



TURBO BASIC XL 1.5 MANUAL

door:

Wil Braakman



TURBO BASIC XL 1.5. MANUAL INDEX

Inleiding:

- Voorwoord..... 1
- Hoe moet ik dit manual gebruiken? 3

Hoofdstuk 1:

- De Basic versies..... 1.1

Hoofdstuk 2:

- Hoe start ik basic op 2.1
 - Atari Basic
 - Turbo Basic XL 1.5
- Uitvoeringsmodi 2.3
 - Indirecte mode
 - Directe mode
- De Basic Programma Editor 2.4
- Speciale functietoetsen voor de Editor 2.4
- De Interpreter 2.5
- Algemene informatie over programmeren in Basic 2.7
- Karakterset 2.8
- Gereserveerde uitdrukkingen 2.9
- Variabelen 2.11
 - Hoe moet ik een variabele noemen?
 - Hoe ken ik een waarde toe aan een variabele?
 - Stringvariabelen.
 - Numerieke variabelen.
 - Een dimensionale arrays.
 - Twee dimensionale arrays.
- Numerieke uitdrukkingen en operatoren 2.14
 - Rekenkundige operatoren.
 - Relationele operatoren.
 - Logische operatoren.
 - Functionele operatoren.
- String uitdrukkingen en bewerkingen 2.16
 - Samenvoegingen.
 - Stringbewerkingen.

Hoofdstuk 3:

- Beschrijving van:
 - Functies.
 - Statements.
 - Commando's.



Aanhangsel A:

- Foutmeldingen en oplossingen A. 1

Aanhangsel B:

- Basic diskette Input en Output B. 1

Aanhangsel C:

- Atascii codes..... C. 1
- Toetsenbord codes..... C. 3

Aanhangsel D:

- Programma's en programmadelen..... D. 1

Tweede ongewijzigde druk, april 1988.

Dit is een uitgave van de Stichting Atari Gebruikers
Postbus 2095, 5202 CB 's-Hertogenbosch

*Gehele of gedeeltelijke overname van de inhoud is alleen
toegestaan met onze schriftelijke goedkeuring.*



TURBO BASIC XL 1.5 MANUAL

INLEIDING



TURBO BASIC

Voorwoord

TURBO BASIC XL 1.5 is een ontwerp van de 24 jarige Frank Ostrowski uit Munchen (1985) en gepubliceerd door het Duitse computertijdschrift HAPPY COMPUTER in december 1985. Alle Atari 8 bits-computergebruikers en vooral de schrijver van dit manual (=handboek) is beiden zeer dankbaar. De een voor het ontwerp en de ander voor de publicatie.

Het TURBO BASIC XL 1.5 is tot nu toe (1987) de snelste basicversie voor de Atari 8 bits-computers met een minimaal geheugen van 64 Kb.

Door het gebruik van deze basicversie krijgt men ook nog eens extra bytes vrij geheugen. In Atari Basic is voor de programmeur 32274 bytes vrij en in TURBO BASIC XL 1.5 is dit 34017 bytes.

Het TURBO BASIC XL 1.5 leent zich zeer goed om gestructureerd te programmeren. De 'vieze' sprongopdracht GOTO kan nu eindelijk in de prullebak worden gedeponeerd. De lusconstructies zoals REPEAT UNTIL, WHILE WEND en DO LOOP zijn veel geschikter, sneller en beter leesbaar.

Aangezien voor Atari voldoende boeken met uitleg aanwezig zijn en voor TURBO BASIC XL 1.5 nog geen, heb ik besloten hier wat aan te doen.

Het resultaat is een manual (dus geen leerboek) voor zowel ATARI BASIC (REV B. en C.) als TURBO BASIC XL 1.5.

Elk commando, statement en instructie wordt in dit manual behandeld en omschreven naar zijn syntax. Vindt u een instructie niet in dit manual, dan bestaat het voor beide dialecten ook niet.

Frank Ostrowski heeft niet alleen dit zeer slimme en snelle Basic ontworpen. Hij heeft hiervoor tevens een compiler en een runtime programma geschreven.

Ook deze zijn door Happy Computer gepubliceerd (Happy Computer Sonderheft 1 02/86).

Volgens publicatie van Happy Computer is het gehele pakket (Turbo basic XL 1.5, Compiler en Runtime) public domain.

Daarom is bij dit manual tevens gratis een diskette gevoegd met daarop de volgende files:

- <1> AUTORUN.SYS
Dit is het eigenlijke TURBO BASIC XL 15.
- <2> COMPILER.COM
De Compiler die Turbo Basic en Atari Basic programma's omzet naar machinetaal.
- <3> RUNTIME.OBJ
Het Runtime programma waar de gecompileerde files mee moeten worden gerund.



Het TURBO BASIC XL 1.5 heeft Microsoft-achtige commando's en een Pascal aandoende structuur.

Mensen die dus met een IBM (of clone) te doen hebben zullen in dit manual zeker functies en commando's terugvinden, die zij gewend zijn te gebruiken met genoemde Personal Computers.

Ik hoop dat een ieder die dit manual door heeft geworsteld zijn profijt er mee kan doen.

Ik wens een ieder net zoveel plezier van het gebruik van dit manual als dat ik zelf er reeds van heb gehad. Ook al heb ik dit geschreven met een bepaalde voorkennis van computers en programmeren, toch ontdekt men aldoende dat men er altijd iets bijleert.

Wil Braakman (04/87).

HOE MOET IK DIT MANUAL GEBRUIKEN ?

Wil men dit handboek juist gebruiken, moet men toch enige ervaring hebben in de basisprincipes van het programmeren. Het is niet de bedoeling dat ik u leer hoe u moet programmeren.

Het manual is verdeeld in 3 hoofdstukken en een aantal aanhangsels.

- Hoofdstuk 1: Geeft een overzicht van de Basicversies voor de Atari computers.
- Hoofdstuk 2: Geeft kort weer hoe u uw Basic opstart en gebruikt.

Geeft algemene informatie wat u dient te weten voordat u aan de slag gaat met programmeren.
- Hoofdstuk 3: Het hart van 't manual. Het "Naslagwerk". Het bevat de syntac van elke instructie, statement en functie in zowel Atari Basic als Turbo Basic XL 1.5 in alfabetische volgorde.

De aanhangsels bevatten andere nuttige informatie, zoals een complete lijst van foutmeldingen, ATASCII-codes e.d.

Ik raad u aan het hoofdstuk 2 aandachtig te lezen, zodat u vertrouwd raakt met Basic en de gebruikte codes in het naslagwerk.

Daarna kunt u de stap maken naar hoofdstuk drie; "Het Naslagwerk". Hier vindt u de informatie hoe elke instructie of functie gebruikt moet worden in uw programma's.

In het naslagwerk zijn verschillende tekens gebruikt, die een bepaalde functie/bedoeling hebben.

*** bij versie, betekent dat de instructie of functie in de betreffende taal kan worden gebruikt.

[] betekent, dat alles wat hiertussen staat niet verplicht is om in te geven bij het programmeren.

... betekent, dat een opdracht, variable of anderszins zovaak als u wilt mag worden herhaald.

Voorbeeld: INPUT [# Kanaal,] var [,var] ...

Dit betekent dat INPUT het statement is, het moet worden gevolgd door een variabele (de eerste var). Indien gewenst mag ook een kanaal welke met OPEN is gespecificeerd worden aangesproken en er mogen meerdere variabelen (tweede var enz.) worden gebruikt.



Of zoals bij het andere INPUT statement:

```
INPUT ["tekst"], var [, var] ...
```

Het statement INPUT moet gevolgd worden door een variabele (de eerste var).

Niet verplicht is de tekst en meerdere variabelen.

Komma's, aanhalingstekens binnen [] zijn wel verplicht als deze worden gebruikt.

Mocht u bij het gebruik van hoofdstuk drie een gebruikt teken niet begrijpen, kijk dan steeds het bovenstaande na.



TURBO BASIC XL 1.5 MANUAL

HOOFDSTUK 1.





HOOFDSTUK 1

De Basic versies.

Voor de Atari 8 bits computers zijn momenteel de volgende basicversies (dialecten) beschikbaar.

Atari Basic Rev. A, B en C
Turbo basic XL 1.5
Basic XE (normaal)
Basic XE (fast)
Action
MMG-compiler
ABC-compiler

De laatste twee zijn eigenlijk geen talen maar compilers (vertalers naar machinetaal).

In dit manual worden alleen de volgende Basicversies behandeld:

Atari Basic en Turbo Basic XL 1.5

In hoofdstuk drie wordt verschil gemaakt of een instructie geldig is in een van de twee dialecten en/of deze geldig is bij het compileren van een basicprogramma.

Het wordt dan als volgt aangegeven:

VERSIE: ATARI BASIC	TURBO BASIC	COMPILER
***	***	***

De sterretjes *** geven aan dat de instructie of functie inderdaad toegankelijk is in het desbetreffende basic (dialect) en of compiler.

Opgemerkt zij nogmaals dat voor TURBO BASIC XL 1.5 een drive nodig is met een computer met minimaal 64 Kb geheugen.

Er is wel een aanpassing voor cassetterecorder gebruikers, doch dan mist men veel van de goede kwaliteiten van dit basic.

Daarom is bewust het cassette-basic hier weggelaten en niet verder besproken. De echte basic-programmeur kan zich toch niet echt volledig ontplooiën zonder een diskdrive.

Speciale toepassingen in Basic met drive zijn:

- * Input/output naar de diskette in tegenstelling tot een cassette-recorder. Een diskdrive is namelijk een adresseerbaar medium.
Zie aanhangsel B. Speciaal gebruik van Input/Output bij gebruik van bestanden in Basic.
- * Snellere transport van informatie van en naar het medium. Waardoor een database ook bruikbaar wordt in tegenstelling tot een geschreven notitie.





TURBO BASIC XL 1.5 MANUAL

HOOFDSTUK 2.





HOOFDSTUK 2

Hoe start ik basic op ?

ATARI BASIC:

Aangezien we met een drive werken gaan we als volgt te werk.

- schakel eerst uw beeldscherm en drive in. De volgorde is hiervoor niet belangrijk. Plaats vervolgens een diskette in uw drive, waarop u uw eventuele te schrijven programma's naartoe kunt schrijven (saven). Niet een diskette gebruiken waar een 'bootfile' (zelfopstartend machinetaal programma) opstaat, want dan komt u in het desbetreffende programma terecht en dat willen we nu juist niet.
- schakel dan pas uw computer in. De drive met de daarin geïnstalleerde diskette zal dan beginnen te werken (busylight aan). Op deze manier herkent de computer dat er een drive is aangesloten; wat weer nodig is om van diskette te lezen en naar diskette te schrijven (Read/Write).
- na enig gedraai en gebrom zal op uw scherm dan het woordje READY verschijnen en dan weet u dat u in Atari Basic zit. De commando's in hoofdstuk drie waar *** onder de versie ATARI BASIC staan kunnen nu gebruikt worden.

TURBO BASIC XL 1.5:

Om Turbo Basic XL 1.5 te kunnen opstarten moet eerst enig huiswerk worden gedaan.

- 1) Start uw Master Dos 2.0 of 2.5 diskette op. Als READY op uw scherm verschijnt, roep dan DOS op door DOS in te typen en op de RETURN toets te drukken.
- 2) Formateer vervolgens met optie I drie lege diskettes.
- 3) Schrijf op de geformateerde diskettes met optie H de DOS files (DOS.SYS en DUP.SYS).
- 4) Plaats uw Master Turbo Basic diskette welke gratis bij dit manual wordt meegeleverd in uw drive.
- 5) Kies vervolgens optie 0 (duplicate file) van het Dosmenu. Als de vraag verschijnt "Which file to move" typ dan AUTORUN.SYS en druk op de RETURN toets en volg de instructies die volgen nauwgezet op.
- 6) Herhaal punt 4 en 5 nogmaals, maar typ nu de filenaam COMPILER.COM. Als de destination diskette wordt gevraagd doe dan de tweede nieuw geformateerde diskette in de drive (niet die waar reeds AUTORUN.SYS heen is gecopieerd.)
- 7) Herhaal punt 4 en 5 nogmaals voor het programma RUNTIME.OBJ welke op de derde nieuw geformateerde disk moet komen te staan.

U heeft nu vier diskettes:

- uw Master diskette (veilig ergens opbergen en niet meer gebruiken).
- een diskette met AUTORUN.SYS erop. Dit is uw TURBO BASIC XL 1.5 werkdiskette.
- een diskette met COMPILER.COM.
Dit is uw diskette waar u uw programma's mee gaat compileren.
- een diskette met RUNTIME.OBJ. Dit is uw diskette waar u uw gecompileerde programma's mee moet runnen.

8) Kies nu optie E van het Dosmenu.

9) Plaats de diskette in de drive waar COMPILER.COM op staat en druk op de RETURN toets. Typ vervolgens in COMPILER.COM,AUTORUN.SYS en druk weer op RETURN. Uw Compilerprogramma wordt herbenoemd in Autorun.sys. De komma tussen de beiden namen niet vergeten.

10) Herhaal punt 9 voor de laatste disk waar RUNTIME.OBJ op staat. Typ nadat u E heeft ingevoerd:
RUNTIME.OBJ,AUTORUN.SYS.

Als alles goed is gegaan moet u nu op de drie geformateerde diskettes overal AUTORUN.SYS hebben staan. Controleer dat met optie A van het Dosmenu. A + 2x op de RETURN toets drukken.

Nu bent u klaar om uw TURBO BASIC XL 1.5 op te starten. Er zijn twee mogelijkheden om uw TURBO BASIC XL 1.5 op te starten, te weten:

- a) Via uw Dos-menu optie L. Enter de naam AUTORUN.SYS en TURBO BASIC wordt geboot.
- b) De makkelijker manier is. Schakel uw computer uit. Doe diskette 1 met Turbo Basic (nu AUTORUN.SYS) in uw drive en schakel uw computer weer in. TURBO BASIC XL 1.5 wordt dan automatisch geboot. NIET OPTION ingedrukt houden.

Als READY op het scherm verschijnt bent u in TURBO BASIC. Dit kunt u controleren door DIR te typen en op de RETURN te drukken. De diskdirectory moet dan op het scherm verschijnen. U zult zien dat de volgende files op de diskette staan:

DOS.SYS, DUP.SYS en AUTORUN.SYS.

DUP.SYS is niet meer nodig en kan worden verwijderd. Doe dit als volgt. Typ: DELETE "D:DUP.SYS" en druk op RETURN. Controle weer met DIR.

U heeft reeds kennis gemaakt met het TURBO BASIC XL 1.5 door het gebruik van DIR en DELETE te gebruiken, wat in feite DOS-commando's zijn. Dit is veel directer, duidelijker dan de instructies uit het DOS menu.

Uw programma kan nu worden ingevoerd. Of er kan een programma worden geladen en gerund.



Het laden en runnen van een basic programma kan op twee manieren, te weten:

- a) Laadt eerst een programma in het geheugen met LOAD "D:filename.ext" en voer dan een RUN uit, of
- b) Run een programma meteen als het in het geheugen wordt geladen. Dit gaat met RUN "D:filename.ext"

U I T V O E R I N G S M O D I

Als Basic (Atari Basic of Turbo Basic) eenmaal opgestart is, wordt de aanduiding READY op het scherm getoond. READY betekent alleen maar dat Basic klaar is en wacht op uw instructies om uit te voeren. Deze toestand is ook wel het 'command level' genaamd. Op dat moment mag u met Basic gaan praten in een van de twee volgende modi: direct mode of indirect mode.

INDIRECTE MODE:

Een programma dat u schrijft doet u in de indirecte mode. Om basic mede te delen dat u in de INDIRECTE MODE werkt moet u met regelnummers werken. De regelnummers zijn een gedeelte van het programma en worden derhalve met de rest van het programma bewaard. Het programma in het geheugen opgeslagen kan worden uitgevoerd met het RUN commando. Bij voorbeeld:

```
READY
10 PRINT 20+5
RUN
25
READY
```

DIRECTE MODE:

DIRECTE MODE betekent dat u Basic zegt het ingevoerde commando meteen uit te voeren zodra op de RETURN toets wordt gedrukt. Dit maak je Basic duidelijk door geen regelnummers te gebruiken voor de statements, commando of instructies. Op deze manier kan men de computer zeer goed gebruiken voor kleine en grotere berekeningen. De resultaten worden door de computer onthouden indien men ze in variabelenamen zet. De statements enz. worden door de computer niet onthouden. Deze mode is erg nuttig om fouten op te sporen in uw programma's. Voorbeeld van een berekening is:

```
READY
PRINT 20+5
25
READY
```

Het gebruik van variabelen in DIRECT MODE gaat als volgt:

```
READY
A=20:B=5:C=A+B:PRINT C
25
```



READY

Indien men hierna wil weten welke waarde respectievelijk A, B of C hebben, behoeft men slechts het volgende in te voeren:

```
READY
PRINT A,B,C
20 5 25
READY
```

Het zal duidelijk zijn dat in bovenstaande voorbeeldjes het woordje READY niet moet worden ingetypt. READY staat reeds op het scherm.

Een aardigheidje in TURBO BASIC is nog dat men hier voor statements, command's of instructies zowel kleine als grote letters en zelfs inverse letters mag gebruiken. TURBO BASIC herkent de instructies en zet deze automatisch om in hoofdletters. Dit in tegenstelling tot het normale ATARI BASIC.

DE BASIC PROGRAMMA EDITOR

Editor betekent letterlijk bewerker of redacteur. Voor de duidelijkheid zullen we het bij het engelse EDITOR houden. Elke willekeurige regel welke men aan Basic mededeelt wordt uitgevoerd door de Basic programma editor. De Editor is een schermregeeditor. Dat wil zeggen men kan veranderen waar men maar wilt op het scherm. Maar er kan maar een regel tegelijk worden gewijzigd. De verandering wordt ook dan nog maar geaccepteerd als men in die bewuste regel op de RETURN toets heeft gedrukt.

Het gebruik van deze editeermethode bespaart een hoop werk. Vooral als men bedenkt dat er computers zijn die deze mogelijkheid niet bezitten. Als in de regel een fout wordt gemaakt en de RETURN toets is reeds ingedrukt, dan moet men de gehele regel opnieuw in typen. Wat een werk!! Om enigszins vertrouwd te geraken met de Editor is het het beste om willekeurig enige regels in te typen en eens Kris Kras met de cursor, dit is het vierkante blokje op het scherm, over het scherm te fietsen. Dit wordt gedaan door de CONTROL toets ingedrukt te houden samen met een PIJL toets (links naast de RETURN en CAPS toets).

SPECIALE FUNCTIETOETSEN VOOR DE EDITOR

Hiermede zijn we meteen beland bij de voor de Editor belangrijke functietoetsen. Dit zijn:

CONTROL samen met een van de vier PIJL toetsen.
Dit heeft als resultaat dat de cursor wordt verplaatst zonder dat er iets wordt gewijzigd.

DELETE BACK SPACE toets.
Hiermee gaat de cursor terug en eet de karakters die hij tegenkomt op.



CONTROLE+INSERT samen maken op de cursorplaats een spatie.

SHIFT+INSERT voegen een gehele blanco regel in.

CONTROLE+DELETE BACK SPACE zelfde als bij DELETE.

SHIFT+DELETE verwijderd een gehele regel van het scherm. Dit heeft niet tot gevolg dat de regel uit het geheugen wordt verwijderd.

CAPS verandert van hoofdletters in kleine letters.

CONTROLE+CAPS roept de grafische tekens van de computer op.

CONTROLE+1 stopt het scrollen van de listing
nogmaals CONTROLE+1 start het scrollen weer.

RETURN toets is waarschijnlijk de belangrijkste toets van uw computer. Indien u klaar bent met het invoeren van programmaregels, instructies of in een programma vragen beantwoord dient u de RETURN toets te gebruiken. U deelt de computer eigenlijk dus mede dat u klaar bent met intikken.

De ATARI toets (geheel rechtsonder). Hiermee vertelt u de computer dat een karakter in inverse video moet worden afgedrukt. Dit wil zeggen dat de achtergrond- en voorgrondkleur moeten worden verwisseld.

CONTROL of SHIFT+CLEAR dit heeft tot gevolg dat het scherm schoon wordt gemaakt. De programmaregels blijven in het geheugen behouden.

Verder zijn er in TURBO BASIC XL 1.5 nog enkele commando's om de editor te activeren tot bepaalde daden. Zo is er het commando DELETE en RENUM, LIST en NEW. Zie voor een juiste toedracht hiervan in hoofdstuk drie in het alfabetisch naslagwerk.

De INTERPRETER, dit is de basicvertaler, reageert bij het maken van fouten praktisch altijd meteen. Fouten die betrekking hebben op de syntac (de opbouw) van een statement of instructie worden prompt gemeld met een ERROR aanduiding. De cursor (wit blokje) staat dan op een plaats die de INTERPRETER niet meer kan begrijpen.

Fouten die betrekking hebben op de uitvoering van het programma worden eerst getoond als RUN wordt ingevoerd. En ook dan nog maar alleen indien het programma de desbetreffende fout moet uitvoeren.

Een veel voorkomende fout die wordt gemaakt is dat men strings vergeet te dimensioneren met DIM. Er wordt dan een Error 9 gegeven door het systeem.

Wilt u de volledige foutmeldingen bekijken raadpleeg dan aanhangsels A.

Tot slot zij nog opgemerkt dat men een regel van een

programma eveneens kan editeren door deze regel eerst met LIST+regelnummer op te roepen en vervolgens te veranderen. In de foute regel staat aan het begin steeds ERROR - enz. enz. Dit ERROR gedeelte moet men verwijderen, anders blijft de INTERPRETER een foutmelding geven. Dit kan gedaan worden met de cursor achter het regelnummer te plaatsen en dan CONTROL+DELETE BACK SPACE in te drukken. Net zolang tot de juiste informatie op de regel overblijft. Bevestig vervolgens met een <RETURN>.



ALGEMENE INFORMATIE OVER PROGRAMMEREN IN BASIC

REGELFORMAAT:

Programmeregels in Basic hebben het volgende formaat:

```
nnnn BASIC statement[:Basic statement...][:. commentaar  
...]
```

en ze worden afgesloten met een <RETURN>. Dit formaat wordt onder meer gedetailleerd besproken.

Regelnummers: "nnnn" geeft de regelnummers aan, welke een tot vijf posities groot mag zijn. Elke BASIC programmaregel begint met een regelnummer. Deze regelnummers worden gebruikt om de volgorde weer te geven van de wijze waarop ze in het geheugen zijn opgeslagen en als raadpleeggegeven voor het editeren. De regelnummers moeten liggen tussen 0 en 65535.

Basic statement: Een BASIC statement is oftewel uitvoerbaar of niet uitvoerbaar. Uitvoerbare statements zijn programma-instructies die BASIC vertellen wat deze moet doen als het programma wordt uitgevoerd. Bij voorbeeld: PRINT A is een uitvoerbaar statement. Niet uitvoerbare statements (opdrachten) zijn o. a. DATA en REM, welke geen actie van BASIC verwachten als Basic ze tegenkomt. Alle BASIC opdrachten zijn in het volgende hoofdstuk uitgebreid uitgelegd.

Indien gewenst mogen er meerdere statements (instructie/opdrachten) op een regel worden vermeld. Deze dienen dan wel door een dubbele punt [:] te worden gescheiden; zodat Basic dit ook als zodanig herkent. De totale lengte van een regel mag echter de grens van 128 niet overschrijden (een beltoon waarschuwt u). De rest wordt niet meer onthouden door BASIC.

Een voorbeeld van 'multiply statements':

```
READY  
10 FOR X=1 TO 5:PRINT X:NEXT X  
RUN  
1  
2  
3  
4  
5  
READY
```

Commentaar: Commentaar of REMARKS mogen aan het einde van een programmaregel worden toegevoegd onder gebruikmaking van een punt [.] of REM om het commentaar van het voorgaande te splitsen.



KARAKTER SET:

De BASIC Karakter (Karakter=teken) set bestaat uit alfabetische Karakters, numerieke Karakters, speciale Karakters en grafische Karakters. Dit zijn de Karakters die door Basic worden herkend.

De alfabetische Karakters zijn de "HOOFDLETTERS" en de "Kleine letters" van het alfabet. De numerieke Karakters zijn de tekens 0 tot en met 9.

De volgende tekens (speciale Karakters) hebben een specifieke betekenis voor Basic.

TEKEN	NAAM/BETEKENIS
	blanco
=	gelijk teken
+	plus teken
-	min teken
*	vermenigvuldigingsteken of sterretje
/	deelteken of schuine streep
\	backslash
^	machtsverheffen
(linker openingshaakje
)	rechter sluihaakje
#	nummerteken of kanaalaanduiding
\$	dollarteken of stringaanduiding
!	uitroepteken
&	ampersand
%	procentteken of integraanduiding
,	Komma
.	punt of decimale punt of remarkteken
;	punt komma of plaatsbepaling volgende opdracht
'	apostroph
:	dubbele punt of opdracht-splitser
?	vraagteken of PRINT opdracht
@	apeslinger (speciaal teken)
"	aanhalingsteken of tekstbegrenzer
<	kleiner dan teken
>	groter dan teken
_	onderlijningsteken

Dit is slechts een greep uit de vele tekens die het Atari toetsenbord rijk is. Voor een volledige lijst van alle Karakters zie aanhangsel C (ATASCII Karakters).

GERESERVEERDE UITDRUKKINGEN

In zowel ATARI als in TURBO BASIC hebben bepaalde uitdrukkingen of woorden een speciale betekenis voor het BASIC. Deze worden ook wel gereserveerde woorden/uitdrukkingen genoemd. Gereserveerde uitdrukkingen bevatten alle BASIC commando's, statements, functies en operatoren. Deze kunnen niet worden gebruikt als een variabelennaam, tenzij het LET statement wordt gebruikt.

Het wordt echter afgeraden om de gereserveerde uitdrukkingen met LET tot een variabelennaam te maken. Dit bevordert de leesbaarheid van het programma niet.

Onderstaand treft u een lijst van gereserveerde uitdrukkingen in BASIC aan. Indien deze alleen in TURBO BASIC XL 1.5 mogen worden gebruikt staat er een [*] sterretje achter.

ABS	*F [*]	PTRIG
ADR	FCOLOR [*]	PUT
ASC	FILLTO [*]	%PUT [*]
ATN	FOR	RAD
B []	FRAC [*]	RAND [*]
BGET [*]	FRE	READ
BLOAD[*]	GET	REM
BPUT [*]	%GET [*]	RENAME [*]
BRUN [*]	GO# [*]	RENUM [*]
BYE	GOSUB	REPEAT [*]
CHR\$	GOTO	RESTORE
CIRCLE [*]	HEX\$ [*]	RESTORE # [*]
CLOAD	IF	RETURN
CLOG	INKEY\$ [*]	RND
CLOSE	INPUT	RUN
CLR	INSTR [*]	SAVE
CLS [*]	INT	SETCOLOR
COLOR	*L [*]	SGN
COM	LEN	SIN
CONT	LET	SOUND
COS	LIST	SQR
CSAVE	LOAD	STATUS
DATA	LOCATE	STICK
DEC [*]	LOCK [*]	STOP
DEG	LOG	STR\$
DEL [*]	LOOP [*]	STRIG
DELETE [*]	LPRINT	TEXT [*]
DIM	-- [*]	TIME [*]
DIR [*]	MOD [*]	TIME\$ [*]
DIV [*]	MOVE [*]	TRACE [*]
DO [*]	#NAME[*]	TRAP
DOS	NEW	TRAP # [*]
DPEEK [*]	NEXT	TRUNC [*]
DPOKE [*]	NOTE	UINSTR[*]
DRAWTO	ON	UNLOCK [*]
DSOUND [*]	OPEN	UNTIL [*]
DUMP [*]	PADDLE	USR
ELSE [*]	PAINT [*]	VAL
END	PAUSE [*]	WHILE [*]



ENDPROC [*]	PEEK	WEND [*]
ENDIF [*]	PLOT	
ENTER	POINT	
ERL [*]	POKE	
ERR [*]	POP	
EXEC[*]	POSITION	
EXIT[*]	PRINT	
EXP	PROC [*]	

VARIABELEN

Variabelen zijn namen waaraan een bepaalde waarde wordt toegekend welke door BASIC kunnen worden herkend als zodanig. Naar de inhoud van de variabelen zijn er twee soorten; te weten de numerieke- en stringvariabelen. De numerieke variabele vertegenwoordigt altijd een cyferwaarde. Een stringvariabele omvat een karakter, welke een cyfer, letter of grafisch teken mag zijn.

De lengte van de variabelen mag maximaal 130 tekens lang zijn. Dit is namelijk de maximale lengte van de invoerbuffer. De inhoud van de variabelen mag zo groot zijn als er vrije geheugenruimte beschikbaar is.

Alle variabelen worden na RUN of CLR op nul gezet. Een numerieke variabele indien niet gespecificeerd is gelijk 0 (nul) en een string variabele eveneens. Stringvariabelen moeten vooraf worden gedimensioneerd. Dit wil zeggen de lengte van de string variabele moet vooraf worden opgegeven (hierover later meer).

HOE MOET IK EEN VARIABELE NOEMEN.

BASIC variabelenamen mogen zolang zijn als u maar wenst. Het is echter beter om in verband met de maximale lengte van de invoerbuffer niet meer dan 130 tekens te gebruiken. De tekens welke voor de variabelenamen moeten worden gebruikt mag vrijelijk door u zelf worden bepaald. De enige restrictie is dat de naam moet beginnen met een letter. Tevens mag geen gebruik worden gemaakt van de gereserveerde uitdrukkingen/woorden. Voor een complete lijst van gereserveerde uitdrukkingen/woorden zie de opgenomen lijst eerder in dit hoofdstuk.

Ook mogen de variabelenamen niet beginnen met een van de gereserveerde uitdrukkingen/woorden in genoemde lijst. Bij voorbeeld:

```
10 GETAL=10
```

Dit is niet toegestaan, aangezien GET een gereserveerde uitdrukking is voor BASIC. Dit levert tevens een foutmelding op.

Wel toegestaan is bij voorbeeld:

```
10 EENGETAL=10
```

Met string variabelen is het uiteraard eveneens zo.

HOE KEN IK EEN WAARDE TOE AAN EEN VARIABELE ?

Van belang is het variabele type; dit kan een string- of numerieke variabele zijn.

Stringvariabelen worden geschreven onder toevoeging van een dollarteken (\$) na het laatste teken. Bij voorbeeld:

```
A$="TURBO BASIC XL 1.5"
```

Het dollarteken \$ is een declaratie-teken. Het verklaart dat de variabele van het "string"-type is.

Wil men de waarde, zoals in bovenstaand voorbeeld toegekennen aan A\$ dan dient men het interne geheugen eerst mee te delen dat er een adres/vakje moet worden geopend en welke lengte (hoe groot) dit vak maximaal kan zijn.

Dit gebeurt met het DIM statement (DIM staat voor dimensionering; geeft een bepaalde dimensie aan een variabele). Om geen foutmelding te krijgen delen we de computer dus eerst mee hoe groot vakje A\$ moet worden. In ons voorbeeld dus 18 tekens lang.

Het is derhalve voldoende A\$ te dimensioneren op een lengte van 18. Genoemd voorbeeldje komt er dan als volgt uit te zien:

```
DIM A$(18):A$="TURBO BASIC XL 1.5"
```

Indien een string vooraf niet wordt gedimensioneerd, volgt een Error-melding "ERROR- 9".

Numerieke variabelen behoeven niet te worden gedimensioneerd, zoals bij stringvariabelen. In Turbo Basic wordt alleen verschil gemaakt tussen numerieke variabele typen van het integertype of single-precision type.

Singleprecision numerieke variabelen zijn alle denkbare namen/uitdrukkingen, zolang zij beginnen met een letter en niet worden afgesloten met een '\$' stringteken, bij voorbeeld:

```
A=10
JAN=15
VARIABELE2=11
PQR347AB=100
```

A, JAN enz. zijn allemaal numerieke variabelen.

Tevens zij opgemerkt dat geen gereserveerde basicwoorden mogen worden gebruikt; zie lijst eerder opgenomen in dit hoofdstuk.

Integer numerieke variabelen kunnen alleen worden toegekend in Turbo Basic. Hiertoe moet men het procentteken (%) toevoegen voor de variabelenaam:

```
A=1.5734
PRINT %A
1
```

In tegenstelling tot stringvariabelen kunnen numerieke variabelen in een tabel/reeks worden opgenomen. Deze tabel kan een of twee-dimensionaal zijn.

Om dit te kunnen realiseren moet de computer eerst weer meegedeeld worden hoe groot de tabel, ook wel array genaamd, moet zijn.

Een dimensionale arrays.

```
10 DIM A1(5)
```

Dit creëert een een dimensionale array genaamd A1. Alle elementen zijn van de single-precision numerieke waarde. De array heeft een lengte van 6 vakken. Aangezien vak 0 ook meetelt zijn er dus 6 vakken (laden in het ladenkastje). Alle elementen hebben in Turbo Basic tevens de waarde nul als ze worden gedimensioneerd.

Twee dimensionale arrays.

```
10 DIM A2(2,3)
```

Dit creëert een twee dimensionale array genaamd A2. Alle elementen zijn weer van de single-precision type en hebben de waarde nul. Tussen haakjes (2,3) geeft aan hoe groot de tabel/array is.

(2,3) betekent dat er 3 rijen en 4 kolommen zijn; dit omdat vak 0 ook meetelt.

In een tekening ziet het er als volgt uit:

Een dimensionale array

```

-----
      A1 (0)
-----
      A1 (1)
-----
      A1 (2)
-----
      A1 (3)
-----
      A1 (4)
-----
      A1 (5)
-----

```

Twee dimensionale array

```

                kolommen
-----
      A2 (0, 0)  A2 (0, 1)  A2 (0, 2)  A2 (0, 3)
R-----
y  A2 (1, 0)   A2 (1, 1)   A2 (1, 2)   A2 (1, 3)
e-----
n  A2 (2, 0)   A2 (2, 1)   A2 (2, 2)   A2 (2, 3)
-----

```

Het element in de 2de rij, 2e kolom is genaamd A2(1,1)

Een voorbeeld van een twee dimensionale tabel in program-
mavorm kan er o. a. als volgt uit zien:

```

10 DIM TABEL(2,3)
20 I=1
30 FOR RY=0 TO 2
40   FOR KOLOM=0 TO 3
50     TABEL(RY,KOLOM)=I
60     I=I+2
70   NEXT KOLOM

```



```
80 NEXT RY
90 FOR RY=0 TO 2
100 FOR KOLOM=0 TO 3
110 PRINT TABEL (RY, KOLOM);
120 NEXT KOLOM
130 PRINT
140 NEXT RY
RUN

1 3 5 7
9 11 13 15
17 19 21 23
READY
```

Numerieke uitdrukkingen en operatoren.

Een numerieke uitdrukking mag een numerieke konstante of variabele zijn. Er mag eveneens gebruik worden gemaakt van combinaties van konstanten en variabelen met operatoren om een bepaalde numerieke waarde te krijgen.

We kunnen verschil maken tussen numerieke en string operatoren. In de meeste gevallen worden alleen de numerieke operatoren gebruikt; we onderscheiden de volgende categorieën:

- * Rekenkundige
- * Relatieve
- * Logische
- *, Functionele

Operatoren.

Rekenkundige operatoren (bewerkingen).

De rekenkundige bewerkingen zijn de meest gebruikte, zoals optellen, aftrekken, vermenigvuldigen enz..

Operator	Bewerking	Voorbeeld
^	machtsverheffen	A^B
*	vermenigvuldigen	A*B
/	delen	A/B
+	optellen	A+B
-	aftrekken	A-B
MOD	modulus	A MOD B
DIV	deling (integer)	A DIV B

Met enige "wiskundige" achtergrond komen de meeste bewerkingen u wel bekend voor. Misschien de modulus en division bewerkingen niet.

MOD geeft de restwaarde aan na een integer deling. DIV is de integer deling zonder rest.

Voorbeeld:

```
READY
10 A=14 MOD 8
20 PRINT A
RUN
```



```
6  
READY
```

```
10 A=10 DIV 4  
20 PRINT A  
RUN  
2  
READY
```

Het resultaat van MOD na deling is $14:8=1$ (integer) met als rest 6. Het resultaat van DIV na deling is $10:4=2.5$ integer is dus 2.

Voor nadere uitleg zie Hoofdstuk 3 onder MOD en DIV.

Relationele operatoren.

Relationele operatoren brengen twee waarden met elkaar in relatie of ook wel vergelijken deze met elkaar. Dit resulteert enerzijds in waar (-1) of onwaar (0). Het verdere verloop van een programma laat men meestal afhangen van dit resultaat. (Zie ook het statement IF in Hoofdstuk 3).

Operator	Bewerking	Voorbeeld
=	is gelijk	A=B
<> of ><	ongelijk	A<>B
<	kleiner dan	A	groter dan	A>B
<= of =<	kleiner of gelijk dan	A<=B
>= of =>	groter of gelijk dan	A>=B

Het "is gelijk"-teken (=) wordt eveneens gebruikt om een waarde aan een variabele toe te kennen (zie LET-statement in Hoofdstuk 3).

Rekenkundige en relationele bewerkingen kunnen ook samen worden gebruikt, zoals in:

```
A+B<(X-1)/Y
```

De uitdrukking is waar (-1) indien de waarde van A plus B kleiner is dan de waarde van X-1 gedeeld door Y.

Of in het volgende voorbeeld:

```
READY  
10 INPUT A  
20 IF A=500 THEN PRINT "GELIJK"  
30 IF A<>500 THEN PRINT "ONGELIJK"  
40 IF A<500 THEN PRINT "GETAL IS GROTER"  
50 IF A>500 THEN PRINT "GETAL IS KLEINER"  
60 GOTO 10
```

Zie ook IF en THEN statement in hoofdstuk 3.



Logische operatoren.

Logische bewerkingen veronderstellen een bepaalde logica. (Logisch heh!) Relationele bewerkingen gaan uit van een vergelijking, om via het resultaat het programma te beïnvloeden.

Logische operatoren worden echter meestal gebruikt om twee of meer vergelijkingen (relaties) met elkaar te vergelijken en dan een true (waar) of false (onwaar) waarde te krijgen bij een beslissing.

De logische bewerkingen maken gebruik van "waar-onwaar" waarden en geeft als resultaat waar of onwaar. Het is "waar" als de waarde niet gelijk aan nul (vergelijkbaar met -1 bij een relationele bewerking) of "onwaar" als de uitkomst gelijk is aan nul.

Het resultaat van de logische bewerking is een waarde welke weer "waar" is als het niet gelijk is aan nul of "onwaar" als het gelijk is aan nul. De waarde is berekend door een "bit met bit" vergelijking.

"WAAR ZIT HIER NU DE LOGICA ?!?"

De logische operatoren zijn:

NOT	logische ontkenning
AND	" vergelijking
OR	" ongelijk
EXOR	binaire excl. OR
&	" AND
!	" OR

Laatste drie alleen Turbo Basic.

Functionele operatoren.

Een functionele bewerking is een voor ingestelde bewerking met een vaste uitkomst volgens de rekenkundige regels. Functionele operatoren zijn:

SIN	(sinus)
SQR	(wortel)
LOG	(logaritme)
CLOG	(normale logaritme)
ABS	(absolute waarde)
ATN	(arctangens)
enz...	

Voor een complete lijst zie de lijst van gereserveerde basicwoorden eerder opgenomen in dit hoofdstuk en de uitleg hiervan in hoofdstuk 3 bij de desbetreffende functiebeschrijvingen.

String uitdrukkingen en bewerkingen.

Een string uitdrukking mag elke opeensomming van letters, tekens en cijfers of variabelen bevatten. Tevens mogen de strings worden samengesteld uit constanten en variabelen door middel van gebruik van bewerkingfuncties om een enkele string te creëren.

We maken onderscheid in twee categorieën:



- * Samenvoegingen
- * Bewerkingen

Opgemerkt zij dat ook de relationele bewerkingen samen met strings kunnen worden gebruikt om twee strings met elkaar te vergelijken. Dit zijn echter geen string bewerkingen aangezien als uitkomst een numeriek resultaat en niet een stringresultaat ontstaat.

Samenvoegingen.

Twee strings met elkaar verbinden tot een noemen we een samenvoeging. In het Engels is er een beter woord voor "concatenation". Aangezien Atari- en Turbo Basic het plus symbool (+) niet kent voor string samenvoegingen wordt dit omzeild door gebruik te maken van de stringfunctie LEN. Voor meer uitleg zie LEN-statement in Hoofdstuk 3.

Voorbeeld:

```
READY
10 DIM A$(15),B$(15),C$(20)
20 A$="TURBO "
30 B$="BASIC XL 1.5"
40 C$=A$
50 C$(LEN(C$)+1)=B$
60 PRINT C$
RUN
TURBO BASIC XL 1.5
READY
```

Zet eerst de waarde van A\$ in de hulpstring C\$ (regel 40). Zet vervolgens de waarde van B\$ achter de waarde van C\$ door de lengte van C\$ te bepalen (met LEN) en te verhogen met 1.

String bewerkingen.

Een string bewerking komt overeen met een numerieke bewerking met deze uitzondering dat niet een numerieke waarde doch een string resultaat het gevolg is. Een string functie kan in een uitdrukking worden gebruikt om een vooraf bepaald resultaat te krijgen. Basic heeft verschillende "vooringestelde" functies waarvan het resultaat "bekend" is.

Enkele functies zijn:

ASC, CHR\$, STR\$, LEN, INSTR, UINSTR enz....

Een volledige beschrijving vindt u in hoofdstuk 3





TURBO BASIC XL 1.5 MANUAL

HOOFDSTUK 3.





ABS
functie

=====

Omschrijving: Geeft de absolute waarde van een getal.

Versie : ATARI BASIC TURBO BASIC COMPILER
*** **

Formaat : A=ABS(x)

Opmerking : x mag elke numerieke waarde zijn

Voorbeeld : READY
PRINT ABS(7*(-5))
35
READY

Omschrijving: Geeft het decimale geheugenadres van het
eerste teken van een string.

Versie : ATARI BASIC TURBO BASIC COMPILER
*** **

Formaat : A=ADR(B\$)

Opmerking : B\$ mag elke string variabele zijn.

Voorbeeld : READY
PRINT ADR("ALTIJD")
65
READY

=====

Omschrijving: Geeft de ATASCII-code van het eerste teken van een string.

Versie : ATARI BASIC TURBO BASIC COMPILER
 *** *** ***

Formaat : A=ASC(B\$)

Opmerking : String-variabelen kunnen worden gebruikt.
B\$ kan ook b.v. ("A") of dergelijke zijn.

Voorbeeld : READY
 10 X\$="TEST"
 20 PRINT ASC(X\$)
 RUN
 84
 READY

=====

Omschrijving: Geeft de arctangens van een waarde in radian-
ten of graden.

Versie : ATARI BASIC TURBO BASIC COMPILER
*** **

Formaat : A=ATN(x)

Opmerking : x mag elke numerieke waarde zijn.
De ATN functie geeft de hoekwaarde van welke
de tangens x is. PI=3.141593

Voorbeeld : READY
10 PI=3.141593
20 RADIANT=ATN(1)
30 GRADEN=RADIANT*180/PI
40 PRINT RADIANT,GRADEN
RUN
.7853983 45
READY

=====
Omschrijving: Ondervanging van de BREAK toets.

Versie : ATARI BASIC TURBO BASIC COMPILER
 *** ***

Formaat : *B[teken]

Opmerking : teken - niet verplicht; kan zijn + of -
 *B of *B+
 Na deze opdracht wordt het drukken van de
 BREAK toets behandeld als zijnde een fout.
 Met TRAP kan de fout worden omgeleid, waar-
 door het programma niet stopt of stopt op een
 door de programmeur gewenste wijze.

*B-
 Heft de bovenstaande modus weer op.
 Bij RUN wordt automatisch *B- uitgevoerd.
 (default)

=====
Omschrijving: Laadt een binary file in het geheugen.

Versie : ATARI BASIC TURBO BASIC COMPILER
*** **

Formaat : BLOAD "D(n):filename.ext"

Opmerking : Gelijk aan de DOS menu keuze L met /N
na de filename.ext.
n is niet verplicht geeft het drivenummer
aan.

Voorbeeld : BLOAD "D:MYPROG.OBJ"

Omschrijving: Zendt een byte uit een bepaald blok geheugen-
adressen naar een te specificeren apparaat.

Versie : ATARI BASIC TURBO BASIC COMPILER
*** **

Formaat : BPUT #n, adr, len

Opmerking : # verplicht teken (IOCB aanduiding)
n kanaalnummer (1 t/m 7), welke verwijst naar
het met OPEN geopende bestand/file.
adr startadres van waaraf geschreven moet
worden.
len aantal te schrijven adressen vanaf adr.

Commando is gelijk aan:
FOR I=0 TO LEN-1
PUT #n, PEEK(adr+I)
NEXT I

Zie ook BGET, PUT en GET

Omschrijving: Laadt een binaire file en runt dit indien een
RUN-adres in de file aanwezig is.

Versie : ATARI BASIC TURBO BASIC COMPILER
*** **

Formaat : BRUN "D[n]:file.obj"

Opmerking : n is het drivenummer; niet verplicht
(default=1).
file.obj is het te laden/runnen bestand.
Dit commando is gelijk aan DOS menu optie L
zonder /N na de bestandsnaam.

=====
Omschrijving: Treedt uit de BASIC mode naar de SELF
TEST-mode.

Versie : ATARI BASIC TURBO BASIC COMPILER
*** **

Formaat : BYE of B.

Opmerking : Met de SELF TEST mode kunnen alle toetsen,
geluidskanalen en de RAM en ROM getest
worden.

Omschrijving: Geeft het teken behorende bij een ATASCII
code.

Versie : ATARI BASIC TURBO BASIC COMPILER
 *** *** ***

Formaat : A\$=CHR\$(n)

Opmerking : Tegenovergestelde van ASC
 zie ook PUT
 n moet vallen tussen de waarde 0 en 255

Voorbeeld : READY
 PRINT CHR\$(66)
 B
 READY

of

```
10 FOR I=0 TO 255
20 PRINT I;" ";CHR$(I)
30 NEXT I
```

Omschrijving: Tekent een cirkel of ellips om de punten
(x,y) met als radius xr.

Versie : ATARI BASIC TURBO BASIC COMPILER
*** **

Formaat : CIRCLE x,y,xr[,yr]

Opmerking : Bij gebruik van xr en yr (verschillende
radius) ontstaan ellipsen.

Voorbeeld : READY
10 GRAPHICS 8+16
20 CIRCLE 40,50,20
30 GOTO 30
RUN

Zie ook het voorbeeld bij FILLTO.

Omschrijving: Laadt een programma vanaf een cassette in het geheugen.

Versie : ATARI BASIC TURBO BASIC COMPILER
*** **

Formaat : CLOAD

Opmerking : Alleen voor cassette-gebruik.
Geeft een BEEP zodat PLAY toets kan worden ingedrukt.
Waarna op de RETURN toets moet worden gedrukt.

Omschrijving: Geeft de normale logaritme (grondgetal 10).

Versie : ATARI BASIC TURBO BASIC COMPILER
 *** *** ***

Formaat : A=COM
CLOG(x)

Opmerking : x moet een numerieke uitdrukking zijn groter dan 0.

Voorbeeld : Het eerste voorbeeld berekent de logaritme van 45:7
 READY
 PRINT CLOG(45/7)
 1.860752
 READY

Het tweede voorbeeld berekent de logaritmen van e en e^2:
READY
E=2.718282
READY
PRINT CLOG(E)
1
READY
PRINT CLOG(E*E)
2
READY



CLOSE

I/O instructie

=====
Omschrijving: Sluit het apparaat af na een Input- of Output
handeling en geeft het IOCB nummer vrij.

Versie : ATARI BASIC TURBO BASIC COMPILER
***1) ***1 en 2) ***

Formaat : 1)CLOSE # n of CL.
2)CLOSE

Opmerking : Mag ook worden gebruikt wanneer geen apparaat
werd geopend. De IOCB nummers zijn 1 t/m 7.
Zie ook OPEN.

ATARI BASIC altijd CLOSE # n
n moet een IOCB nummer zijn van 1 t/m 7.

TURBO BASIC XL 1.5
CLOSE # n of
CLOSE
dit is de verkorte vorm voor:
FOR I=1 TO 7:CLOSE # I:NEXT I

 Omschrijving: Zet alle numerieke variabelen op nul.
 Zet tevens alle gedimensioneerde arrays/
 strings op nul.

Versie : ATARI BASIC TURBO BASIC COMPILER
 *** *** ***

Formaat : CLR

Opmerking : Nadat CLR in een geprogramma is gebruikt
 moeten alle waarden weer opnieuw worden
 gedimensioneerd.

Voorbeeld : READY
 A=100
 READY
 PRINT A
 100
 READY

of

```

READY
10 A=100:B=10
20 C=A*B
30 PRINT A, B, C
40 CLR
50 PRINT A, B, C
RUN
  100    10    1000
   0     0     0
READY
  
```

De waarden die eerst in de regels 10 en 20
 aan A, B en C zijn toegekend zijn door de CLR
 opdracht in regel 40 teniet gedaan.

Omschrijving: Maakt het scherm schoon.

Versie : ATARI BASIC TURBO BASIC COMPILER
*** **

Formaat : CLS [#6]

Opmerking : #6 niet verplicht; wordt alleen
gebruikt bij een Graphicscherm.
CLS is gelijk aan ? CHR\$(125) of ? ")".
Na CLS wordt het scherm schoon gemaakt en de
cursor in de HOME positie (linksboven 0,0)
geplaatst.

=====
Omschrijving: Kiest een kleur uit het kleuregister in de
grafische modi 3-11.

Versie : ATARI BASIC TURBO BASIC COMPILER
*** *** ***

Formaat : COLOR n of C. n

Opmerking : Wordt veelal gebruikt om een te plotten punt/
lijn een bepaalde kleurwaarde toe te kennen.

Voorbeeld : READY
10 GRAPHICS 3+16
20 COLOR 2
30 PLOT 20,10:DRAWTO 20,20
40 GOTO 40

In de modi 0-2 wordt een ASCII teken gekozen,
waarvan de waarde n is voor de PLOT.



COM

Bewerkingsinstructie

=====

Omschrijving: Reserveert geheugenruimte voor strings en
numerieke reeksen.

Versie : ATARI BASIC TURBO BASIC COMPILER
*** *** ***

Formaat : COM var(subscript)[, var(subscript)]...

Opmerking : Gelijk aan DIM.

Voorbeeld : Zie DIM.

Omschrijving: Zet de uitvoering van het programma voort
nadat "BREAK" ingedrukt werd.

Versie : ATARI BASIC TURBO BASIC COMPILER
*** **

Formaat : CONT

Opmerking : Voortzetting van het programma na een STOP of
een END is eveneens mogelijk. Instructies op
de regel waar BREAK gedrukt is of STOP of END
vermeld staat en die daarna vermeld zijn
worden niet meer uitgevoerd na een CONT.
Het programma gaat verder op de eerstvolgende
regel.

=====
Omschrijving: Geeft de cosinus van een hoek.

Versie : ATARI BASIC TURBO BASIC COMPILER
 *** *** ***

Formaat : A=COS(x)

Opmerking : x is de hoekwaarde waarvan de cosinus van
 moet worden berekend.

Voorbeeld : READY
 10 PI=3.141593
 20 PRINT COS(PI)
 30 GRADEN=180
 40 RADIANT=GRADEN*PI/180
 50 PRINT COS(RADIANT)
 RUN
 -1
 -1
 READY

Dit voorbeeld geeft aan:

- 1) de cosinus van PI radiant= -1
- 2) zet dan de graden om in radiant en berekend de cosinus van 180 graden welke eveneens gelijk is aan -1. (180 graden=PI radiant)

Omschrijving: Slaat een BASIC programma uit het geheugen
op een cassette/tape op.

Versie : ATARI BASIC TURBO BASIC COMPILER
*** **

Formaat : CSAVE of CS.

Opmerking : Nadat CSAVE of SC. is ingetypt en op RETURN
is gedrukt, meldt de computer zich met 2
pieptontjes.
Druk de PLAY en RECORD toets in en druk op
RETURN.

Omschrijving: Geeft aan dat de lijst volgend na DATA informatie bevat welke met READ moeten worden gelezen.

Versie : ATARI BASIC TURBO BASIC COMPILER
 *** *** ***

Formaat : DATA of D. constant [,constant].....

Opmerking : De lijst volgende na het statement DATA kan zowel numerieke als string constanten bevatten.
De DATA statements worden niet door het programma uitgevoerd en kunnen dus op elke willekeurige plaats in het programma worden vermeld.
Het READ statement houdt een teller/wijzer bij waar het laatste DATA-constant is gelezen.
Met RESTORE (lineno) kan op elk willekeurige moment naar een nieuwe DATA regel worden gesprongen. Deze kan zowel voor als na de laatste gelezen DATA regel liggen.

Voorbeeld : Zie READ instructie.

Omschrijving: Geeft de decimale waarde van een hexadecimale string.

Versie : ATARI BASIC TURBO BASIC COMPILER
*** ***

Formaat : v-DEC(H\$)

Opmerking : H\$ is een hexadecimale waarde. Indien H\$ meer dan 4 posities lang is tellen de laatste vier posities mee.
 DEC is vergelijkbaar met VAL en is het tegen-gestelde van HEX\$.

Voorbeeld : READY
 5 DIM H\$(50)
 10 FOR N=50 TO 60
 20 ? HEX\$(N); " = "; N
 30 H\$(LEN(H\$)+1)=HEX\$(N)
 40 NEXT N
 50 ? H\$
 60 FOR X=1 TO LEN(H\$) STEP 2
 70 ? DEC(H\$(X, X+1)); " = "; H\$(X, X+1)
 80 NEXT X
 RUN

Het scherm zal dan na RUN het volgende resultaat opleveren:

```
32=50
33=51
34=52
35=53
36=54
37=55
38=56
39=57
3A=58
3B=59
3C=60
32333435363738393A3B3C
50=32
51=33
52=34
53=35
54=36
55=37
56=38
57=39
58=3A
59=3B
60=3C
READY
```



DEG

Trigonometrisch functie

Omschrijving: Na het DEG commando worden alle volgende trigonometrische functies in graden uitgedrukt.

Versie : ATARI BASIC TURBO BASIC COMPILER
*** **

Formaat : DEG

Opmerking : -

Voorbeeld : -



DEL
Systeem instructie

Omschrijving: Verwijdert programmaregels uit het geheugen.

Versie : ATARI BASIC TURBO BASIC COMPILER
*** **

Formaat : DEL van, tot

Opmerking : van - de beginregel van waaraf de programma-
regels moeten worden verwijderd uit het
programma.
tot - de eindregel welke als laatste moet
worden gewist.
Tevens zij opgemerkt dat DEL definitief is,
er bestaat geen mogelijkheid om de regels die
verwijdert zijn terug te halen in het
programma, tenzij ze nog op het scherm staan.
Dan dient men de cursor naar het regelnummer
te sturen welke men terug wenst te hebben en
op RETURN te drukken.

Voorbeeld : Dit voorbeeld wist slechts 1 regel;
n.l. regel 100
DEL 100,100
Het volgende voorbeeld verwijdert de regel 30
tot en met 100
DEL 30,100

=====

Omschrijving: Wist een file van diskette.

Versie : ATARI BASIC TURBO BASIC COMPILER
 *** ***

Formaat : DELETE "D[n]:filename.ext"

Opmerking : n is het drivenummer, niet verplicht indien
 drive 1 bedoeld wordt.
 filename.ext is de programma- of bestandsnaam
 van het te wissen programma.
 Wildcards zijn hier toegestaan.
 DELETE is gelijk aan XIO 33.

Voorbeeld : DELETE "D:*.BAS"
 verwijdert alle files met als extender BAS.

 Omschrijving: Reserveert geheugenruimte voor numerieke reeksen en strings.

Versie : ATARI BASIC TURBO BASIC COMPILER
 *** *** ***

Formaat : DIM var(subscript)[, var(subscript)]...

Opmerking : Elk string teken neemt een byte in beslag; elk onderdeel in een numerieke reeks neemt zes bytes geheugenruimte in beslag.

Voorbeeld : DIM A\$(10) = Een string met maximaal 10 tekens.
 DIM X(10) = een numerieke reeks X welke de elementen 0 t/m 10 (dus totaal 11) kan bevatten.
 DIM Y(20,20) = Een tweedimensionale reeks.
 DIM A\$(10),X(10) = Er kunnen verschillende variabelen met een komma van elkaar worden gescheiden achter een DIM instructie.

In TURBO BASIC XL 1.5 wordt de gedimensioneerde reeks eveneens gevuld met nul. Vooraf alle laadjes van de ladekast op nul zetten is derhalve niet meer nodig.
 DIM A(100) betekend dus:
 DIM A(100):FOR I=0 TO 100:A(I)=0:NEXT I

Omschrijving: Zet de inhoudsopgave van de diskette op het scherm.

Versie : ATARI BASIC TURBO BASIC COMPILER
 *** ***

Formaat : DIR ["Dn:*. *"]

Opmerking : DIR toont alle programma's / files van de diskette in drive 1 (1=default). Indien een ander dan drive 1 b.v. drive 2 gewenst is, dient dit opgegeven te worden als volgt:

DIR "D2:*. *"

Wildcards mogen hier worden gebruikt. Wenst men alle files te zien die als extender .BAS hebben, dan invoeren:

DIR "D:*.BAS"

Bij deze instructie zijn stringvariabelen eveneens toegestaan.

=====
Omschrijving: Geeft als uitkomst bij een deling als rest nihil.

Versie : ATARI BASIC TURBO BASIC COMPILER
*** **

Formaat : v= a DIV b

Opmerking : a en b zijn numerieke uitdrukkingen.
a DIV b geeft altijd een hele waarde als uitkomst. Is gelijk aan:

TRUNC(a/b)

Voorbeeld : ? 10.25 DIV 3.33 geeft 3
? 10 DIV 3 geeft 3
? 3 DIV 10 geeft 0



DO en LOOP
programma instructie

Omschrijving: Een eindeloze lus. Herhaalt steeds weer hetzelfde.

Versie : ATARI BASIC TURBO BASIC COMPILER
*** **

Formaat : DO ..opdracht .. LOOP

Opmerking : De ..opdracht .. tussen DO en LOOP wordt eindeloos herhaald. Uit deze lus kan men normaal niet komen of springen. Hier helpt alleen EXIT (zie aldaar)

Voorbeeld : READY
10 CLS
20 DO
30 PRINT "Hoofdmenu"
40 PRINT
50 PRINT "1 Afbeelden."
60 PRINT "2 Stoppen."
70 PRINT "Uw keuze graag";
80 REPEAT
90 GET KEY
100 UNTIL KEY>48 OR KEY<51
110 IF KEY=49 EXEC AFBEELDEN
120 IF KEY=50:EXIT
130 LOOP
140 END
.
.
.
200 PROC AFBEELDEN
.
.
.
299 ENDPROC

Bovenstaand geeft een gestructureerd keuzemenu weer. Hierin kunnen natuurlijk nog meer programmadelen in een PROCEDURE worden opgenomen.

Omschrijving: Roept het DOS menu op van diskette of vanuit de ramdisk.

Versie : ATARI BASIC TURBO BASIC COMPILER
*** ***

Formaat : DOS

Opmerking : Alleen te gebruiken met een diskdrive.
DOS = Disk Operating System.
Het omvat het besturingssysteem van de diskdrive.
Er zijn verschillende DOSversies uitgebracht.

DOS2.0 / DOS 2.5 / DOS 3.0 / DOS 4.0

Het DOS 2.0 en 2.5 zijn wel de meest gebruikte versies. Ze zijn gebruikersvriendelijk en gebruiken het minste geheugen.

=====

Omschrijving: Leest twee bytes uit twee geheugenadressen.
Dubbel-byte -Peek.

Versie : ATARI BASIC TURBO BASIC COMPILER
 *** ***

Formaat : A=DPEEK (adr)

Opmerking : adr - moet een integerwaarde zijn van het te
lezen geheugenadres tussen 0 en 65535.
De waarde die aan de variabele A wordt
toegekend is de berekende waarde van:

PEEK (adr)+256*PEEK (adr+1)

DPOKE is het tegengestelde van DPEEK (zie
aldaar). DPEEK heeft geen invloed op het te
lezen geheugenadres in tegenstelling tot
DPOKE.

Voorbeeld : READY
 A=DPEEK (560):PRINT A
 48160
 READY

Omschrijving: Schrijft twee bytes naar twee geheugenadres-
sen. Dubbel-byte-Poke.

Versie : ATARI BASIC TURBO BASIC COMPILER
*** **

Formaat : DPOKE adr, word

Opmerking : adr - moet een intergerwaarde zijn voor het
geheugenadres tussen 0 en 65535, waar
"word" naar toe moet worden geschreven.
Tevens wordt een waarde naar adr+1
geschreven.

word- een waarde of numerieke uitdrukking
welke naar het opgegeven adres (adr)
moet worden getransporteerd.

Voorbeeld : DPOKE 560,100
Hiermede wordt het volgende bedoeld:

POKE 560,100-256*INT(100/256):POKE 561,INT
(100/256)

of

POKE adr,word-256*INT(word/256):POKE adr+1
,INT(word/256)

DPOKE bespaart niet alleen geheugenruimte
maar ook schrijf of typwerk.
Zie ook DPEEK en POKE en PEEK.

Omschrijving: Tekent een lijn tussen 2 punten.

Versie : ATARI BASIC TURBO BASIC COMPILER
*** **

Formaat : DRAWTO x,y of DR. x,y

Opmerking : x en y :de coördinaten van de x-as en y-as
x is de positie en y is de regel
De lijn die wordt getrokken gaat uit van de
laatste positie van de cursor en de in x en y
gespecificeerde coördinaten.

Voorbeeld : READY
10 GRAPHICS 11
20 FOR Y=0 TO 191
30 C=Y/16
40 X=SQR(191-Y)*2
50 PLOT X,Y
60 DRAWTO X+7,Y
70 PLOT 36,Y
80 DRAWTO 43,Y
90 PLOT 72-X,Y
100 DRAWTO 79-X,Y
110 NEXT Y
120 GOTO 120

Omschrijving: Produceert geluid via de luidspreker van
TV of monitor.

Versie : ATARI BASIC TURBO BASIC COMPILER
*** **

Formaat : DSOUND voice, freq, dis, vol
DSOUND

Opmerking : Gelijk aan SOUND
voor beschrijving zie aldaar, waarbij zij op-
gemerkt dat bij DSOUND de berekende frequen-
tie in HERZ nu $1789790 / (2 * \text{freq} + 14)$ bedraagt.
De frequentie bedraagt dan 16 Bit
(0 65535).

Omschrijving: Toont de variabelen en procedures in een programma.

Versie : ATARI BASIC TURBO BASIC COMPILER

Formaat : DUMP [filespec]

Opmerking : filespec - is niet verplicht

Dit is het apparaat waar de lijst van variabelen en procedures naar toe moeten worden geschreven/gedumpt.
Als device/apparaat mag voor filespec de P voor Printer worden gebruikt.
De hele lijst wordt dan dus naar de printer gestuurd.

Voorbeeld : DUMP "P:"

Vergeet niet de ":" aanhalingstekens en dubbele punt.

Een dumplijst kan er als volgt uitzien:

```
A = 100 numerieke variabele
B(10,1 Array, DIM B(9) of DIM B(9,0)
C(0,0 ongedimensioneerde array.
      Bij arrays worden de beide dimensies
      die mogelijk zijn steeds met 1 ver-
      hoogd, getoond
D(10,10 DIM D(9,9)
E$ 10,20 String, LEN=10, DIM E$(20)
F$ 0,0 niet gedimensioneerde string
G$ 0,10 DIM G$(10), LEN(G$)=0
H PROC 100 Procedure H begint op regel 100
I # 120 Benoemde regel begint op 120
J ? Niet benoemde regel of procedure
```

De volgorde van de dump lijst staat in dezelfde volgorde zoals ze ook in de Variabelentabel staan.



ENDPROC
programma instructie

Omschrijving: Geeft het einde van een subroutine
(procedure) aan.

Versie : ATARI BASIC TURBO BASIC COMPILER
*** **

Formaat : ENDPROC

Opmerking : De instructie is vergelijkbaar met de
RETURN instructie bij een GOSUB ... RETURN
routine.

Zie voor nadere uitleg en voorbeeld onder
EXEC



ENTER

System instructie

Omschrijving: Voegt de regels uit een ATASCII file in een programma welke reeds in het geheugen is.

Versie : ATARI BASIC TURBO BASIC COMPILER
*** **

Formaat : ENTER "dev[devnr]:filename.ext"
of
E. enz...

Opmerking : dev = device C voor Cassette
D voor Diskdrive

devnr = 1 t/m 8 (drivenummer)

: moet worden vermeld.

filename.ext = programmaam + extender
zoals deze met LIST is weggeschreven.
Wees voorzichtig met het mengen van program-
maregels in een bestaand programma. Regels
die hetzelfde regelnummer hebben worden in
een programma, welke reeds in het geheugen
aanwezig is, overschreven/vervangen.

Voorbeeld : ENTER "C:"

ENTER "D2:MYPROG.LST"


```

300 PROC DISPLAY
310 DL=DPEEK(550)
320 POKE DL+3,70:POKE DL+6,7
330 ? #6;"by MIJNNAAM (c) 1987";
340 ? #6;" Turbo mEnU "
350 ENDPROC

```

In regel 10 wordt het scherm schoon gemaakt.
 Regel 20 en 30 roepen de procedure INIT
 (initialisatie) en DISPLAY (schermopbouw) op.
 Regel 40 ... tot 100 volgt de uitvoering van
 het hoofdprogramma.

Regel 200 tot 230 de procedure INIT wordt
 uitgevoerd, nadat deze is opgeroepen met
 EXEC (=execute) INIT.

Regel 300 tot 350 de procedure DISPLAY.
 Volgens bovenstaande constructie is uw pro-
 gramma gestructureerd opgebouwd.

Zie ook ON en EXEC

Omschrijving: Berekend het exponent van een numerieke
variabele.

Versie : ATARI BASIC TURBO BASIC COMPILER
*** **

Formaat : A=EXP(x)

Opmerking : x mag elke willekeurige numerieke uitdruk-
king zijn.
Het geeft de waarde van e (ongeveer 2.718)
tot de aangegeven macht. In sommige gevallen
is EXP alleen tot 6 cyfers nauwkeurig. EXP
is het tegengestelde van LOG.

Voorbeeld : READY
10 X=2
20 PRINT EXP(X-1)
RUN
2.718282

Omschrijving: Lustester.

Versie : ATARI BASIC TURBO BASIC COMPILER
*** **

Formaat : *F[teken]

Opmerking : teken - niet verplicht. Tekenen kan zijn + of -
*F of *F+

Na deze opdracht wordt bij een FOR .. NEXT lus eerst getest of de teller de eindwaarde reeds bereikt heeft, voordat de eigenlijke lus wordt uitgevoerd.

Is dit zo dan wordt de lus tot NEXT overgeslagen en volgt geen uitvoering van de lus.

*F-

herstelt de normale toestand. FOR .. NEXT lussen worden dan minstens een keer doorlopen. Ook bij RUN wordt door het systeem automatisch een *F- uitgevoerd.

Voorbeeld : 10 *F+
20 FOR X=3 TO 1
30 PRINT X
40 NEXT X
50 END

De beginwaarde van de lus is 3; de lusteller wordt vervolgens met 1 verhoogd. Deze waarde wordt vergeleken met de eindwaarde welke 1 is. De lus wordt verlaten omdat de voorwaarde waar (TRUE) is.

Na *F of *F+ wordt eerst de variabele X met de beginwaarde 3 geladen en dan met de variabele X (eindwaarde=1) vergeleken.

De lus wordt tot NEXT X overgeslagen.

Er vindt geen uitvoer naar het scherm plaats.

Omschrijving: Kiest een kleur uit het Kleurregister.

Versie : ATARI BASIC TURBO BASIC COMPILER
*** **

Formaat : FCOLOR n

Opmerking : n is de kleurcode uit het Kleurregister.
Deze moet van een integerwaarde zijn en mag
de waarde 0,1,2 of 3 krijgen.

FCOLOR kan alleen gebruikt worden bij het
commando FILLTO.

FCOLOR geeft dus de kleurwaarde voor het in
te vullen gebied.

FCOLOR is gelijk aan POKE 765,n

Voorbeeld : Zie voorbeeld bij FILLTO.

Omschrijving: Vult een begrensd gebied in met een gekozen
Kleur.

Versie : ATARI BASIC TURBO BASIC COMPILER
 *** ***

Formaat : FILLTO x,y

Opmerking : x = x coördinaat op het grafische
beeldscherm (positie).
y = y coördinaat (regel).

Het FILLTO commando is gelijk aan:

POS. x,y:XIO 18,#6,0,0,"S:":POKE 765,color

Om een gebied te kleuren moet men dit eerst
begrenzen. De begrenzing moet als volgt
geschieden:

PLOT een punt rechtsboven

DRAWTO ergens rechtsboven

DRAWTO ergens linksonder

FILLTO ergens tot linksonder.

en geef dit de kleur met FCOLOR (zie aldaar).

Het bezwaarlijke aan FILLTO of XIO 18 is dat
de linker-begrenzing steeds een rechte lijn
moet zijn anders werkt FILLTO en XIO 18 niet
altijd correct.

Zie in dit verband ook PAINT.

Voorbeeld : 10 GRAPHICS 24
20 SETCOLOR 2,10,0
30 SETCOLOR 1,6,10:COLOR 1
40 PLOT 250,170:REM RECHTSBOVEN
50 DRAWTO 175,10:REM RECHTSBOVEN
60 DRAWTO 125,10:REM LINKSBOVEN
70 FILLTO 50,70:REM LINKSONDER
80 FCOLOR 1
90 TEXT 110,100,"TURBO BASIC"
100 CIRCLE 40,80,30
110 CIRCLE 270,80,30
120 PAINT 40,80:PAINT 270,80
130 GOTO 130

Omschrijving: Herhaalt een serie opdrachten binnen een lus met een bepaalde op te geven tijdsperiode.

Versie : ATARI BASIC TURBO BASIC COMPILER
*** **

Formaat : FOR var=x TO y [STEP z]

.
NEXT var [,var]...

ook

F.var= enz. enz.

.
N.var enz.

Opmerking : var is een integerwaarde welke wordt gebruikt als teller.
x is een numerieke uitdrukking welke de beginwaarde geeft aan de teller var.
y is een numerieke uitdrukking welke de eindwaarde aangeeft.
z is de numerieke uitdrukking die de stapgrootte aangeeft.
STEP z is niet verplicht. Zonder STEP z is de stapgrootte 1 (=default).
Om een goed verloop van een programma te waarborgen mag slechts uit een lus worden gesprongen via de POP-instructie in Atari Basic en via de POP of EXIT-instructie in Turbo Basic.

Voorbeeld : FOR TELLER=1 TO 10

.
NEXT TELLER
Teller zal bij 1 beginnen, telkens met 1 worden verhoogd en eindigen bij 10.
FOR A=10 TO 1 STEP -2
A begint bij 10 en wordt bij iedere doorloop met 2 verminderd.
FOR A=QIN TO P*S STEP Z
Een berekende waarde voor x, y en z zijn ook toegestaan.

READY
10 FOR X=0 TO 255
20 POKE 712,X
30 FOR WACHT=1 TO 25:NEXT WACHT
40 NEXT X

Dit programmaatje wisselt de kleur voor de schermborder en laat alle kleuren zien die de Atari computer rijk is.

=====

Omschrijving: Spoort het aantal cijfers achter de komma op.

Versie : ATARI BASIC TURBO BASIC COMPILER
*** **

Formaat : v=FRAC(exp)

Opmerking : exp = een numerieke uitdrukking.

FRAC(exp) is niet altijd vergelijkbaar met:
exp-INT(exp).
Dit omdat INT altijd de naast kleinere
waarde/getal geeft.

Voorbeeld : INT(-0.3) geeft -1 en
FRAC(-0.3) geeft -0.3

Zie ook TRUNC.

Omschrijving: Geeft weer hoeveel ongebruikte geheugenruimte
nog vrij is.

Versie : ATARI BASIC TURBO BASIC COMPILER
*** **

Formaat : FRE(0)
var=FRE(0)

Opmerking : (0) is een verplichte dummy variabele.
De hoeveelheid ongebruikte geheugenruimte
betreft het RAM gedeelte.

Voorbeeld : READY
PRINT FRE(0)
32274 in Atari Basic
34017 in Turbo Basic XL 1.5
READY

=====

Omschrijving: Neemt een enkele byte (teken) uit het gespecificeerde apparaat (device) en zet het in de aangegeven variabele.

Versie : ATARI BASIC TURBO BASIC COMPILER
 ***1) ***1)2) ***

Formaat : 1) GET #n, key
 2) GET key

Opmerking : Het programma stopt zodra het een GET instructie tegenkomt en wacht tot een toets is ingedrukt. Kijkt dan naar de ATASCII waarde en kent deze toe aan de variabele key.
 n = een integer nummer van 1 t/m 7

Voorbeeld : In ATARI BASIC :
 10 OPEN #1,4,0,"K:"
 20 GET #1,A
 30 PRINT A
 40 CLOSE #1
 50 GOTO 10

of lezen van disk of cassette:
 10 OPEN #1,4,0,"D:BESTAND.DAT"
 20 GET #1,A
 30 FOR I=1 TO A
 40 GET #1,B
 50 PRINT B
 60 NEXT I
 70 CLOSE #1

In TURBO BASIC XL 1.5:
 GET KEY is gelijk aan:
 OPEN #1,4,0,"K:":GET #1,KEY:CLOSE #1
 10 GET KEY
 20 PRINT KEY
 30 GOTO 10

Dit voorbeeldje is hetzelfde als het eerste voorbeeld in Atari Basic.
Er kan dus met recht worden gesteld dat dit beduidend korter is.

Werken met bestanden in Turbo Basic zie ook %GET en BGET.

Omschrijving: Springt naar een subroutine, voert deze uit
en springt terug naar de opdracht na GOSUB.

Versie : ATARI BASIC TURBO BASIC COMPILER
 *** *** ***

Formaat : GOSUB line of: GOS. line
 .
 RETURN RET.

Opmerking : line - het regelnummer van de eerste regel
van de subroutine.
De uitvoering van een programma in 't interne
geheugen (RAM) wordt op de aangegeven regel
vervolgd tot de eerste RETURN instructie.
Deze instructie laat de uitvoering teruggaan
naar de instructie die direct volgt op de
GOSUB instructie.
line mag ook een berekende waarde hebben, zo-
als GOSUB X+10

Voorbeeld :READY
 10 INPUT A
 20 PRINT A
 30 GOSUB 100
 40 PRINT A,B
 50 END
 99 REM *** SUBROUTINE BEREKENING ***
 100 B=A+10
 110 RETURN

Zie ook EXEC-PROC-ENDPROC

Omschrijving: Springt naar een opgegeven programmaregel.

Versie : ATARI BASIC TURBO BASIC COMPILER
 *** *** ***

Formaat : GOTO line

Opmerking : line - is een willekeurig regelnummer of
 berekende waarde waar de programmauitvoering
 moet worden voortgezet.

Voorbeeld 1 : 10 INPUT X
 20 IF X<100 THEN GOTO 500
 30 GOTO 10
 500 PRINT "X is kleiner dan 100 !"
 510 PRINT "X is n.l. : ";X
 520 GOTO 10

Voorbeeld 2 : 10 X=RND(0)*10+1
 20 IF X<5 THEN 10
 30 PRINT X
 40 GOTO 10

Zie ook GO# name

Omschrijving: Zet een integer waarde om in een hexadecimale string.

Versie : ATARI BASIC TURBO BASIC COMPILER
 *** ***

Formaat : v\$=HEX\$(n)

Opmerking : n - is een numerieke uitdrukking welke moet liggen tussen 0 en 65535. Tevens moet de waarde n een integer waarde vertegenwoordigen (heel getal). Indien n kleiner is dan 256 dan is de uitkomst een string met twee plaatsen, anders is de string vier plaatsen groot. HEX\$ is vergelijkbaar met STR\$

Voorbeeld : READY
 10 FOR N=250 TO 260
 20 PRINT HEX\$(N);" = ";N
 30 NEXT N
 RUN
 FA = 250
 FB = 251
 FC = 252
 FD = 253
 FE = 254
 FF = 255
 0100 = 256
 0101 = 257
 0102 = 258
 0103 = 259
 0104 = 260

Zie in dit verband ook DEC.

 Omschrijving: Vergelijking van twee waarden opgegeven na IF. Is de vergelijking waar dan wordt opdracht na THEN uitgevoerd.

Versie : ATARI BASIC TURBO BASIC COMPILER
 *** *** ***

Formaat : IF var [vergelijking var][logische operator] [var] THEN instructie

Opmerking : var - mag elke numerieke of stringbewerking zijn.
 vergelijking - kan zijn:
 = is gelijk
 <> is ongelijk
 < kleiner dan
 > groter dan
 <= kleiner of gelijk dan
 >= groter of gelijk dan

logische operator - kan zijn:
 AND logisch AND
 OR " OR
 NOT " ontkenning

instructie - mag elke opdracht zijn.

Voorbeeld : IF X=Y THEN PRINT "X IS GELIJK AAN Y"
 IF X<>Y THEN PRINT "X IS NIET GELIJK AAN Y"
 IF X<Y THEN GOTO 100
 IF X>Y THEN GOTO 300
 IF X<=Y THEN GOSUB 1000
 IF X>=Y THEN END
 IF X=Y AND A=B THEN SOUND 1,121,10,10
 IF X=Y OR A=B THEN GRAPHICS 0
 IF X THEN PRINT "X IS ONGELIJK NUL"

Omschrijving: Vergelijking van meerdere variabelen
zodat of opdracht 1 of opdracht 2 kan worden
uitgevoerd.

Versie : ATARI BASIC TURBO BASIC COMPILER
*** **

Formaat : IF aexp opdr1 [ELSE opdr2] ENDIF

Opmerking : Als de vergelijking 'aexp' na IF waar is
wordt opdracht 1 (opdr1) uitgevoerd, anders
wordt opdracht 2 (opdr2) uitgevoerd.
Ter ondersteuning van 'aexp' en opdracht
staat niet THEN doch een dubbele punt (:)
of een <RETURN> regeleinde.
Na ELSE mag geen sprongopdracht GOTO worden
gebruikt.
ENDIF dient om het einde van de 'lus' of
opdracht(en) aan te geven. Het is niet nodig
om IF ... ELSE ... ENDIF op een regel te
schrijven.

Voorbeeld : 10 INPUT "Geef een getal ";A
20 IF A<10 vergelijking
30 PRINT "A is < 10; A = ";A opdr1
40 ELSE
50 PRINT "A is > 10; A = ";A opdr2
60 ENDIF
70 GOTO 10

Omschrijving: Leest een karakter (teken) welke door het
toetsenbord wordt gegeven als een toets wordt
gedrukt.

Versie : ATARI BASIC TURBO BASIC COMPILER
*** **

Formaat : K\$=INKEY\$

Opmerking : K\$ - een stringvariabele waar het teken in
wordt gezet.
Wordt geen toets ingedrukt dan bevat K\$
een lege string "".
Zo kan het indrukken van een toets wor-
den verwerkt zonder dat de uitvoering
van het programma wordt onderbroken.

Omschrijving: Zet de inhoud uit een apparaat in een numerieke of string variabele.

Versie : ATARI BASIC TURBO BASIC COMPILER
 *** *** ***

Formaat : ATARI en TURBO BASIC:
 INPUT [# Kanaal] var [,var]

TURBO BASIC:
 INPUT ["tekst"],var [,var] ...
 INPUT ["tekst"];var [,var] ...
 of voor INPUT alleen I.

Opmerking : De inhoud cq regel tekens moet met een <RETURN> teken of CHR\$(13) worden afgesloten.
Kanaal - een IOCB Kanaal/bestand welke eerst met de OPEN instructie moet zijn geopend.
Kanaal is een nummer van 1 t/m 7

var - elke numerieke of string variabele.

TURBO BASIC XL 1.5:

Deze INPUT-instructie benadert sterk het Microsoft Basic. Het is mogelijk om een tekst na INPUT te gebruiken. Word na de tekst een komma gezet dan volgt het ?
Wil men het soms wel storende '?' niet op het scherm hebben moet men in plaats van een komma en punt komma ';' zetten.

Voorbeeld : INPUT A (Vraagt een numerieke waarde en zet die in variabele A)
 INPUT X,Y,Z (Vraagt 3 getallen, gescheiden door komma's en zet deze in X, Y en Z)
 INPUT A\$ (Vraagt een reeks tekens en zet deze in A\$ exclusief het <RETURN> teken; A\$ moet vooraf worden gedimensioneerd; zie DIM)

```
10 OPEN #1,4,0,"D:MYPROG.DAT"
20 INPUT #1,A$,A
30 CLOSE #1
```

Haal een reeks tekens uit een bestand MYPROG.DAT, welke zich op de diskette bevindt en zet deze in de variabelen A\$ en A.

Voor TURBO BASIC:
 INPUT "Geef een getal ",A
 INPUT "Geef uw naam ";NS
 INPUT "";A

De afkorting I. mag ook hier worden gebruikt.

Omschrijving: Geeft het grootste gehele getal (integer) kleiner dan of gelijk aan de ingevoerde waarde.

Versie : ATARI BASIC TURBO BASIC COMPILER
 *** *** ***

Formaat : v=INT(x)

Opmerking : v - een numerieke variabele
 x - mag elke numerieke waarde van een numerieke bewerking zijn.
 Tevens mogen in de bewerking andere rekenkundige functies voorkomen.

Voorbeeld : READY
 10 X=2.3456
 20 A=INT(X)
 30 PRINT "A = ";A
 RUN
 A = 2
 READY

 A=INT((RND(0)*10)+0.05)/10

Omschrijving: Tabulatorinstelling voor de LIST-functie.

Versie : ATARI BASIC TURBO BASIC COMPILER
*** **

Formaat : *L[teken]

Opmerking : Tekens niet verplicht. Tekens kan zijn:
+ of -

*L of *L+

Na deze opdracht wordt de tabulator (2 spaties) weer ingeschakeld. Dit is ook de normale stand na het opstarten van TURBO BASIC. Er wordt 2 posities bij lussen ingesprongen.

*L-

Schakelt de tabulator voor de LIST-functie uit, zodat er, indien noodzakelijk, meer ruimte ontstaat. Dit kan nodig zijn bij het editeren van lange programmaregels of om ruimte te besparen bij het opslaan op de diskette (LIST "D:MYFILE.LST).

Het REM teken -- wordt na *L- slechts tweevoudig in plaats van dertigvoudig afgedrukt.

 Omschrijving: Bepaalt de lengte van een stringvariabele en zet deze in een numerieke variabele.

Versie : ATARI BASIC TURBO BASIC COMPILER
 *** *** ***

Formaat : x=LEN(A\$)
 x=LEN("STRING EXPRESSION")

Opmerking : Het aantal tekens wordt bepaald, welke zich in een stringvariabele bevinden; ongeacht welk teken of spatie wordt opgegeven.

Voorbeeld : READY
 10 DIM A\$(100)
 20 A\$="DIT IS EEN STRING"
 30 A=LEN(A\$)
 40 PRINT A
 RUN
 17
 READY

Op deze manier kan de lengte bepalend zijn voor de uitkomst van een berekening. Zie voorbeeld als boven en vervolg deze met:

```
50 B=A*LEN(A$)
60 PRINT B
RUN
289
READY
```

Omschrijving: Kent een waarde toe aan numerieke of string
variabelen.

Versie : ATARI BASIC TURBO BASIC COMPILER
*** **

Formaat : [LET] var[\$]=["]waarde["]

Opmerking : Gebruik van de instructie LET is niet
verplicht.

Voorbeeld : LET A=B

De waarde van B wordt aan A toegekend.

LET A\$="JAN KLAASSEN"

of:

A=B
A\$="JAN KLAASSEN"

Omschrijving: Toont een programma op het beeldscherm of
schrijft het naar een randapparaat (device).

Versie : ATARI BASIC TURBO BASIC COMPILER
*** *** ***

Formaat : LIST [device n:][filespec][, line][, line]
of in plaats van LIST alleen L.

Opmerking : Er is een verschil tussen het "LISTEN" van
programmaregels op het beeldscherm in ATARI
BASIC en TURBO BASIC XL 1.5

Bij LIST in TURBO BASIC worden de lussen
telkens 2 posities ingesprongen, wat de lees-
baarheid ten goede komt. Tevens kan de compu-
ter medegedeeld worden om vanaf een bepaald
regelnummer tot het einde van het programma
alle regelnummers te listen door een komma
[,] achter het begin regelnummer te zetten:
LIST 3000,

Voorbeeld : LIST Schrijft het gehele program-
ma naar het beeldscherm.
LIST 10 toont regel 10.
LIST 10,100 toont regel 10 t/m 100.
LIST "P:" print gehele programma uit
op de printer.
LIST "P:",10,100 Print regel 10 t/m 100 op de
printer.
LIST 3000, toont alle regels vanaf 3000
tot het einde.
LIST "D:MYPROG.LST"
Stuurt het programma naar
de diskette in de ATASCII
form
Terughalen met ENTER.
Het programma kan nu zelfs
met een tekstverwerker wor-
den bekeken.

LIST "D2:MYPROG.LST",10,100
Zie boven. Nu worden slechts
de regels 10 t/m 100 naar de
diskette geschreven in drive
station 2.
LIST "C:" idem doch naar cassette.

Zie ook *L

Omschrijving: Haalt een programma van een extern apparaat naar het geheugen (RAM).

Versie : ATARI BASIC TURBO BASIC COMPILER
 *** *** ***

Formaat : LOAD "device [n]:"[filespec]
 of in plaats van LOAD alleen LO.

Opmerking : device - moet een gegevensapparaat zijn
 zoals Diskdrive, Cassette of
 Ramdisk.

n - drivenummer 1 t/m 8

filespec-een willekeurige programmaam
bestaande uit max. 8 + 3 tekens
gescheiden door een punt [.]

Voorbeeld : LOAD "C:"
 LOAD "D:MYPROG.BAS"
 LOAD "D8:TEST"

Omschrijving: Bekijkt de data van een aangegeven scherm-locatie.

Versie : ATARI BASIC TURBO BASIC COMPILER
 *** *** ***

Formaat : LOCATE x, y, d
 of voor LOCATE alleen LOC.

Opmerking : Haalt tekens op in modi 0-2 en color-nummers in de modi 3-11.

Voorbeeld : LOCATE 100,100,D

Gekeken wordt naar de schermlocatie met de x en y. De gevonden waarde wordt dan toege-
kend aan de variabele 'D'.
In de modi 0-2 geeft de LOCATE instructie de ATASCII waarde op.
In de modi 3-11 wordt de COLOR-waarde in de variabele 'D' gezet.
Voor x en y mogen ook berekende waarden worden gebruikt.

LOCATE PEEK (86) *256+PEEK (85), PEEK (84), D

De LOCATE instructie wordt ook gebruikt om een zogenaamde "botsing" (collision) te ontdekken in combinatie met Player Missile.

Omschrijving: Geeft de natuurlijke logaritme (grondtal e).

Versie : ATARI BASIC TURBO BASIC COMPILER
 *** *** ***

Formaat : v=LOG(x)

Opmerking : Dit is de tegengestelde functie van EXP.

x - is een numerieke waarde/getal groter dan nul of een berekende waarde.

Voorbeeld : 10 X=1.753:P=3
 20 A=P*LOG((INT(X*10)+.5)/10)
 30 PRINT A

Omschrijving: Schrijft gegevens naar de printer.

Versie : ATARI BASIC TURBO BASIC COMPILER
 *** *** ***

Formaat : LPRINT var[,var][,var]....
 of alleen LP.

Opmerking : Bij de LPRINT instructie is het niet nodig
 een kanaal naar de printer te openen en te
 sluiten met OPEN en CLOSE.

Voorbeeld : LPRINT A\$
 LPRINT A;A\$
 LPRINT B,X\$
 LPRINT "Dit is een test."
 LP. "LP. of LPRINT is hetzelfde"

Omschrijving: Blocktransfer.

Versie : ATARI BASIC TURBO BASIC COMPILER
*** **

Formaat : [-]MOVE source, dest, count

Opmerking : source - is de eerste te lezen geheugenadres
welke moet worden verhuist.

dest - is de bestemming waar heen de inhoud
van 'source' moet worden gecopieerd.

count - het aantal te copieren geheugen-
adressen vanaf 'source'.

De vertaling naar Atari Basic kan als volgt
geschreven worden:

```
FOR I=0 TO COUNT-1
  POKE dest+I, PEEK(source+I)
NEXT I
```

Indien voor de instructie MOVE een - minteken
wordt geplaatst vindt uitvoering in omgekeer-
de volgorde plaats. Eerst het laatste byte,
enz. enz....

Dit maakt het mogelijk informatie naar een
hoger adres te verschuiven, zonder dat er
bij overlapping (indien: source+count > dest)
iets gewist wordt.

De instructie wordt dan:

```
-MOVE source, dest, count
```

Hetgeen betekend:

```
FOR I=count-1 TO 0 STEP -1
  POKE dest+I, PEEK(source+I)
NEXT I
```

name
speciale functie

=====
Omschrijving: Speciaal te benoemen regel.

Versie : ATARI BASIC TURBO BASIC COMPILER
*** **

Formaat : # name

Opmerking : # - verplicht teken.
name - een willekeurige stringuitdrukking.
Deze functie dient als herkenning-
adresregel voor de sprongopdrachten:

GO# name
ON arg GO# name,name ...
TRAP# name
RESTORE# name

Zie aldaar

Omschrijving: Wist het programma uit het geheugen (RAM) en
zet alle variabelen op nul.

Versie : ATARI BASIC TURBO BASIC COMPILER
*** **

Formaat : NEW

Opmerking : Alle variabelen worden op nul gezet. Tevens
wordt het geluid uitgezet.
Kan zowel in direct mode als binnen een pro-
gramma worden gebruikt.

Voorbeeld : READY
NEW
READY

Het programma welke in het geheugen was is nu
gewist.

Omschrijving: Beeindiging van een FOR NEXT lus.

Versie : ATARI BASIC TURBO BASIC COMPILER
 *** *** ***

Formaat : FOR var=var TO var
 .
 NEXT var

Opmerking : Zie bij FOR var
 NEXT mag ook als N. worden geschreven.

Omschrijving: Bepaalt de plaats van het volgende te lezen
of te schrijven byte op de diskette.

Versie : ATARI BASIC TURBO BASIC COMPILER
 *** *** ***

Formaat : NOTE #n, sec, byte

 of in plaats van NOTE alleen NO.

Opmerking : # - is verplicht.
 n - is een kanaalnummer van 1 t/m 8
 sec is een variabele waar het sectornummer
 wordt ingezet en
 byte is de variabele waar het bytenummer
 wordt ingezet, welke gelezen of geschre-
 ven wordt.



... GOSUB en ON ... GOTO
programma instructie

Omschrijving: Springt naar een van de gespecificeerde regelnummers afhankelijk van de waarde van een uitdrukking.

Versie : ATARI BASIC TURBO BASIC COMPILER
*** **

Formaat : ON n GOTO line[,line]...
ON n GOSUB line[,line]...

Opmerking : n - is een numerieke uitdrukking. Het moet een integerwaarde hebben tussen 0 en 256
line is het regelnummer welke men wenst aan te springen als aan de gewenste uitdrukking is voldaan.

Is de uitdrukking 'n' niet waar wordt het programma op de volgorde regel voortgezet. ON .. GOTO of ON .. GOSUB is een ideale manier om via een keuze-menu naar een routine te springen en weer terug.

Voorbeeld : In TURBO BASIC XL 1.5:
100 GET KEY
110 IF KEY <48 OR KEY >52 THEN 100
120 ON KEY-48 GOSUB 1000,2000,3000,4000
130 GOT0 10
1000 REM START SUBROUTINE ALS KEY-48=1
:
:
1990 RETURN
:
:

Omschrijving: Staat een kanaal/apparaat toe dat er naar geschreven en/of gelezen wordt (input/output).

Versie : ATARI BASIC TURBO BASIC COMPILER
 *** *** ***

Formaat : OPEN #iocb,bew,0,"dev[n]:[filespec]"
 of voor OPEN allen 0.

Opmerking : # - verplicht teken.

iocb - kanaalnummer 1 t/m 7; welke men wenst te openen.
IOCB: In/Out Control Block
dit zijn de poorten waardoor gegevensstroom kan worden verstuurd.

bew - bewerkingscode. Dit kan zijn:
4 INPUT - alleen lezen.
6 DIRECTORY - inhoudsopgave diskette.
8 OUTPUT - alleen schrijven.
9 APPEND - toevoegen aan het einde van een bestand (sequentiele opbouw, chaotisch)
12 UPDATE - lezen en schrijven, mutatie van een bestand.

0 - verplicht teken.

dev - device/apparaat welke men wenst aan te spreken / openen. Toegestaan is:
K - Keyboard
P - printer
C - cassette recorder
D - diskdrive
E - editor
R - RS-232 seriële poort
S - beeldscherm

n - is devicenummer / apparaatnummer is niet verplicht; default = 1 indien niet gespecificeerd. Toegestaan is 1 t/m 8. Wordt alleen gebruikt bij D(rive) en R(S-232).
Indien R wordt gebruikt dient een seriële poort-programma geladen te worden voordat de seriële poort wordt geopend.

filespec - opgave bij D(rive) van de filenaam + extender van het te openen diskettebestand.

":" - verplichte tekens



Voorbeeld : Inputbewerking

```
10 Open #1, 4, 0, "K:" REM open toetsenbord
20 GET #1, K REM wacht op input via
toetsenbord
30 CLOSE #1 REM sluit toetsenbord
40 PRINT K REM ATASCII-nummer
afdrukken.
```

Outputbewerking

```
10 OPEN #1, 8, 0, "D:MYPROG.DAT" REM openen
bestand
20 FOR I=1 TO 100 REM lus van 1 tot 100
30 PRINT #1;I REM schrijf naar bestand
40 NEXT I REM 1 tot 100
50 CLOSE #1 REM sluit bestand
```

Inputbewerking in TURBO BASIC:

```
10 GET KEY
20 PRINT KEY
```

Dit heeft hetzelfde gevolg als bovenstaand voorbeeld van Inputbewerking.

Via Drive:

```
10 OPEN #1, 4, 0, "D:MYPROG.DAT"
20 FOR I=1 TO 100
30 INPUT #1, X
40 PRINT X;
50 NEXT I
60 CLOSE #1
```

Haal de getallen 1 t/m 100 van het vorige voorbeeld terug van de diskette en print deze achterelkaar op het scherm. Voeg de volgende regels toe en dan worden getallen naar de printer gestuurd.

```
15 OPEN #2, 8, 0, "P:"
45 PRINT #2; X
60 CLOSE
```

Directory van de diskette:

```
10 OPEN #1, 6, 0, "D:*. *": REM open directory
20 TRAP 60 REM fout gevonden? Ga
naar regel 60
30 INPUT #1, FILE$ REM haal een string op
40 PRINT FILE$ REM print die string
50 GOTO 30 REM Ga weer een string
halen enz...
60 CLOSE #1 REM sluit directory
```

In TURBO BASIC kan het bovenstaande programmaatje worden vervangen door het commando DIR *. * in regel 10 is een zogenaamde wildcard. Deze mag ook vervangen worden als men gericht wil zoeken naar programmanamen met gelijke waarden.

b. v. : *.BAS of MY*. *

Omschrijving: Geeft de waarde van een extern apparaat
(Paddle) op.

Versie : ATARI BASIC TURBO BASIC COMPILER
*** **

Formaat : x=PADDLE(n)

Opmerking : x - een numerieke variabele.
n - Paddle-nummer. Toegestaan is 0 t/m 3.
De Waarde die in x wordt gezet ligt tus-
sen 1 en 228.
Een linksdraaiende beweging (dus tegen de
klokrichting in) geeft een oplopende
waarde.

Voorbeeld : IF PADDLE(0)>14 THEN 500
De Paddle controllers worden niet zoveel ge-
bruikt. De joystick is bij de spelletjes-
spelers meer geliefd.

Omschrijving: Onderbreking van de programmauitvoering gedurende n/50 seconden.

Versie : ATARI BASIC TURBO BASIC COMPILER
*** **

Formaat : PAUSE n

Opmerking : n is de tijdseenheid waarmee een programma onderbroken wordt, voordat verder wordt gegaan.
PAUSE vervangt hierdoor derhalve de geheugen vretende FOR - NEXT lussen zonder inhoud.

Voorbeeld : PAUSE 100

Twee seconden onderbreking.

Omschrijving: Geeft het getal dat op een gespecificeerd geheugenadres is opgeslagen.

Versie	: ATARI BASIC	TURBO BASIC	COMPILER
	***	***	***

Formaat : A=PEEK(n)

Opmerking : n - is een integerwaarde tussen 0 en 65535.
n geeft het te lezen geheugenadres aan.
De waarde die aan A wordt toegekend is eveneens een integerwaarde, welke ligt tussen 0 en 255.
PEEK is het tegengestelde van POKE (zie aldaar).
PEEK heeft geen invloed op het te lezen geheugenadres in tegenstelling tot POKE

Voorbeeld : Met het volgende programmaatje kunnen alle geheugenadressen bekeken worden.

```
10 FOR N=1 TO 65535
20 PRINT N;" ";PEEK(N)
30 NEXT N
```

De waarde kan eveneens gebruikt worden in berekeningen.

```
A=PEEK(560)+256*PEEK(561)
```

Zie ook DPEEK.

Omschrijving: Plaatst een punt of teken op een opgegeven plaats op het beeldscherm.

Versie : ATARI BASIC TURBO BASIC COMPILER
 *** *** ***

Formaat : PLOT x, y
 of
 PL. x, y

Opmerking : x en y zijn de waarden / coördinaten van de te plotten beeldschermpositie. Deze zijn sterk afhankelijk van de Graphic modus welke men kiest.

Voorbeeld : Met het volgende voorbeeld worden in Graphics modus 5 op willekeurige plaatsen punten op het scherm neergezet (geplot).

```

5 GRAPHICS 5+16 :REM GRAPHICS 5 zonder
                  tekstvenster.
10 FOR I=1 TO 25 :REM 25 punten
20  X=INT(RND(0)*80)+1:REM willekeurige
30  Y=INT(RND(0)*48)+1:REM plaatsen x,y
40  PLOT X,Y:REM zet een punt op het scherm
50 NEXT I
60 GOTO 60:REM wacht op BREAK

```

Hetzelfde korter in TURBO BASIC XL 1.5;
vervang regel 20 en 30 door:

```

20 X=RAND(80)
30 Y=RAND(48)

```

Omschrijving: Verwijst het Disk Operating System waar volgende te lezen of schrijven byte zich bevindt

Versie : ATARI BASIC TURBO BASIC COMPILER
 *** *** ***

Formaat : POINT #n, sec, byte
 of in plaats van POINT alleen P.

Opmerking : n - is een kanaalnummer 1 t/m 7, welke
 vooraf dient te worden geopend.
 sec - is het betreffende sectornummer en
 byte- is het betreffende bytenummer op
 de diskette.

Voorbeeld : -

Omschrijving: Schrijft een waarde in een geheugenadres.

Versie : ATARI BASIC TURBO BASIC COMPILER
*** **

Formaat : POKE n, m

Opmerking : n - moet een integerwaarde zijn voor het geheugenadres tussen 0 en 65535, waar de waarde naar toe kan worden geschreven.

m - is het gegeven(data), welke naar het opgegeven geheugenadres wordt geschreven.
De waarde moet tussen 0 en 255 liggen (integerwaarde).
Het tegengestelde is PEEK.

WAARSCHUWING:

Basic controleert de geheugenadressen niet.
Dus poke niet luk raak in de basic stack,
in de variabelentabel of in uw programma.

Voorbeeld : POKE 82,4

Zet de waarde 4 in geheugenadres 82.
Of zet de linkerkantlijn op 4.

Voor een uitgebreide lijst van de meest gebruikte POKE-adressen zie Appendix A.

Zie ook DPOKE.

Omschrijving: Zet de terugkeer waarde op nul die opgeslagen is in de laatst uitgevoerde FOR - of GOSUB instructie.

Versie : ATARI BASIC TURBO BASIC COMPILER
 *** *** ***

Formaat : POP

Opmerking : POP is geldig voor routines met GOSUB en EXEC (Turbo Basic) en lussen als:

```
FOR ... NEXT
REPEAT ... UNTIL
WHILE ... WEND
```

Indien het nodig mocht blijken bovengenoemde lussubroutine constructie voortijdig te verlaten dan is het beter deze te verlaten via de POP-instructie.

Voorbeeld :

```
5 DIM A$(100)
10 FOR I=1 TO 100
20 INPUT A$
30 IF A$="STOP" THEN POP:GOTO 60
40 PRINT A$
50 NEXT I
60 IF A$="STOP" THEN PRINT "U heeft STOP
ingetoets!"
70 END
```

Bovenstaand voorbeeld vraagt 100 x een stringinput. Indien men STOP intoetst voordat de teller I op 100 staat wordt de lus FOR ... NEXT beëindigd en naar regel 60 gesprongen.

Zie ook EXIT.

Omschrijving: Plaats de cursor op een op te geven beeld-
schermpositie.

Versie : ATARI BASIC TURBO BASIC COMPILER
 *** *** ***

Formaat : POSITION x,y
 of POS. x,y

Opmerking : x - is de positie op een regel.
 y - is de betreffende regel.

Voorbeeld : 10 FOR X=1 TO 20
 20 POSITION X,X:PRINT "TEST"
 30 NEXT X

Dit voorbeeld print het woordt TEST op het
beeldscherm af, beginnende op positie 1 van
regel 1.

Dan op positie 2 van regel 2 enz. enz.
Het gevolg is een schuin naar onder lopend
beeld gevuld met het woord TEST.

Ook te gebruiken in andere graphics modi.

```
10 GRAPHICS 18
20 FOR X=3 TO 8
30 POSITION X,X:? #6;"TEST"
40 NEXT X
50 GOTO 50
```

Zie ook XI0.

Omschrijving: Stuurt gegevens (data) naar het beeldscherm
of een ander apparaat.

Versie : ATARI BASIC TURBO BASIC COMPILER
 *** *** ***

Formaat : PRINT [#n;][“lyst van uitdrukkingen”][;]...
 of in plaats van PRINT alleen PR. of ?

Opmerking : #n; - n is het kanaalnummer/IOCB
 1 t/m 7, indien naar een bestand of
 graphic modus wordt geprint.

 “lyst van uitdrukkingen”
 mag elk teken zijn welke ingetoetst
 kan worden via het toetsenbord.
 Tevens mag elke numerieke of string-
 variabele verstuurd worden met PRINT.

Voorbeeld : PRINT - drukt een blanco regel af.
 PRINT A - drukt variabelewaarde A af.
 PRINT A\$ - drukt " " " A\$ af.
 PRINT "TEST!" - drukt TEST! af.

Of naar een apparaat?
(zie voorbeeld onder OPEN)

Of naar een Graphics modus?
(zie voorbeeld 2 onder POSITION)

Er kunnen meender uitdrukkingen of variabelen
na een PRINT-instructie worden uitgevoerd,
gescheiden door een komma (,) of een puntkom-
ma (;)

Een komma (,) na een uitdrukking stuurt de
volgende uitdrukking of variabele naar de
volgende ingestelde Tabulatorinstelling. Een
puntkomma (;) plaatst de uitdrukking / varia-
bele direct achter elkaar zonder spatie.

```
READY
PRINT "TEST", "TEST" <RETURN>
TEST TEST
READY
PRINT "TEST"; "TEST" <RETURN>
TESTTEST
READY
```



Na de PRINT-instructie mogen eveneens berekeningen worden uitgevoerd.

```
READY
10 X=5
20 PRINT X+5,X-5,X*(-5)
RUN
10      0      -25
READY
```

Opgemerkt kan worden dat de PRINT instructie met van een van de belangrijkste instructies is in Basic.

Met deze opdracht wordt namelijk de mogelijkheid geschapen iets mee te delen op het scherm of naar een extern apparaat.

Omschrijving: Aanduiding voor het begin van een Subroutine
(PROCEDURE).

Versie : ATARI BASIC TURBO BASIC COMPILER
*** **

Formaat : PROC name

Opmerking : name - is de PROCEDUREnaam onder
welke de subroutine wordt aangeroepen.
Zie voor nadere uitleg en voorbeeld
onder:

EXEC

ON arg EXEC pname,

Omschrijving: Geeft de waarde van de vuurknop van de paddle aan.

Versie : ATARI BASIC TURBO BASIC COMPILER
 *** *** ***

Formaat : A=PTRIG(n)

Opmerking : n - paddlenummer waarnaar gekeken moet worden hoe de stand van de vuurknop is (van 0 tot en met 3)
 A krijgt de waarde 0 als de vuurknop is ingedrukt en de waarde 1 als er niet op de vuurknop wordt gedrukt.

0=aktie 1=ruststand.

Omschrijving: Zendt een byte (datagegeven) naar een te specificeren apparaat.

Versie : ATARI BASIC TURBO BASIC COMPILER
 ***1) ***1)en)2 ***

Formaat : 1) PUT #n, byte
 2) PUT n

Opmerking : 1) # - verplicht teken.
 n - kanaalnummer 1 t/m 7 welke verwijst naar het met OPEN geopende bestand of file.
 2) n - waarde van een integertype welke moet liggen tussen 0 en 255. Met deze instructie kan een karakter naar het beeldscherm worden gestuurd. Aangezien hier net zoals onder 1) de IOCB aanduiding #n ontbreekt, wordt automatisch IOCB #0 geopend; het beeldscherm of de Editor.

PUT n is equivalent aan:
PRINT CHR\$(n)

Voorbeeld : 1)

```
10 OPEN #2, 8, 0, "D:MYPROG.DAT"
20 FOR BYTE=1 TO 100
30   PUT #2, BYTE
40 NEXT BYTE
```

Met dit voorbeeld worden 100 tekens in het bestand MYPROG.DAT geschreven.

PUT #2, BYTE is gelijk aan PRINT #2, A

Gegevens terughalen uit MYPROG.DAT
Zie het voorbeeld bij GET

```
2)

10 FOR KARAKTER=0 TO 255
20   PUT KARAKTER
30 NEXT KARAKTER
```

Met dit voorbeeld worden alle karakters van de Atari op het scherm getoond.

Zie ook BPUT en %PUT.

Omschrijving: Schrijft een byte naar een gespecificeerd
apparaat.

Versie : ATARI BASIC TURBO BASIC COMPILER
 *** ***

Formaat : %PUT #n, byte

Opmerking : % - indicatie voor de interpreter dat de byte
die wordt opgegeven in gecomprimeerde
vorm (6 byte-form) moet worden opgeslag-
en.
Ruimtebesparing op het opslagmedium en
snellere overdracht.

- verplicht teken.

n - Kanaalnummer (1 t/m 7) verwijst naar het
geopende Kanaal (zie ook OPEN).

byte - Het getal welke moet worden opgeslagen

Voorbeeld : 10 OPEN #1,8,0,"D:BESTAND.DAT"
20 FOR I= 1 TO 1000
30 BYTE=RAND(100)
40 %PUT #1,BYTE
50 NEXT I
60 CLOSE #1

Schrijft 1000 willekeurige getallen tussen
0 en 100 naar de geopende file: BESTAND.DAT

Zie ook %GET

=====

Omschrijving: Alle trigometrische functies welke na de RAD
instructie volgen worden in radialen uitge-
drukt.

Versie : ATARI BASIC TURBO BASIC COMPILER
*** **

Formaat : RAD

Opmerking : RAD zet alle trigometrische functies om in
radialen totdat de DEG instructie wordt
gespecificeerd, dan volgt uitdrukking in
graden.

Omschrijving: Geeft een willekeurig heel getal van 0 to n.

Versie : ATARI BASIC TURBO BASIC COMPILER
*** **

Formaat : v=RAND(n)

Opmerking : n - een numerieke uitdrukking welke de bovengrens voorsteld van het te Kiezen getalgebied.
Indien n=100 dan wordt een willekeurig getal geleverd tussen 0 en 100 (incl. 0 en 100).

RAND(n) is de verkorte notatie van:
TRUNC(RND(0)*n)

Voorbeeld : ? RAND(100)
of
V=RAND(100):? V

Zie ook RND, FRAC en TRUNC.

Omschrijving: Leest de gegevens in een DATA-regel en kent deze aan een variabele toe.

Versie : ATARI BASIC TURBO BASIC COMPILER
 *** *** ***

Formaat : READ var[, var]...

Opmerking : var is een numerieke of stringvariabel welke het gegeven uit de DATA regel wordt toegekend.
 Het READ statement houdt een pointer (teller) bij zodat hij bij een volgende READ instructie weet waar hij is gebleven.
 Dus alle DATA's uit een DATA regel hoeven niet in een keer te worden gelezen.
 Elk moment kan deze pointer weer hersteld worden, zodat de wijzer weer naar de eerste of andere DATA regel wijst.
 Zie hiervoor de RESTORE instructie.

Voorbeeld : 10 DIM NAAM\$(20),PLAATS\$(20)
 20 PRINT "NAAM", "PLAATS", "TEL. NR"
 30 READ NAAM\$,PLAATS\$, TEL
 40 PRINT NAAM\$,PLAATS\$, TEL
 50 END
 60 DATA JAN, ZWOLLE, 1234567
 RUN
 NAAM PLAATS TEL. NR
 JAN ZWOLLE 1234567
 READY

Omschrijving: Laat het toe dat verklarende tekst of anderszins in het programma kan worden geplaatst.

Versie : ATARI BASIC TURBO BASIC COMPILER
 *** *** ***

Formaat : REM opmerking
 of R. of [spatie].

Opmerking : Elk teken of opmerking achter REM wordt door BASIC genegeerd.
 Veel REM regels in een programma maakt het programma voor een ander leesbaarder, doch het programma wordt er wel trager door.
 REM mag direct na een regelnummer worden geplaatst of achter een opdracht, functie of wat dan ook, gescheiden door een dubbele punt (:)

Voorbeeld : 10 REM Hier begint het programma.
 20 REM nu volgt een lus
 30 FOR
 40 GOSUB 500:REM ga naar een subroutine
 .
 .
 100 NEXT ...
 110 END:REM dit is het einde
 500 REM *** dit is subroutine x ***
 .
 .
 599 RETURN:REM terug

Omschrijving: Geeft een file op diskette een andere naam.

Versie : ATARI BASIC TURBO BASIC COMPILER
*** **

Formaat : RENAME "D[n]:old,new"

Opmerking : n - is het drivenummer, niet verplicht als
drive 1 bedoeld wordt.

old is de oude filename.ext

new is de nieuwe filename.ext

Voorbeeld : RENAME "D:MYPROG.BAS, YOURFILE.BAS"

Dit is vergelijkbaar met:
XIO 32, #7, 0, 0, "D[n]:old,new"

en met de RENAME optie van het DOSmenu.



RENUM
systeem instructie

=====
Omschrijving: Hernummert programmaregels.

Versie : ATARI BASIC TURBO BASIC COMPILER
*** **

Formaat : RENUM old,new,incr

Opmerking : old - het oude regelnummer vanaf waar de her-
nummering beginnen moet.
new - het nummer wat de oude regelnummer moet
worden.
incr- de verhoging (stapgrootte) waarmee de
hernummering moet plaats vinden.

De regelnummers voor "oud" blijven ongewij-
zigd. De regelnummers na een instructie als
GOTO, GOSUB, TRAP, RESTORE, LIST, DEL, ON-
GOTO en ON-GOSUB worden eveneens hernummerd.
Bij sprongopdrachten waaraan een berekening
te pas komt, werkt RENUM uiteraard niet;
zoals GOTO A, GOSUB 100+10*A, TRAP A+B,
RESTORE A*10+1000 enz....

Indien na de spronginstructies een getal
staat (GOTO 100+10*A) wordt dat eerste ge-
tal wel normaal behandeld door RENUM, de
rest blijft echter ongewijzigd.

Sprongopdrachten meet namen (procedures)
kunnen wel behandeld worden.

Voorbeeld : Het volgende voorbeeld hernummerd het pro-
gramma van regel 130 af. De nieuwe beginwaar-
de voor 130 wordt 200 en de stapgrootte is
20. De regels vanaf 200 (oud 130) worden dus
steeds met 20 verhoogd; 200-220-240 enz....

RENUM 130,200,20

of het hele programma (stel regel 1 is 10):

RENUM 10,10,10

Alle drie waarden moeten worden ingevuld.

Omschrijving: Verwijzing voor READ om een DATA regel te kunnen lezen.

Versie : ATARI BASIC TURBO BASIC COMPILER
*** *** ***

Formaat : RESTORE [line]
of RES.

Opmerking : line - is niet verplicht.

Indien RESTORE zonder line (regelnummer) in een programma wordt geplaatst wordt de pointer/wijzer, voor READ altijd naar het eerste gegeven in de eerste DATA regel gezet. Met RESTORE 150 wordt bij een volgende leesopdracht (READ) met het eerste DATA gegeven in reegel 150 begonnen. Ook al zijn er voor regel 150 dataregels geplaatst, die nog niet zijn gelezen.

Voorbeeld : READY
10 READ A
20 RESTORE
30 READ B
40 PRINT A, B
100 DATA 150, 200, 250
110 DATA
.
.
150 DATA 123, 456, 789
RUN
150 123
READY

Omschrijving: Verwijzing voor READ om een DATA regel te
Kunnen lezen.

Versie : ATARI BASIC TURBO BASIC COMPILER
*** **

Formaat : RESTORE # name

Opmerking : Zie opmerking bij GO #.., TRAP #.. en RESTORE

Voorbeeld : -

Omschrijving: Beeindigt een subroutine en laat het programma vervolgen met de instructie direct gevolgd op de laatste uitgevoerde instructie.

Versie : ATARI BASIC TURBO BASIC COMPILER
 *** *** ***

Formaat : RETURN
 of RET.

Opmerking : Dit is de enige mogelijkheid om een subroutine na het volledig uitvoeren van de instructies in die subroutine, te laten beëindigen.
En andere, niet elegante manier en zeker niet in gestructureerd programmeren, om een routine te laten stoppen is de POP instructie (zie aldaar).

Voorbeeld : 5 DIM A\$(FRE(0)-1000)
 10 INPUT A\$
 20 IF A\$<>"STOP" GOSUB 100:GOTO 10
 30 PRINT A\$
 40 END
 100 REM *** begin subroutine ***
 110 A=A+LEN(A\$)
 120 PRINT "Totaal ingevoerde tekens: ";A
 130 RETURN

Zie ook EXEC, PROC en ENDPROC.

 Omachrijving: Geeft een willekeurig (random) nummer tussen 0 en 1.

Versie : ATARI BASIC TURBO BASIC COMPILER
 ***1) ***1)2) ***

Formaat : 1) A=RND(0)
 2) A=RND[(n)]

Opmerking : 1) verplicht formaat in ATARI BASIC.
 2) in TURBO BASIC is de optie tussen [] niet verplicht.
 (n) - mag een willekeurige waarde zijn.
 Om geheugenruimte te besparen is RND voldoende.
 De waarde die aan de variabele A wordt toegerekend ligt tussen 0 en 1, doch zal nooit 1 worden.
 Wil men een random nummer groter dan 1, dan moet men vermenigvuldigen met de hoogste gewenste waarde.
 Wil men hele getallen (integers) dan moet de uitkomst met INT bewerkt worden.

Voorbeeld : 1) A=RND(0) - getal tussen 0 en 1
 A=RND(0)*8 - getal tussen 0 en 8
 A=INT(RND(0)*8)+1 - integer tussen 0 en 9

 2) A=RND
 A=RND*8
 A=INT(RND*8)+1

=====

Omschrijving: Start de uitvoering van het programma.

Versie : ATARI BASIC TURBO BASIC COMPILER
 *** *** ***

Formaat : RUN ["dev[n]:[filespec]"]
 of RU.

Opmerking : dev - device of aan te spreken apparaat kan
 zijn D of C voor Diskdrive of Cassette-
 recorder.
 n - is niet verplicht. Geeft het drivenummer
 aan tussen 1 en 8; default is 1.
 : - verplicht als dev is ingevuld net zo als
 de " " (aanhalingstekens).
 filespec - programmaam + extender bij ge-
 bruik van diskdrive of ramdisk.

RUN zet alle variabelen, alsmede de toegeken-
de dimwaarden van reeksen (arrays) en strings
op nul.

Voorbeeld : 10 FOR X=1 TO 10
 20 PRINT "X= ";X
 30 NEXT X
 RUN
 X=1
 X=2
 .
 .
 .
 enz

RUN "C:"
of
RUN "D:MYPROG.BAS" laadt eerste en voert
respectievelijk het programma op de cassette-
recorder of MYPROG.BAS op de diskette in
drive 1 uit.



SAVE
system instructie

Omschrijving: Stuurt een basicprogramma naar een randapparaat voor opslag.

Versie : ATARI BASIC TURBO BASIC COMPILER
*** **

Formaat : SAVE "dev[n]:[filespec]"
of S.

Opmerking : dev - device of apparaat, kan zijn:
D voor Drive of
C voor Cassetterecorder.

n - devicenummer 1 t/m 8 is niet verplicht;
default is 1

filespec - programmanaam + extender

Bij gebruik van cassetterecorder is het niet nodig een programmanaam + extender mee te sturen naar de tape, aangezien de naam niet wordt opgeslagen in een directory zoals bij de diskdrive. Bij het laden moet de opnamekop van de cass.rec tot vlak voor het punt worden gebracht, waar het te lezen programma eerder is gesaved.

Bij gebruik van een drive is de naam wel verplicht, aangezien deze met een verwijzing waar het programma op de diskette begint, wordt opgeslagen in de diskettedirectory.

Voorbeeld : SAVE "C:"
SAVE "D2:MYPROG.BAS"

Omschrijving: Zet het Kleurregister met en Kleur van een bepaalde helderheid aan.

Versie : ATARI BASIC TURBO BASIC COMPILER
 *** *** ***

Formaat : SETCOLOR reg, k1, lum
 of SE.

Opmerking : reg - het Kleurregister; Keeuze uit
 0 t/m 4
 k1 - de Kleur 1 t/m 15 = 16 Kleuren
 lum - de luminatie of helderheid van de Kleur
 0 t/m 15 = 16 luminaties.

Een klein rekensommetje leert ons dat de ATARI-computer 256 (16x16) kleuren aan kan.

SETCOLOR tabel met default ingestelde kleuren

REG.	KL.	LUM.	KLEUR

0	2	8	Oranje
1	12	10	Groen
2	9	4	Donker blauw
3	4	6	Rose of rood
4	0	0	Zwart

De te kiezen Kleuren met de corresponderende nummers zijn:

0 grijs	5 violet-rood	10 turquoise
1 goud	6 violetblauw	11 blauw groen
2 oranje	7 blauw	12 groen
3 oranje rood	8 blauw	13 geel-groen
4 rose	9 licht blauw	14 oranje groen
		15 licht oranje

 Omschrijving: Geeft het teken (sign) van een numerieke uitdrukking.

Versie : ATARI BASIC TURBO BASIC COMPILER
 *** *** ***

Formaat : var=SGN(x)

Opmerking : var - variabele
 x - mag elke numerieke uitdrukking zijn.

Indien x positief is dan is de waarde van de variabele 1

Als x gelijk nul is dan is var=0

Als x negatief is dan is var=-1

Voorbeeld : ON SGN(X)+2 GOSUB 1000,2000,3000

Ga naar de subroutine 1000 als x negatief is en naar de subroutine 2000 als x nul is en naar de subroutine 3000 als x positief is.

=====
Omschrijving: Geeft de sinus van een hoek.

Versie : ATARI BASIC TURBO BASIC COMPILER
*** **

Formaat : var=SIN(x)

Opmerking : var - een variabele
x - is de hoek in radialen aangegeven.
Wenst men graden naar radialen te transpor-
teren, vermenigvuldig dan met PI/180;
PI=3.14593

Voorbeeld : READY
10 PI=3.14593
20 GRADEN=90
30 RADIALEN=GRADEN*PI/180
40 PRINT SIN(RADIALEN)
RUN
1
READY

Omschrijving: Produceert geluid via de luidspreker van TV of monitor.

Versie : ATARI BASIC TURBO BASIC COMPILER
 ***1) ***1)2) ***

Formaat : 1)SOUND voice, freq, dis, vol of SO.
 2)SOUND (Zie ook DSOUND)

Opmerking : 1) Activeert een van de vier geluidskanalen om via de TV of de Monitor een signaal te laten horen. Het signaal blijkt hoorbaar tot een nieuw SOUND-commando, NEW, RUN of END volgt.
De SOUND opdracht moet in ATARI BASIC door 4 waarden gevolgd worden. Deze waarden mogen cijfers, variabelen of numerieke uitdrukkingen zijn.

voice - geluidskanaal 0 t/m 3

freq - frequentie. De frequentie bedraagt 8 bit (0...255), hetgeen resulteert in een berekende frequentie van $63921/(2*freq+2)$. Deze waarde komt uit een Amerikaanse publicatie en verschilt van de Europese norm. E.e.a vindt zijn oorzaak in het NTSC-systeem in Amerika en de Europese PAL-norm. De publicatie van Atari geeft als frequentie:
 $31960/freq+1$

dis - disonantie/vervorming
De vervormingswaarde moet liggen tussen 0 en 14, waarbij alleen de even getallen geldig zijn voor "zuivere" tonen. Bij de oneven getallen treden andere vervormingen op.

vol - volume (0-15). Hoe hoger het getal des te sterker het geluid. 0 als volume is geen geluid (uit).
Let erop dat het totaalvolume van de 4 kanalen tezamen niet boven 32 komt. Het kan een "brommen" van de luidspreker tot gevolg hebben.

Zie verder de tabel van de toonwaarden met de bijbehorende Klavierwaarden.

2) SOUND zonder de toevoegingen bij 1) dient om de 4 geluidskanalen uit te zetten. Het is de verkorte vorm van:
FOR I=0 TO 3: SOUND I, 0, 0, 0: NEXT I

Voorbeeld : 10 FOR I= 1 TO 10
20 READ FREQ
30 SOUND 1, FREQ, 14, 12
40 FOR WAIT=1 TO 90: NEXT WAIT
50 SOUND 1, 0, 0, 0
60 FOR WAIT=1 TO 90: NEXT WAIT
70 NEXT I
80 DATA 255, 243, 230, 217, 204, 193, 182, 173

Opgemerkt zij nog dat de Atari homecomputers 3 1/2 octaaf aan kan.

=====

Omschrijving: Zet de positieve wortel van een positief getal in een variabele.

Versie : ATARI BASIC TURBO BASIC COMPILER
*** **

Formaat : A=SQR(x)

Opmerking : x moet groter of gelijk nul zijn.

Voorbeeld : READY
10 X=10
20 PRINT X, SQR(X)
RUN
10 3.162278
READY

Omschrijving: Zet de statuswaarde van het door IOCB geopende
 apparaat in een numerieke variabele.

Versie : ATARI BASIC TURBO BASIC COMPILER
 *** *** ***

Formaat : STATUS #n, var
 of ST.

Opmerking : De statuswaarde wordt gelezen van een
 nader te specificeren apparaat,
 n - device/apparaatnummer 1 t/m 7
 var - de variabele waar de statuswaarde wordt
 ingezet.

Omschrijving: Geeft een waarde af naar gelang de stand van de joystick (stuurknuppel).

Versie : ATARI BASIC TURBO BASIC COMPILER
 *** *** ***

Formaat : X=STICK(n)

Opmerking : n - is 0 voor joystickpoort 1 (eerste poort aan de rechterzijde van de computer).
 n - is 1 voor poort 2 (volgende poort)

De waarden die geleverd kunnen worden zijn:

- 15 - joystick in ruststand
- 14 - noord (naar voren)
- 13 - zuid (naar achteren)
- 11 - west (naar links)
- 7 - oost (naar rechts)
- 6 - noordoost (rechtsvoor)
- 5 - zuidoost (rechtsachter)
- 9 - zuidwest (linksachter)
- 10 - noordwest (linksvoor)

Voorbeeld : 10 GRAPHICS 5+16
 20 X=40:Y=24
 30 X=STICK(0)
 40 IF X<>15 THEN GOSUB 100
 50 PLOT X,Y
 60 GOTO 10
 100 IF X=14 THEN Y=Y-1
 110 IF X=10 THEN Y=Y-1:X=X-1
 120 IF X=6 THEN Y=Y-1:X=X+1
 130 IF X=13 THEN Y=Y+1
 140 IF X=9 THEN Y=Y+1:X=X-1
 150 IF X=5 THEN Y=Y+1:X=X+1
 160 IF X=11 THEN X=X-1
 170 IF X=7 THEN X=X+1
 180 RETURN

=====
Omschrijving: Stopt het programma en keert terug naar de
direct mode.

Versie : ATARI BASIC TURBO BASIC COMPILER
*** **

Formaat : STOP

Opmerking : Indien in een programma de instructie STOP
wordt geplaatst, wordt de programma uitvoering
gestopt. De bestanden worden echter niet
gesloten (zie END, NEW, RUN) en het geluid
blijft klinken.
Het programma kan worden voortgezet door in
de direct mode CONT en een return in te
toetsen.

Voorbeeld : IF A>100 THEN STOP

Het regelnummer waar gestopt is, wordt op het
scherm getoond.

Omschrijving: Zet een getal om in een string.

Versie : ATARI BASIC TURBO BASIC COMPILER
*** **

Formaat : X\$=STR\$(x)

Opmerking : x - mag elke numerieke uitdrukking zijn.
De numerieke uitdrukking wordt na omzetting met STR\$ niet meer herkend als een getal of numerieke uitdrukking.
Het tegenovergestelde is VAL (zie VAL).

Voorbeeld : X\$=STR\$(100)

X\$ wordt nu "100" en is dus gelijk aan:
X\$="100"

```
10 GRAPHICS 0
20 PRINT "GEEF EEN GETAL ";:INPUT A
30 PRINT "GETAL ";A;" ALS STRING ";STR$(A)
```

Omschrijving: Geeft de waarde van de vuurknop van de joystick aan.

Versie : ATARI BASIC TURBO BASIC COMPILER
 *** *** ***

Formaat : A=STRIG(n)

Opmerking : n - joysticknummer waarnaar gekeken moet worden hoe de stand van de vuurknop is (0 of 1)
 Zie STICK

A - variabele krijgt de waarde 0 bij ingedrukte vuurknop en 1 als deze niet is ingedrukt.

0 = actie 1 = ruststand

Voorbeeld : Twee gelijke programma's om de borderkleur te veranderen met een druk op de vuurknop. Voorbeeld 1 in normaal Atari Basic en voorbeeld 2 in Turbo Basic XL 1.5

```
10 GRAPHICS 0
20 POKE 752,1
30 A=STRIG(0)
40 IF A=0 THEN GOSUB 100
50 POKE 712,KL
60 GOTO 20
100 KL=INT(RND(0)*254)+1
110 RETURN
```

Uitleg:

```
Regel 10: maak scherm schoon
" 20: zet cursor uit 0-aan
" 30: vuurknop voor joystick 1
" 40: wordt knop gedrukt spring dan naar subroutine 100
" 50: POKE een kleurcode in het geheugenadres van de borderkleur
" 60: spring terug naar regel 20
" 100: Kies een willekeurig nummer voor de kleur tussen 0 en 255
" 110: verlaat de subroutine (terug)
```

In Turbo Basic:

```
10 CLS
20 POKE 752,1
30 DO
40 A=STRIG(0)
50 IF A=0
60 EXEC KLEUR
70 POKE 712,KL
80 ENDIF
90 LOOP
```

(zie verder)



```
100 PROC KLEUR
110 KL=INT (RND*254)+1
120 ENDPROC
```

Uiteraard werkt voorbeeld 1 ook onder Turbo Basic, doch voorbeeld 2 werkt niet in normaal Atari Basic.

De regels 50, 60 en 70 mogen ook vervangen worden door:

```
50 IF A=0 THEN EXEC KLEUR
60 POKE 712, KL
```

Voor de leesbaarheid is voorbeeld 2 wel het meest duidelijk.



TEXT

beeldscherm instructie

Omschrijving: Plaatst tekst in een graphic scherm.

Versie : ATARI BASIC TURBO BASIC COMPILER
*** **

Formaat : TEXT x, y, exp

Opmerking : x - x coördinaat van een graphic scherm
(positie)
y - y coördinaat van een graphic scherm
(regel)
exp een stringuitdrukking welke tussen aan-
halingstekens moet worden geplaatst.
Een numerieke uitdrukking kan zonder aan-
halingstekens worden geplaatst.

De x en y coördinaten zijn de positie van het
eerste teken (linksboven). De positie wordt
in beeldpunten geteld. De TEXT routine maakt
gebruik van een zeer snelle PLOT routine,
omdat kort naastelkaar liggende beeldpunten
relatief eenvoudig berekend kunnen worden.

Voorbeeld : 10 GRAPHICS 8
20 TEXT 50,90, "TURBO BASIC"
30 TEXT 70,95,1000
40 GOTO 40

Zie ook voorbeeld bij FILLTO

Omschrijving: Onthoudt de tijd van de interne timer.
(RTCLCK)

Versie : ATARI BASIC TURBO BASIC COMPILER
 *** ***

Formaat : V=TIME

Opmerking : V - is de variabele waar de tijd van de interne timer van de Atari computers in wordt opgeslagen.
De tijd wordt geregistreerd in 1/50 seconden.
De waarde wordt uit de adressen 18, 19 en 20 gehaald.

Voorbeeld : PRINT TIME -geeft de tijd uit de adressen
 18 t/m 20

 PRINT RAND(TIME) - Kiest een willekeurig getal aan de hand van TIME
 (adres 18 t/m 20)

 Omschrijving: Ondervangt een fout en stuurt het programma naar een gespecificeerde regel.

Versie : ATARI BASIC TURBO BASIC COMPILER
 *** *** ***

Formaat : TRAP line
 of T.

Opmerking : line
 Is de regelnummer waar het programma verder moet gaan als een fout wordt ontdekt. Nadat er een fout is geconstateerd schakelt TRAP zich zelf uit. Wil men bij een eventuele volgende fout eveneens een verwijzing, dan moet TRAP line opnieuw gespecificeerd worden. TRAP kan ook door het programma worden uitgeschakelt; dan moet men voor 'line' een nummer hoger dan 32767 invoeren. 40000 is een veel gebruikt nummer. TRAP 40000

Voorbeeld : 10 TRAP 100
 20 INPUT A
 30 PRINT A
 40 GOTO 20
 100 PRINT "ER IS GEEN GETAL INGEVOERD"
 110 GOTO 10

Typt men bij de vraag in regel 20 een string, b.v. STOP, in dan constateert het programma een foute invoer, die door TRAP wordt ondervangen.

De foutmelding gebeurt dan in regel 100.

PEEK(195) geeft het foutnummer aan en PEEK(187)*256+PEEK(186) het regelnummer waar de fout is ontdekt.

Zie ook TRAP#...



TRAP#....

programma instructie

Omschrijving: Sprongopdracht naar een gespecificeerde lijn
na constatering van een fout.

Versie : ATARI BASIC TURBO BASIC COMPILER
*** **

Formaat : TRAP # name

Opmerking : # - verplicht teken om de interpreter duidelij-
k te maken dat een naam in plaats van
een regelnummer volgt.

name - een naam/string welke al dan niet
betrekking mag hebben op de fout-
melding.

Zie ook GO#... en RESTORE#...

Voorbeeld : 10 TRAP # FOUT
20 REPEAT
30 INPUT "Voer een getal in: ";B
40 A=A+B
50 UNTIL A>100
60 PRINT A
70 END
100 #FOUT
110 PRINT "U heeft een foutieve invoer
gegeven."
120 GOTO 10



UINSTR
programma instructie

Omschrijving: Zoekt een string in een langere string.

Versie : ATARI BASIC TURBO BASIC COMPILER
*** **

Formaat : A=UINSTR(A\$,B\$[,I])

Opmerking : Zie opmerking bij INSTR.

Als extra geldt dat nu niet wordt gelet op het verschil tussen hoofd- en kleine letters. Bij het zoeken b. v. naar MODEM telt ook Modem of MoDeM mee. Tevens mag de tekst in inverse video zijn.

Voorbeeld : Zie voorbeeld bij INSTR.
Verander in dat voorbeeld B\$="tEnTEn"
Het resultaat blijft positie 10

=====

Omschrijving: Roept een machinetaal subroutine aan in Basic

Versie : ATARI BASIC TURBO BASIC COMPILER
 *** *** ***

Formaat : A=USR(adres, arg[, arg])

Opmerking : adres - is het geheugenadres waar de machinetaal subroutine begint.

arg - zijn 1 of meerdere argumenten welke in de vorm van een enkele byte aan een stapel worden toegevoegd. Het toevoegen geschiedt in tegengestelde volgorde. Dus het laatste argument het eerst, enz. enz.....

Bij het aanroepen wordt gebruik gemaakt van een dummy variabele b.v. A (zie formaat) Eerst nadat alle argumenten van de stapel zijn gehaald zal de controle weer aan Basic worden gegeven.

Worden geen argumenten opgegeven, dan wordt een nul aan de stapel toegevoegd.

Voorbeeld : A=USR(1536)

De argumenten in adres 1536 worden uitgevoerd; dit is de zgn. PAGE 6, een blanco pagina in de Map. Hier kunnen machinetaal onderprogramma's worden opgeborgen en samen (geheel onafhankelijk van en) met Basic worden uitgevoerd.

B.v. een leuk melodietje op de achtergrond tijdens de uitvoering van het programma.

Omschrijving: Verandert een string in een numeriek gegeven.

Versie : ATARI BASIC TURBO BASIC COMPILER
 *** *** ***

Formaat : x=VAL (string uitdrukking)

Opmerking : De string uitdrukking moet beginnen met een
 getal, anders strandt het programma met een
 foutnummer 18.

Voorbeeld : READY
 10 DIM A\$(10)
 20 A\$="1000"
 30 ? VAL (A\$)
 RUN
 1000
 READY

of

```
10 DIM A$(10)
20 A$="1AMSTERDAM"
30 ? VAL (A$)
RUN
1
READY
```

VAL is het tegengestelde van STR\$
(zie aldaar).



WHILE en WEND
programma instructie

Omschrijving: Herhaling van opdrachten totdat aan een gespecificeerde voorwaarde wordt voldaan.

Versie : ATARI BASIC TURBO BASIC COMPILER
 *** ***

Formaat : WHILE arg opdr WEND

Opmerking : arg - De voorwaarde waaraan voldaan moet worden wil men de lus kunnen verlaten.
 De voorwaarde-test wordt aan het begin van de lusconstructie uitgevoerd. Is deze waar dan vervolgt de programma uitvoer na WEND.

 opdr- Opdrachten die worden uitgevoerd totdat -arg- waar is.

 Indien de voorwaarde de eerste keer reeds waar is, wordt de lus geen enkele keer uitgevoerd.

Voorbeeld : 10 CLS
 20 EXEC VRAAG
 30 WHILE KEY<>65
 40 PRINT KEY
 50 EXEC VRAAG
 60 WEND
 70 PRINT "U heeft de juiste toets gevonden"
 80 PRINT "het was: ";STR\$(KEY)
 90 END
 100 PROC VRAAG
 110 PRINT "DRUK EEN TOETS IN!!"
 120 GET KEY
 130 ENDPROC

Hetzelfde programma als bij REPEAT en UNTIL maar dan tegengesteld. (zie aldaar)





TURBO BASIC XL 1.5 MANUAL

AANHANGSEL A.





AANHANGSEL A: FOUTMELDINGEN

Voorwat betreft enige uitleg wanneer een fout wordt geconstateerd door BASIC zie Hoofdstuk 2-5 onder de Interpreter.

Het is mogelijk om een ERROR- welke de programma-uitvoering normalerwijs gesproken stopt te omzeilen. Dit doet men met de TRAP-functie. (zie TRAP-statement in Hoofdstuk 3).

Andere statements welke met ERROR's te maken hebben zijn ERR en ERL (Zie hoofdstuk 3)

Onderstaande lijst is in verband met zijn compleetheid geheel overgenomen uit een in Atari Magine van de Stichting Atari Gebruikers gepubliceerd artikel.

Een completere lijst ben ik tot op heden nog niet tegengekomen en neem deze onder dankzegging aan Wim Denie over.

Enige aanvulling mijnerzijds daargelaten.

ERROR : Omschrijving
nummer :

-
- nr. 2 Insufficient Memory
Teweinig geheugenruimte om nog nieuwe opdrachten of een nieuwe string of array in die omvang te kunnen verwerken. Bekijk of uw programma soms lege variabelen bevat, die gewist kunnen worden of breid uw geheugen uit.
- nr. 3 Value Error
U hebt een te groot getal (positief of negatief) gebruikt. Het is mogelijk dat u een te hoog regelnummer boven 32767 hebt ingevoerd of u bent tijdens een PLOT of DRAWTO opdracht over de mogelijke beeldcoördinaten gegaan; zie ook nr. 144.
- nr. 4 Too many Variables
Het programma bevat in Atari Basic meer dan 128 en in Turbo Basic meer dan 256 variabelen. Ook de eerder gebruikte (toegekende) variabelen tellen hierbij mee. LIST uw programma naar diskette. Typ NEW en Enter uw programma weer naar het geheugen. De Variabelentabel is dan geschoond en bevat alleen de in het programma voorkomende variabelen.
- nr. 5 String Length Error
Er is een string gedimensioneerd en nu probeert u te lezen of te schrijven naar een locatie, die buiten de string-dimensie ligt. Maak de dimensie van die string groter.
- nr. 6 Out of Data
Er staan te weinig Data in een Dataregel om elke variabeel te vullen die met een READ-opdracht wordt gelezen. Verwarrend is dat de foutmelding de READ-

ERROR : Omschrijving
nummer :

-
- opdracht als de foute regel aangeeft. Vul de DATA aan tot alle variabelen een waarde hebben. Meestal gebeurt het dat bij het invoeren van de Data regels een punt i.p.v. een komma is ingetypt. De READ-opdracht leest deze twee waarden als een; met als gevolg dat aan het einde van de data reeks een cijfer of string te kort is. In een ietwat komische vorm komt deze fout steeds voor als u met de cursor over het woordje READY gaat en op <return> drukt. De computer denkt READ Y te lezen en deelt "verontwaardigd" mee niet voldoende DATA te hebben om Y te kunnen vullen.
- nr. 7 Number greater then 32767
Er is een regelnummer groter dan 32767 gebruikt. Kijk ook uw GOTO en GOSUB opdrachten na.
- nr. 8 Input Statement Error
Veel voorkomend als er getracht wordt om aan een numerieke variabele een stringwaarde toe te kennen of als er om een getal wordt gevraagd en men drukt alleen op de <return> toets als input.
- nr. 9 Array of String DIM error
Er is getracht om een array of string een waarde boven resp. 5460 of 32767 toe te kennen, of een array of string opnieuw te dimensioneren of verwezen naar een niet-gedimensioneerde array of string. Vergelijk uw variabelen met de gedimensioneerde waarden.
- nr. 10 Stack overflow
Er zijn teveel of te ingewikkelde subroutines in een stack ondergebracht.
- nr. 11 Floating point overflow/underflow
Getracht een waarde groter/kleiner dan 10/-10 tot de macht 98 gebruikt, waarschijnlijk door te delen door 0.
- nr. 12 Line not found
Verwijzing naar een niet-bestaand regelnummer b.v. met GOTO, GOSUB of na THEN enz. .
- nr. 13 No matching FOR
Ook wel genaamd: Next without For; er is een Next te gencekomen zonder de benodigde For. Mogelijk hebt u geneste lussen niet helemaal binnen de andere lussen gezet.
- nr. 14 Line too long
Basicregel van meer dan 114 tekens of 3 schermregels gemaakt. Kort eventuele statements af, dan accepteerd de computer wel meer tekens. Er mag dan wel niet meer in deze regel worden gewijzigd.



ERROR : Omschrijving
nummer:

-
- nr. 15 Gosub or For line deleted
De met een Next of Return corresponderende instructies resp. For en Gosub zijn gewist.
- nr. 16 Return error
Return without Gosub; er staat een return opdracht zonder de bijbehorende gosub opdracht. Een enkele keer kan ook een End instructie na een hoofdprogramma ontbreken.
- nr. 17 Syntax-error (Garbage error)
De computer vindt onzin. U kunt een fout in de hardware hebben of een onjuiste POKE instructie hebben gebruikt.
- nr. 18 VAL Function error
De string in een VAL-functie is geen numerieke string.
- nr. 19 Load Program too long
Te weinig geheugen beschikbaar om een programma dat u van disk of tape haalt geheel in het interne geheugen van de computer te plaatsen.
- nr. 20 Device Number Error
Het gebruikte devicenummer ligt niet tussen 1 en 7.
- nr. 21 Load File Error
U hebt geprobeerd een programma te laden met LOAD dat niet met SAVE maar met CSAVE of met LIST was opgeslagen. Laad het programma met CLOAD of ENTER. Komt ook voor indien u een machinetaalprogramma of een bestand met LOAD wilt laden.
- nr. 22 Nest-error
U heeft een lusconstructie geprogrammeerd waarin een onderdeel mist bv. ENDWHILE bij WHILE, ENDF bij IF; ook na *F+; eveneens kan deze error optreden indien ongeoorloofd een FOR NEXT lus of een subroutine wordt beëindigd.
- nr. 23 Wend without While
In een While ... Wend lus ontbreekt het statement While. Dit kan ook voorkomen indien men middels een Goto opdracht in een lus springt. Als de programma-uitvoer dan Wend tegenkomt volgt de errormelding.
- nr. 24 Until without Repeat
Gelijk aan error nr. 23 maar dan met Until en Repeat lussen.
- nr. 25 Loop without Do
Gelijk aan error nr. 23 maar dan met Loop en Do lussen.

ERROR : Omschrijving
nummer :

-
- nr. 26 Exit-error
Er wordt getracht uit een niet-bestaande lus te springen.
- nr. 27 Xproc-error
Executing Procedure; deze fout treedt op indien een procedure wordt uitgevoerd zonder dat deze met Exec is aangeroepen. Veel voorkomend na ongeoorloofde springopdrachten.
- nr. 28 Endproc without Exec
Gelijk aan error nr. 23 maar dan met Endproc en Exec lussen.
- nr. 29 Proc error
Er is een onbekende procedure aangeroepen. Komt voor indien typfouten worden gemaakt bij het invoeren van een listing. De procedurenamen na EXEC en PROC moeten gelijk zijn. Controleer eventueel met de DUMP-functie.
- nr. 30 # error
Er is een onbekende regelnaam aangeroepen. Zie verder opmerkingen bij nr. 29
- nr. 128 Break abort
Er is tijdens een I/O operatie op de Break-toets gedrukt en daardoor de uitvoering afgebroken.
- nr. 129 IOCB Already Open
Er is geprobeerd een IOCB (INPUT OUTPUT CONTROL BLOCK; een stukje geheugenruimte, gereserveerd voor de communicatie met een bepaald randapparaat) te openen, terwijl dat reeds geopend was en al voor een andere Device in gebruik was.
- nr. 130 Non-existent Device Error
U hebt een Device (randapparaat) aangesproken, dat niet bestaat; byv. een niet aangesloten printer. Meestal is er echter alleen maar de D: vergeten te typen bij het invoeren van de filenaam
- nr. 131 IOCB Write Only
Er werd een READ opdracht gegeven aan een IOCB dat alleen voor Write (schrijven) gereserveerd was; m. n. het IOCB voor de Printer. Desgewenst kan het betreffende IOCB voor Read of voor Read en Write worden geopend.
- nr. 132 Illegal Handler Command
Er is aan een bepaalde Device een opdracht gegeven, welke deze niet kan uitvoeren.

ERROR : Omschrijving
nummer:

-
- nr. 133 Device/File Not Open
Een aangesproken Device is niet geopend. Er is b.v. getracht Plot en Drawto commando's te gebruiken in een tekstmode. Het scherm werd echter niet eerst voor de juiste grafische mode geopend.
- nr. 134 Bad IOCB Index
Er is een verkeerd IOCB nummer opgegeven. In Basic mag dit 1 t/m 7 zijn.
- nr. 135 IOCB Read Only
Het omgekeerde van nr. 131; er is getracht naar een device te schrijven, dat alleen kan lezen.
- nr. 136 End of File
De computer heeft een "End of File" blok gevonden (dit zijn de laatste 128 bytes van een file), terwijl eerder is verteld dat de file langer is. Ook kan deze fout voorkomen als men probeert van de disk te lezen uit een niet geopend bestand.
- nr. 137 Truncated Record
Een poging om een record te lezen, groter dan het aan het Operating System (CIO) gespecificeerde maximum voor zo'n record (in Basic 119 bytes, anders 256). Het gebruik van INPUT instructies in een met PUT opgezet file leidt tot deze fout.
- nr. 138 Device Timeout
Een aangesproken apparaat reageert niet binnen de daarvoor door het Operating System bepaalde tijd. Het verkeerde device is aangesproken of er is onjuist devicenummer gebruikt. Andere mogelijkheden zijn:
- device stuk
- device staat niet aan
- onjuist drivenummer
Bij cassette-recorders kan de baud onjuist zijn of men heeft de tape niet op de goede plaats gestart. Komt deze fout vaker voor, terwijl alle aansluitingen correct zijn, dan moet het betreffende randapparaat nagekeken worden; waarschijnlijk is dan een IC stuk.
- nr. 139 Device Nak (Not Acknowledged)
Een device reageert niet omdat het onbegrijpelijke of illegale commando's krijgt. Kijk alle aansluitingen na en controleer of u geldige commando's hebt ingegeven. Deze fout is sterk afhankelijk van een bepaald randapparaat en in dit geval is de gebruiksaanwijzing van het specifieke Device helaas vaak de enige bron van afdoende informatie.

ERROR : Omschrijving
nummer :

nr. 140 Serial Bus Error

Er is iets mis in de POKEY-chip; daardoor wordt de informatieuitwisseling tussen computer en een bepaald device onmogelijk. Indien deze error vaak voorkomt zal naar alle waarschijnlijkheid uw computer of randapparaat moeten worden nagekeken. Alleen bij een cassette-recorder wil het juist plaatsen van de tape nog weleens helpen. (bit 7 van SKSTAT in Pokey is gezet)

nr. 141 Cursor out of Range

Er is getracht de cursor buiten het scherm van een bepaalde mode (grafisch-tekst mode) te plaatsen. Dit komt vaak voor in grafische modi. Bepaal een kleinere waarde voor de X en Y coördinaten voor de cursorpositie. Kijk uw Position, Plot en Drawto opdrachten na.

nr. 142 Serial Bus Overrun

Hiervoor geldt vrijwel hetzelfde als voor Error 140. De Pokey-chip krijgt nieuwe informatie toegezonden terwijl de oude nog niet is verwerkt. (bit 5 van SKSTAT in Pokey is gezet).

nr. 143 Serial Bus Checksum Error

Onduidelijke Errormelding; fouten zowel in hardware als software mogelijk. De door een device meegestuurde controlecode klopt niet met de door de computer berekende waarde.

nr. 144 Device Done

Het device kan het opgegeven commando niet uitvoeren.

Mogelijkheden zijn:

- naar een write protected diskette schrijven
- write protect schakelaartje staat verkeerd
- schrijven naar een beschadigde sector op disk

nr. 145 Illegal Screen Mode

Er is een grafische mode aangesproken met een ongeldig nummer. Ook mogelijk dat bij het schrijven naar disk de verify niet klopt. Geschrevene is niet in overeenstemming met hetgeen er had moeten worden geschreven.

nr. 146 Function Not Implemented

Er is een opdracht gegeven voor een device terwijl deze niet voor het device bestemd kan zijn; b.v. :

- lezen van een printer
- een PUT naar het Toetsenbord ("K: ")

nr. 147 Insufficient RAM

Teweinig Ramgeheugen vrij voor de gekozen grafische mode (m.n. de 800XL in modes met zeer hoge resolutie).

ERROR : Omschrijving
nummer:

-
- nr. 160 Drive Number Error
Het gekozen drivenummer lag niet tussen 1 en 8 of de desbetreffende drive is niet opgestart (ingeschakeld).
- nr. 161 Too Many OPEN files
Er is geen buffer meer vrij om aan een nieuwe file toe te kennen. Controleer of files zijn geopend, welke gesloten kunnen worden. In DOS 2.* Kan met POKE 1801, aantal het maximaal aantal geopende files veranderen. Met optie H van Dosmenu kunt u dit dan weer vastleggen.
- nr. 162 Disk Full
Tijdens het op diskette zetten van een file raakt de disk vol (als hij dat al niet van meet af aan was). Het laatste bestand staat niet geheel op de disk en er zal een nieuwe disk moeten worden genomen om de hele file opnieuw op te nemen. (het onvolledige bestand komt wel in de Disk-directory voor als een volwaardig file). Wis dit van de disk om zo plaats te maken voor kleinere files).
- nr. 163 Unrecoverable System I/O Error
Er is iets mis met uw DOS of een slechte diskette is geïnstalleerd.
Probeer eerst een andere DOS.
- nr. 164 File Number Mismatch
In elke file staan twee belangrijke nummers. Het eerste is de zgn. sector link dat doorverwijst naar de volgende sector van dat file. Het tweede is het filenummer. Dit nummer is in elke sector aanwezig. Als een van deze twee nummers in een bepaalde sector verkeerd staat (ook weleens gebruikt voor beschermingen) dan kan deze fout optreden. Drastische maatregelen als Disk Doctor of Disktool kunnen misschien nog hulp bieden, anders gaat het betreffende programma verloren.
- nr. 165 File Name Error
In de filename+text zijn illegale tekens gebruikt of het eerste teken van de bestandsnaam is geen letter. Als eerste teken mag geen cijfer worden gebruikt; verder mogen geen kleine letters, leestekens of spaties in de naam voorkomen.
- nr. 166 Point Data Length Error
De lengte van een record is onjuist.
- nr. 167 File Protected
Er is getracht met een beveiligd (Locked) file iets anders te doen dan te lezen. Eerst "unlocken".

ERROR : Omschrijving
nummer :

-
- nr. 168 Device Command Invalid
Er is een commando gegeven, dat voor de software van dat device ongeldig is.
- nr. 169 Directory Full
Er is getracht meer dan 64 file namen op een diskette te zetten. Dit heeft niets te maken met de diskruimte (zie hiervoor nr. 162).
- nr. 170 File Not Found
Er is een onbekend bestand opgeroepen. Controleer de ingevoerde filename+ext; deze moet precies overeenkomen met de namen opgenomen in de diskdirectory. Roep eventueel eerst de diskdirectory op met DIR of met Dos-optie A.
- nr. 171 Point Invalid
Er is een POINT instructie gegeven en het daarbij behorende aantal bytes komt niet overeen met de bytes in die file.
- nr. 172 Illegal Append
Een heel zeldzame error, die alleen voorkomt als men Dos 2 probeert te gebruiken om een DOS 1 file te openen voor append (toevoegen). Copieer eerst de betreffende file naar een Dos 2 schijf en herhaal het commando.
- nr. 173 Bad Sectors Format Time
De drive heeft bij het formateren "bad sectors" geconstateerd. Er zal een andere diskette moeten worden genomen. Mocht deze fout vaker voorkomen dan moet uw drive worden nagekeken. (De utility Archiver en Disk Wizzard II hebben opties om bad sectors te formateren)
- nr. 174 Duplicate Filename
Er is getracht door een RENAME operatie twee files een zelfde naam te geven op de disk. De drive weigert uw commando uit te voeren. Mochten toch om de een of andere reden twee files op de diskette komen dan kunt u dit herstellen door het programma:
"DISKFIX.COM"
van uw DOS 2.5 master diskette te laden. Kies daarna optie "rename by file#".
- nr. 175 Bad Load file
Er is iets mis met de file waardoor de drive niet in staat is die file te laden. De aanduiding "Bad Load File" kan heel gewoon betekenen dat er getracht wordt een Basic file te laden als een machinetaal file (met optie L van Dos of met BRUN).

ERROR : Omschrijving
nummer :

nr. 176 Incompatible Format

Er wordt geprobeerde een DOS 3 operatie uit te voeren met een Dos 2/2.5 schijf of omgekeerd. Eerst zal Acces DOS2 of Copy 32 moeten worden gedraaid om de schijf naar een ander Dos te vertalen.

nr. 177 Disk Structure Damaged

De schijf is door mechanische/electrische invloeden beschadigd en kan niet meer Dos gelezen worden. Mogelijk is er zelfs sprake van een of ander exotisch formaat, dat door Dos niet als zodanig herkend wordt.

Bovenstaande Error-meldingen Krijgen in Turbo Basic X1 1.5 nog een extra dimensie. Hier wordt de foutmelding nog gecomplementeerd met een aanduiding die het opsnorren van de fout kan vergemakkelijken.

De foutmelding in Turbo Basic ziet er als volgt uit:

```
ERROR-22? NEST
ERROR-29? PROC
ERROR- 2? MEM
```

enzovoort.

De aanduiding achter het ERROR nummer geeft al een kleine hint naar de geconstateerde fout. Mocht trots voorgaande lijst van foutmeldingen nog iets niet duidelijk zijn dan kan de handleiding van het desbetreffende apparaat misschien uitkomst bieden. Bepaalde fouten die nauw samenhangen met een device worden in de handleiding vaak extra aangetipt.

Laat u verder niet ontmoedigen door deze ellen lange lijst. De meeste fouten zijn maar onbenulligheden en zijn daarom zo opgelost.





TURBO BASIC XL 1.5 MANUAL

AANHANGSEL B.





AAHANGSEL B: BASIC DISKETTE INPUT EN OUTPUT

Dit hoofdstuk beschrijft de handelingen welke met het gebruik van Input en Output van en naar de diskette moeten worden verricht.

Het bevat een lijst van commando's en functies welke met diskette gebruikt worden. Verschillende voorbeelden of verwijzingen naar een plaats elders in dit manual trachten het gebruik met diskette/bestanden zo duidelijk mogelijk te maken.

Om enige uniformiteit in het gebruik van programmanamen cq. bestanden te waarborgen, volgt een opsomming van de meest gebruikte extenders. Hierdoor is meestal af te leiden om welk soort 'file' het handelt.

- *.BAS - basicprogramma's (Atari Basic)
- *.COM - gecompileerde basic programma's
- *.SYS - systeempagina's (DOS.SYS en DUP.SYS)
- *.OBJ - machinetaal source programma's
- *.LST - geliste basicprogramma's
- *.DAT - bestanden met gegevens (data's)
- *.TXT - teksten (bestanden van tekstverwerkers)
- *.TUR - basic programma's (Turbo Basic)
- *.PIC - picturefiles

U bent natuurlijk geheel vrij in het gebruik van een filename+ext naar eigen keus, maar om niet in een jungle van allerlei bizarre namen te geraken is het toch raadzaam om filename+ext zoveel mogelijk op de file af te stemmen. Dit vergemakkelijkt niet alleen uw eigen maar ook andermans leesbaarheid.

Commando's

Onderstaand treft men een lijst aan met I/O functies, welke in hoofdstuk 3 meer in detail zijn beschreven.

SAVE filespec

Schrijft het programma welke zich in het RAM-geheugen bevindt naar de diskette.

LIST filespec

Zie SAVE, maar dan in ATASCII-formaat.

LOAD filespec

Laadt een programma van de diskette naar het Ram-geheugen.

ENTER filespec

Zie LOAD, maar dan de ATASCII-vorm. Deze functie opent de mogelijkheid om meerdere programma's aan elkaar te koppelen (mergen).

- RUN filespec**
Laadt een programma van de diskette naar het Ram-geheugen en runt dit vervolgens.
- DELETE filespec**
Vernietigt een "file" op de diskette.
- RENAME filespec**
Veranderd de ene filenaam in een andere filenaam.
- LOCK filespec**
Beschermd een programma tegen ongewenste vernietiging via DELETE. Is niet bestand tegen formateren, wel tegen hersaven.
- UNLOCK filespec**
Haalt de bescherming weg welke met LOCK is verkregen.
- BLOAD filespec**
Zie LOAD, maar dan voor binary-files (machinetaal).
- BRUN filespec**
Zie RUN, maar dan voor binary-files (machinetaal, gecompileerde basicprogramma's)
- DIR filespec**
Roept de disk-directory op.

Commando's met betrekking tot het sturen van data naar bestanden.

De commando's / functies welke gebruikt worden om bestanden op te bouwen zijn:

BGET	BPUT
CLOSE #	GET
%GET	INPUT #
NOTE	OPEN #
POINT	PRINT #
PUT	%PUT

Om een sequentieel bestand te creëren en data naar dit bestand te sturen moeten de volgende stappen in uw programma worden gevolgd.

- 1) Open het bestand voor OUTPUT of APPEND, middels de OPEN-functie.
- 2) Schrijf de gegevens (data) naar het bestand middels de statements:

PRINT #, BPUT, PUT of %PUT

- 3) Sluit het bestand met CLOSE #

Het volgende voorbeeld maakt bovenstaande stappen duidelijk.

```

10 OPEN #1,8,0,"D:MYPROG.DAT"  stap 1
20 PRINT #1;A$                 stap 2
30 PRINT #1;B$                 stap 2
40 CLOSE #1                     stap 3

```

Lezen van een bestand middels:

```
INPUT #, BGET, GET, %GET
```

Voorbeeld:

```

10 OPEN #1,4,0,"D:MYPROG.DAT"  stap 1
20 INPUT #1,X$,Y$              stap 2
30 CLOSE #1                     stap 3

```

Voor meer informatie betreffende de genoemde statements zie hoofdstuk 3.

Het volgende voorbeeld creëert een bestand van, via het toetsenbord ingevoerde gegevens.

```

10 DIM NAAMS$(20),S$(20),W0$(20)
20 INPUT "NAAM ";NAAMS:OPEN #1,8,0,"D:MYPROG.DAT"
30 WHILE NAAMS<>"EINDE"
40 INPUT "STRAAT+NR ";S$
50 INPUT "WOONPLAATS ";W0$
60 GOSUB 1000
70 INPUT " NAAM ";NAAMS
80 WEND
90 CLOSE
100 TRAP 170
110 OPEN #1,4,0,"D:MYPROG.DAT"
120 INPUT #1,NAAMS,S$,W0$
130 PRINT NAAMS$
140 PRINT S$
150 PRINT W0$
160 GOTO 120
170 CLOSE
180 END
1000 PRINT #1;NAAMS$
1010 PRINT #1;S$
1020 PRINT #1;W0$
1030 RETURN

```





TURBO BASIC XL 1.5 MANUAL

AANHANGSEL C.





AANHANGSEL C: ATASCII KARAKTERS EN TOETSENBORD CODES

Dit hoofdstuk bevat een lijst met de karakters die in het ROM gedeelte van de Atari 8 bitters zijn opgeslagen. Men noemt deze karakters ook wel ATASCII codes (Atari ASCII codes).

De tabel geeft de decimale (DEC) en de ATASCII (AT)- waarde weer.

Het programma waarmee deze lijst tot stand is gekomen kunt u terug vinden in Aanhangsel D (Programma-voorbeelden).

DEC	AT	DEC	AT	DEC	AT	DEC	AT	DEC	AT
WAARDE		WAARDE		WAARDE		WAARDE		WAARDE	
000	♥	001	♠	002	♣	003	♠	004	♣
005	♠	006	/	007	\	008	▲	009	■
010	▲	011	■	012	■	013	—	014	—
015	■	016	+	017	+	018	—	019	+
020	•	021	■	022	!	023	†	024	‡
025	!	026	!	027	!	028	↑	029	↓
030	←	031	→	032		033	!	034	"
035	#	036	\$	037	%	038	&	039	'
040	(041)	042	*	043	+	044	,

DEC	AT	DEC	AT	DEC	AT	DEC	AT	DEC	AT
WAARDE		WAARDE		WAARDE		WAARDE		WAARDE	
045	-	046	.	047	/	048	0	049	1
050	2	051	3	052	4	053	5	054	6
055	7	056	8	057	9	058	:	059	;
060	<	061	=	062	>	063	?	064	@
065	A	066	B	067	C	068	D	069	E
070	F	071	G	072	H	073	I	074	J
075	K	076	L	077	M	078	N	079	O
080	P	081	Q	082	R	083	S	084	T
085	U	086	V	087	W	088	X	089	Y

DEC	AT	DEC	AT	DEC	AT	DEC	AT	DEC	AT
WAARDE		WAARDE		WAARDE		WAARDE		WAARDE	
090	Z	091	[092	\	093]	094	^
095	_	096	•	097	a	098	b	099	c
100	d	101	e	102	f	103	g	104	h
105	i	106	j	107	k	108	l	109	m
110	n	111	o	112	p	113	q	114	r
115	s	116	t	117	u	118	v	119	w
120	x	121	y	122	z	123	•	124	
125	•	126	◻	127	◻	128	◻	129	◻
130	◻	131	◻	132	◻	133	◻	134	◻

DEC AT HAARDE	DEC AT HAARDE	DEC AT HAARDE	DEC AT HAARDE	DEC AT HAARDE
135 N	136 V	137 F	138 V	139 L
140 J	141 M	142 M	143 F	144 G
145 F	146 =	147 ::	148 O	149 =
150 M	151 ::	152 ::	153 I	154 L
155 E	156 O	157 O	158 O	159 O
160 M	161 O	162 M	163 ::	164 S
165 Z	166 K	167 M	168 O	169 O
170 G	171 O	172 M	173 =	174 M
175 Z	176 O	177 O	178 O	179 S

DEC AT HAARDE	DEC AT HAARDE	DEC AT HAARDE	DEC AT HAARDE	DEC AT HAARDE
180 A	181 S	182 G	183 Z	184 B
185 O	186 H	187 M	188 K	189 =
190 X	191 O	192 O	193 A	194 B
195 O	196 O	197 E	198 F	199 G
200 H	201 I	202 U	203 K	204 L
205 K	206 N	207 O	208 P	209 O
210 R	211 S	212 I	213 U	214 U
215 K	216 X	217 V	218 Z	219 I
220 N	221 O	222 G	223 M	224 O

DEC AT HAARDE	DEC AT HAARDE	DEC AT HAARDE	DEC AT HAARDE	DEC AT HAARDE
225 S	226 B	227 G	228 O	229 G
230 F	231 O	232 H	233 I	234 M
235 K	236 I	237 K	238 N	239 O
240 P	241 G	242 M	243 S	244 I
245 U	246 U	247 K	248 X	249 M
250 Z	251 O	252 II	253 A	254 4
255 >				

Onderstaand treft u een lijst met codes aan welke wordt gegenereerd indien een toets op het toetsenbord wordt ingedrukt. Deze codes zijn niet gelijk aan de waarden die eerder in dit hoofdstuk zijn opgenomen bij de ATASCII waarden.

Met programma 3 in aanhangsel D kunt u de codes in onderstaande tabel zelf testen.

TOETS	+SHIFT	+CTRL	TOETS	+SHIFT	+CTRL
CODE	CODE	CODE	CODE	CODE	CODE
A	63	127	0	50	114
B	21	85	1	31	95
C	18	82	2	30	94
D	58	122	3	26	90
E	42	106	4	24	88
F	56	120	5	29	93
G	61	125	6	27	91
H	57	121	7	51	115
I	13	77	8	53	117
J	1	65	9	48	112
K	5	69	;	2	66
L	0	64	+	6	70
M	37	101	*	7	71
N	35	99	=	15	79
O	8	72	<	54	118
P	10	74	>	55	119
Q	47	111	ESC	28	92
R	40	104	SPC	33	97
S	62	126	TAB	44	108
T	45	109	INV	39	103
U	11	75	CAPS	60	124
V	16	80	DEL	52	116
W	46	110	RTN	12	76
X	22	86			
Y	43	107			
Z	23	87			





TURBO BASIC XL 1.5 MANUAL

AANHANGSEL D.





AANHANGSEL D: PROGRAMMA'S EN PROGRAMMADELEN

Dit aanhangsel bevat enkele programma's of programmadelen aan de hand waarvan kan worden nagegaan hoe men eventueel in Turbo Basic XL 1.5 kan programmeren. Zoals men weet zijn er altijd slechtere maar zeker ook vele betere methoden om in Turbo Basic XL 1.5. te programmeren. Maar de beginnende Turbo Basicfan kan er in ieder geval zijn of haar voordeel meedoen.

Turbo Menu

Het eerste programma is het Turbo Menu. Met dit programma kunt u de Directory van een diskette op het scherm krijgen vier kolommen van elk zestien files. Door met de cursor toetsen naar een file of programmaam te gaan kan het desbetreffende programma worden gerund door simpel op RETURN te drukken. Ook Binary Load files kunnen worden gestart op gelijke wijze. Bestanden welke een error-21 opleveren kunnen uiteraard niet worden uitgevoerd, evenals de geliste files in Atascii formaat.

Dit programma staat eveneens op de achterzijde van bijgevoegde Public Domain diskette. Het heeft hier de naam AUTORUN.BAS en wordt daarom steeds bij het booten van de diskette gestart/uitgevoerd.

SAVE dit programma op al uw diskette's met Turbo Basic onder de naam AUTORUN.BAS.

Compileren:

Mocht u dit programma willen compileren met de Compiler dan dient u instructie NEW in regel 90 te vervangen door bijvoorbeeld:

```
90 IF KEY=27 THEN GO# BOOT
```

Dan dient u uiteraard een routine met de naam # BOOT te creëren, b. v. :

```
1200 # BOOT
1210 POKE 580,1
1220 GR.2:? #6:? #6;"OP DEZE DISK GEEN"
1230 ? #6;" BASIC "
1240 ? #6;"PLAATS basic DISK"
1250 ? #6;" IN DE DRIVE"
1260 ? #6;"EN DRUK OP RESET !"
1270 PAUSE 200
1280 GO# BOOT
```

TURBO MENU

```
10 REM * MENU KAART VOOR TURBO BASIC *
20 REM * door Wil Breakman (C) 1986 *
30 REM * 045-418695 *
40 CLR :GRAPHICS 0
50 EXEC INIT
60 EXEC DISPLAY
70 EXEC DIRECTORY
80 EXEC SELECTION
```

```

90 IF KEY=27 THEN GRAPHICS 0:NEW
100 IF KEY=77 THEN RUN
110 REM IF KEY=155 THEN RUN HULP$
120 IF KEY=155
130   GRAPHICS 2+16
140   A$="":A$="D:":A$(LEN(A$)+1)=HULP$
150   IF LEN(A$)>9
160     EXT$=HULP$(9,11)
170     IF EXT$<>" "
180       A$(11,11)="":A$(12)=EXT$
190       REPEAT
200         FLAG=1
210         FOR I=1 TO LEN(A$)-1:TRAP 270
220           IF ASC(A$(I,I))=32
230             FLAG=0
240             A$(I)=A$(I+1,LEN(A$))
250             REM A$(LEN(A$))=" "
260           ENDIF
270         NEXT I
280       UNTIL FLAG=1
290       TRAP OFF
300     ENDIF
310   ENDIF
320   POSITION 4,2:? #6;"0gEnB1Ik A. u. B. "
330   POSITION 5,4:? #6;"IK LaAd:":POSITION 7,8:? #6;A$(3)
340   TRAP 350:RUN A$
350   RUN
360 ENDIF
370 -----
380 PROC INIT
390   DIM FILE$(20),A$(1000),HULP$(20),B$(2)
400   DIM EXT$(4)
410   OFF=40000:POKE 729,20:POKE 730,1:POKE 65,0
420 ENDPROC
430 -----
440 PROC DISPLAY
450   DL=DPEEK(560):POKE 82,0
460   GRAPHICS 0:POKE 710,50:POKE 712,50:POKE DL+3,70:POKE
DL+6,7:POKE 708,25:POKE 752,1
470   ? #6;"by W. Braakman (c)'86";
480   ? #6;"      TurbO mEnU "
490   POSITION 2,2:? "FILENAME FILENAME FILENAME"
500   POSITION 1,3:? "[1xctrl Q 8xctrl R 1xctrl W 8xctrl R
1xctrl W 8xctrl R 1xctrl W 8xctrl R 1xctrl E]"
510   FOR I=4 TO 19:POSITION 1,I:? " ";      ;      ;
;      ;":NEXT I
520   POSITION 1,20:? "[1xctrl Z 8xctrl R 1xctrl X 8xctrl R
1xctrl X 8xctrl R 1xctrl X 8xctrl R 1xctrl C]":REM [ en ]
NIET TYPEN
530   ? " KEUZE MET PIJLEN M=MENU ":? " ESC=RUN BASIC
RETURN=RUN PROGRAM": REM M, ESC EN RETURN IN INVERSE
540 ENDPROC
550 -----
560 PROC DIRECTORY
570   CLOSE #1:OPEN #1,6,0,"D:*. *"
580   TRAP 690
590   P=2:R=4
600   REPEAT
610     FOR I=R TO 19

```

```

620     INPUT #1, FILE$
630     IF FILE$(11, 13) = "SYS" OR FILE$(11, 13) = "DAT" OR
FILE$(11, 13) = "COM" THEN 620
640     IF FILE$(5, 16) = "FREE SECTORS" THEN POSITION
29, 21: ? FILE$(1, 3); " VRY ": GOTO 660: REM VRY IN INVERSE
650     POSITION P, I: ? FILE$(3, 10): EXEC OPSLAG
660     NEXT I
670     P=P+9
680     UNTIL P>29
690     CLOSE #1: TRAP OFF
700 ENDPROC
710 -----
720 PROC OPSLAG
730     A$(LEN(A$)+1) = FILE$(3, 13)
740 ENDPROC
750 -----
760 PROC SELECTION
770     TRAP OFF
780     P=2: R=4
790     KEUZE = (R-3) + (INT(P/9) * 16)
800     EXEC INVERS: POSITION P, R: ? FILE$(1, 8)
810     REPEAT
820         X=P: Y=R
830         GET KEY
840         IF KEY=61 THEN R=R+1: IF R>19 THEN R=4
850         IF KEY=45 THEN R=R-1: IF R<4 THEN R=19
860         IF KEY=42 THEN P=P+9: IF P>29 THEN P=2
870         IF KEY=43 THEN P=P-9: IF P<2 THEN P=29
880         KEUZE = (R-3) + (INT(P/9) * 16): EXEC INVERS
890         POSITION P, R: ? FILE$(1, 8)
900     UNTIL KEY=27 OR KEY=77 OR KEY=155
910 ENDPROC
920 -----
930 PROC INVERS
940     TRAP 80
950     IF ASC(FILE$(1, 1)) > 127
960         FOR I=1 TO LEN(FILE$)
970             B=ASC(FILE$(I, I))-128
980             FILE$(I, I)=CHR$(B)
990         NEXT I
1000    POSITION X, Y: ? FILE$(1, 8)
1010 ENDIF
1020 HULP$=A$(KEUZE*11-10, KEUZE*11)
1030 FILE$=""
1040 FOR I=1 TO LEN(HULP$)
1050     B$=HULP$(I, I)
1060     B=ASC(B$)+128
1070     FILE$(I, I)=CHR$(B)
1080 NEXT I
1090 TRAP OFF
1100 ENDPROC
1110 -----

```

Het onderstaande programma geeft u de mogelijkheid om de kadertjes met de ATASCII codes vermeld in aanhangsel C zelf op het scherm te toveren.

Steeds als een scherm gevuld is wacht het programma op een willekeurige toetsdruk (behalve natuurlijk BREAK of RESET).

```

0 REM ***** ATASCII CODE'S *****
1 REM ***** by *****
2 REM ***** Wil Braakman *****
3 REM ***** (C) 1987 *****
10 CLS :POKE 82,0:POKE 752,1
20 EXEC SCHERM
30 DIM T(55,5),X$(7),HULP$(5)
40 FOR RY=0 TO 55
50 FOR KOLOM=1 TO 5
60 T(RY,KOLOM)=I
70 I=I+1
80 NEXT KOLOM
90 NEXT RY
100 -----
110 R=2
120 FOR RY=0 TO 51
130 P=-6:R=R+2:IF R=22 THEN R=4:GET KEY:EXEC SCHERM
140 FOR KOLOM=1 TO 5
150 P=P+7
160 X$="000"
170 X=T(RY,KOLOM)
180 IF X<256
190 X$(LEN(X$)+1)=STR$(X)
200 EXEC KARAKTER_1
210 IF FLAG=0
220 POSITION P,R: ? X$(LEN(X$)-2,LEN(X$));"
";CHR$(X)
230 ELSE
240 POSITION P,R: ? X$(LEN(X$)-2,LEN(X$));" ";HULP$
250 ENDIF
260 ENDIF
270 NEXT KOLOM
280 ?
290 NEXT RY
300 GET KEY:RUN
310 END
320 -----
330 PROC SCHERM
340 POSITION 0,0
350 ? "[1xctrl R 1xctrl W 6xctrl R 1xctrl W
6xctrl R 1xctrl Q 6xctrl R 1xctrl W 6xctrl R 1xctrl E]"
360 FOR KOLOM=0 TO 20
370 ? " ! ; : ; ; ; ; ;"
380 NEXT KOLOM
390 ? "[1xctrl A 6xctrl R 1xctrl S 6xctrl R 1xctrl S
6xctrl R 1xctrl S 6xctrl R 1xctrl S 6xctrl R 1xctrl D]"
400 POSITION 1,3: ? "[1xctrl Z 6xctrl R 1xctrl X 6xctrl R
1xctrl X 6xctrl R 1xctrl X 6xctrl R 1xctrl X 6xctrl R
1xctrl C]"
410 POSITION 1,1: ? "DEC AT!DEC AT!DEC AT!DEC AT!DEC AT"
420 POSITION 1,2: ? "WAARDE!WAARDE!WAARDE!WAARDE!WAARDE"

```

```

430 ENDPROC
440 -----
450 PROC KARAKTER_1
460   FLAG=0
470   IF X=27 THEN HULP$="[2xesc ]":FLAG=1
480   IF X=28 THEN HULP$="[esc esc -]":FLAG=1
490   IF X=29 THEN HULP$="[esc esc =]":FLAG=1
500   IF X=30 THEN HULP$="[esc esc +?]":FLAG=1
510   IF X=31 THEN HULP$="[esc esc *]":FLAG=1
520   IF X=125 THEN HULP$="[esc esc clear]":FLAG=1
530   IF X=126 THEN HULP$="[esc ctrl delete]":FLAG=1
540   IF X=127 THEN HULP$="[esc ctrl insert]":FLAG=1
550   IF X=155 THEN HULP$="[RET]":FLAG=1
560   IF X=156 THEN HULP$="[esc shift delete]":FLAG=1
570   IF X=157 THEN HULP$="[esc shift insert]":FLAG=1
580   IF X=158 THEN HULP$="[esc ctrl tab]":FLAG=1
590   IF X=159 THEN HULP$="[esc shift tab]":FLAG=1
600   IF X=253 THEN HULP$="[esc ctrl 2]":FLAG=1
610   IF X=254 THEN HULP$="[esc delete]":FLAG=1
620   IF X=255 THEN HULP$="[esc tab]":FLAG=1
630 ENDPROC

```

Met het volgende programma kunt u zelf de codes van het toetsenbord testen. Zie ook de tabel op bladzijde C-3 welke middels onderstaand programma is tot stand gekomen.

```

10 CLS :POKE 710,0:POKE 712,0:POKE 700,15:POKE 82,0
20 DIM HEX$(16):HEX$="0123456789ABCDEF"
30 POSITION 5,2:? "KEYBOARD CODE'S"
40 POSITION 5,3:? " by W. Braakman "
50 WHILE DUMMY=0
60   POKE 764,255
70   POSITION 0,5:? "KEYBOARD. CODES IN HEX EN DEC":?
80   REPEAT
90     KEYCODE=PEEK(764)
100    UNTIL KEYCODE<>255
110    HI=INT(KEYCODE/16):LOW=KEYCODE-16*HI
120    PRINT "KEYCODE: HEX $";
130    PRINT HEX$(HI+1,HI+1);HEX$(LOW+1,LOW+1);
140    PRINT ", DECIMAL ";KEYCODE;" "
150 WEND

```

Programma vier geeft de zogenaamde Bubble sort methode weer. De string A\$ bevat de te sorteren gegevens. Het is het eenvoudigste om er voor te zorgen dat alle te sorteren elementen van dezelfde lengte zijn.

```

10 DIM A$(1000)
20 DIM W$(3), H$(20)
30 A$="JOS..... PIET..... KLAAS..... ARNO...
..... DIRK..... ZUS..... JAN..... "
40 L=LEN(A$)/15
50 CLS
60 REPEAT
70   X=1:W$="NEE"
80   FOR I=1 TO L-(L-(L-I))
90     IF A$(X,X+14)>A$(X+15,X+29)
100    H$=A$(X,X+14)

```

```

110     A$(X, X+14)=A$(X+15, X+29)
120     A$(X+15, X+29)=H$(X, X+14)
130     W$="JA"
140     ENDIF
150     X=X+15
160     NEXT I
170 UNTIL W$="NEE"
180 ? A$
  
```

De eeuwigdurende kalender geeft de mogelijkheid om na te gaan welke datum overeenkomt met welke dag van de week. Zo is b. v. 9 januari 1953 een vrijdag, enz. Probeer zelf maar!

```

10 CLS :? "      EEUWIG DURENDE KALENDER"
20 ?
30 ? "VOER IN: DD,MM,EEJJ"
40 INPUT "----->",D,M,J
50 F=J+D+3*M-3
60 X=0
70 IF M<3
80   X=-1
90 ENDIF
100 J=J+X
110 F=F-(X+1)*INT(0.4*M+2.3)
120 F=F+INT(J/4)-INT(3*INT(J/100)/4)-1
130 F=F-7*INT(F/7)
140 IF F=0 THEN ? :? "ZATERDAG"
150 IF F=1 THEN ? :? "ZONDAG"
160 IF F=2 THEN ? :? "MAANDAG"
170 IF F=3 THEN ? :? "DINSDAG"
180 IF F=4 THEN ? :? "WOENSDAG"
190 IF F=5 THEN ? :? "DONDERDAG"
200 IF F=6 THEN ? :? "VRIJDAG"
210 ? :? "DRUK EEN TOETS IN!"
220 GET KEY
230 GOTO 10
  
```

Met de RETURN GENERATOR kunt u zelf DATA regels aan uw programma toevoegen of een tweede gedeelte van een programma naladen van disk of cassette en dit vervolgens laten uitvoeren.

```

0 REM ***** RETURN GENERATOR *****
1 REM *****      by      *****
2 REM *****   Wil Braakman   *****
10 EXEC INFO
20 DIM A$(20)
30 REGEL=990
40 WHILE A$<>"x"
50   REGEL=REGEL+10
60   ? :? "      x = stoppen"
70   ?
80   ? "DATA: ";:INPUT A$
90   EXEC GENERATOR:CLS
100 WEND
110 LIST :END
  
```



```
120 -----
130 PROC GENERATOR
140   CLS
150   POSITION 2,2:? REGEL;" DATA ";"$
160   ? "CONT"
170   POSITION 2,0:POKE 842,13:STOP
180   POKE 842,12
190 ENDPROC
200 -----
210 PROC INFO
220   POKE 82,0:CLS
230   ? "          RETURN GENERATOR          ":REM
gehele regel in inverse
240   ? :? "CREERT DATA REGELS"
250   ? " een hulp bij het"
260   ? " programmeren !?!"
270   ? :? " een * als DATA is stoppen!"
280   ? :?
290 ENDPROC
```

Een demonstratie in Graphics met de functies FILLTO, FCOLOR, TEXT, CIRCLE en PAINT.

```
10 GRAPHICS 24
20 SETCOLOR 2,10,0
30 SETCOLOR 1,6,10:COLOR 1
40 PLOT 250,170
50 DRAWTO 175,10
60 DRAWTO 125,10
70 FILLTO 50,170
80 FCOLOR 1
90 TEXT 110,100,"TURBO BASIC"
100 TEXT 125,120,"XL 1.5"
110 CIRCLE 40,80,30
120 CIRCLE 270,80,30
130 PAINT 40,80:PAINT 270,80
140 GOTO 140
```



