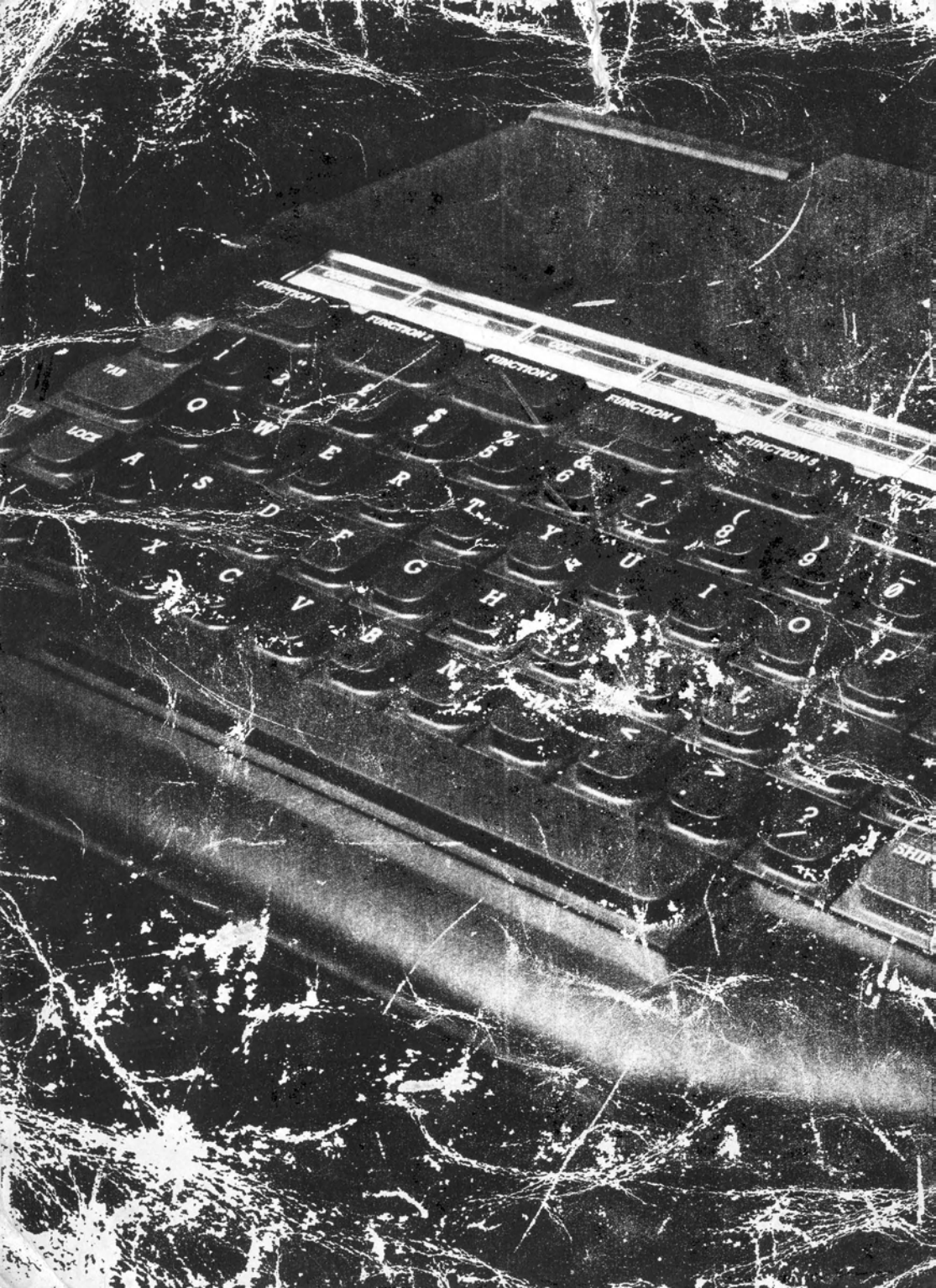
INPUT (and its trimmings)	189
input and output	232-33
input checking	140
INPUT PROMPT	25
insert mode	45
INT	227
INTERRUPT	131, 212
IP	227
joy	227
joystick	43
justification	47
key click	213
key delay	213
key rate	213
keywords, what are	22
LBOUND	227
LCASE$	67, 227
LEN	63, 228
LET	28, 190
LINE INPUT	190
LINE MODE	116, 219
line numbers	19, 172
LINE STYLE	116, 219
LIST	7, 41, 199
LLIST	191
LOAD	191
local variables	94, 127
LOG	228
LOG 2	228
LOG 10	228
LOOK	192
LOOP	70, 192
LPRINT	192
LTRIM$	228
machine code	161
machine options	211
MAGENTA	228
MAX	228
MAXLEN	88, 228
memory usage	232
MERGE	57, 192
MIN	228
MOD	228
modular programming	134
nested loops	74

net	156-160
NET CHANNEL	213
NET MACHINE	213
NET NUMBER	213
network channels	158
NEW	7, 192
NEW ALL	192
NEXT	192
NUMERIC	192
ON	193
OPEN	193-97
operator priority	32-35
OPTION ANGLE or BASE	198
ORD	122, 228
OUT	198
overwrite mode	45
pages and channels	117
PALETTE	112-115, 219
PAPER	22
parameter passing	103
parameter referencing	103
PEEK	228
PI	228
PING	198
PLOT	198
PLOT PAINT	199
POKE	200
POS	228
PRINT	200
PRINT AT	37, 200
printing text	50
program	201
program elements	134
program lines	19
progam, modular	134
program, multiple	169
program title	201
program, what is	6
RAD	229
random numbers	78-80
RANDOMIZE	201
READ/DATA	89, 134, 201
RED	229
REDIRECT	201
reference parameters	178

reference section guidelines 168
RELEASE 130
relational operators 31
REM 229
REMark 202
REM 1 215
REM 2 215
remote control 215
RENUMBER 40, 202
'reset' button 15
resolution 104
RESTORE 91, 203
RETRY 203
return 203
RGB 229
RND 229
ROUND 229
RTRIM$ 229
RUN 6, 203
SAVE 203
saving to cassette 50
SCROLL 220
SEC 229
SELECT 204
SELECT CASE 81, 204
SERIAL BAUD 214
SERIAL FORMAT 214
SET 205
SGN 229
'shift' key 6
SIN 229
SINH 229
SIZE 230
SOUND 125-133, 205
SOUND BUFFER 233
SOUND OPTIONS 223-24
sound queues 129
sound sources 223
sound statements 125
SOUND STYLE 223
spaces 63
SPEAKER 224
SPEEK 230
SPOKE 206
SQR 21, 230

START	54, 207
STATUS	215
STEP	40, 72
STOP	207
'stop' key	6, 16
STR$	230
STRING	207
strings, adding together	64
strings, introduction to	22, 63–69
substrings	65
TAB	230
tabulation	48
TAN	230
TANH	230
TAPE LEVEL	215
TAPE SOUND	215
teletext primary colours	112
TEXT	105, 208
Text formatting	36–39
THEN	208
TIME/TIME$	208, 230
TIMER	215
TOGGLE	208
TRACE ON/OFF	208
trig functions	230
TRUNCATE	230
turtle graphics	107
TYPE	209
UBOUND	230
UCASE$	68, 230
UNTIL	71
user defined characters	122, 218
using variables	29
USR	230
VAL	68, 230
variable	26 216
variable names	27
VERIFY	57, 209
VIDEO COLOUR	221
VIDEO MODE	221
video options	217–22
VIDEO X	221
VIDEO Y	221
WAIT DELAY	209
WHEN	209

WHILE 181
WHITE 230
WORD$ 164, 231
word processing mode 46-51
word processing functions 40-45
YELLOW 231


FUNCTION 1
FUNCTION 2
FUNCTION 3
FUNCTION 4
FUNCTION 5
TAB
CAPS LOCK
SHIFT