Qbasic - deel 1

Table of Contents

Inleiding

Basic is een programmeertaal voor computers die reeds in de jaren '60 werd gedoceerd aan de universiteiten, maar we moesten wachten op de grote doorbraak tot in 1975, toen Bill Gates en Paul Allen Basic herschreven voor gebruik op de toen uit te brengen "personal computer".
Microsoft werd toen opgericht en begon stilaan met de inpalming van de software markt. MS-DOS verstootte CP/M (van Digital Research) en Basic kon overal op de IBM-compatibele markt doordringen.

Basic staat voor "Beginners All Purpose Symbolic Instruction Code" en wordt nogal eens als een taaltje voor beginners versleten. Nochtans is Basic een vrij flexible taal (er bestaan honderden "dialecten") en vormt het dikwijls een basis voor nieuwe programmeertalen. Ook zijn er verschillende firma's die Basic als standaard programmeertaal gebruiken.

Het is waar dat je in Pascal gestuctureerd programmeert, maar dat kan je in Basic ook: het mag, maar het moet niet.
Het is waar dat C veel sneller en krachtiger is, maar C is ook veel moeilijker en foutengevoeliger dan Basic.

Het grote voordeel van Basic (in deze cursus) is dat het een geïnterpreteerde taal is. Er moet niet telkens opnieuw geëditeerd, gecompileerd, gelinkt en uitgevoerd te worden zoals in andere talen. In Basic is elke verandering in het programma onmiddellijk gekend door de computer.
Het nadeel hiervan is wel de traagheid van de taal. Elke opdracht moet immers elke keer opnieuw vertaald worden naar iets dat de computer begrijpt. Maar tegenwoordig heeft men hier ook compilers voor ontwikkeld, zodat dit laatste argument wegvalt.

Herhaling hardware

CPU

CPU = Central Processing Unit
Is de chip die ongeveer voor alles verantwoordelijk is wat er binnen in uw computer gebeurt. Deze leest instructies en data uit het geheugen, voert de instructies uit, plaatst data terug in het geheugen, geeft opdrachten aan de randapparatuur, ...
CPU's krijgen een "naam" mee vb. bij Intel: 8086, 80486, Pentium, ...
Ze werken aan een bepaalde snelheid, uitgedrukt in MHz (MegaHertz). De standaard IBM compatibele werkt aan 4.77 MHz, tegenwoordig zijn er al die 1000 MHz halen. Deze snelheid duidt aan hoeveel machine instructies er per seconde kunnen verwerkt worden.

RAM geheugen

RAM = Random Access Memory
Is het interne geheugen van de computer dat zeer snel toegankelijk moet zijn en bevat de programma's en de data die direct beschikbaar moeten zijn. Het is een vluchtig geheugen, d.w.z. bij stroomonderbreking is de inhoud van het RAM geheugen weg. Daarom moeten programma's en data op niet-vluchtige media opgeslagen worden voor later gebruik (vb. hard disk).
De grootte van het RAM geheugen wordt uitgedrukt in KB (KiloByte) of MB (MegaByte). 1 KB = 1024 bytes en 1 MB = 1024 KB
De allereerste computers vonden 64 KB een zee van geheugen, later was 640 KB een magische grens (voor DOS), tegenwoordig is 128 MB een gewone zaak.

Diskette drive

PC's hebben minstens 1 diskette drive. Vroeger waren dit meestal 5.25" diskettes, nu worden meestal 3.5" diskettes gebruikt. De diskettes bevatten een schijfje met een magnetiseerbare coating waardoor er informatie op vastgelegd kan worden. Deze informatie kan later terug uitgelezen worden of opnieuw overschreven met andere info.
5.25" diskettes hebben tegenwoordig een capaciteit van 1.2 MB; 3.5" diskettes hebben een capaciteit van 1.44 MB.
Voordeel van diskettes is het gemakkelijk kunnen "doorgeven" van files, het nadeel is hun geringe capaciteit en traagheid.

Hard disk

De dag van vandaag hebben PC's ook minstens 1 harde schijf. In de jaren '80 waren harde schijven dure, luxe artikelen, tegenwoordig woedt er een prijzenoorlog. Harde schijven hebben nu gewoonlijk een capaciteit van 10 GB. (1 GB = 1024 MB) en groeien met de dag.
Harde schijven hebben dus een capaciteit van duizenden diskettes (wat gezien de huidige software geen overbodige luxe is) en zijn zeer snel toegankelijk.

Beeldscherm

Het beeldscherm geeft de gegevens in uw PC alfanumerisch of grafisch weer. Alles hangt eigenlijk af van de grafische kaart die in uw PC steekt. In den beginne kon die kaart enkel tekst afbeelden op het beeldscherm. Later kwamen er kaarten die ook grafisch gegevens konden weergeven. Eerst waren de beelden nog zwart-wit met een lage resolutie, later kwamen er meer en meer kleuren en werd de resolutie opgedreven. Geschiedenis der kaarten: Hercules, CGA, EGA, VGA, SVGA.
De beeldschermen evolueerden ook van monochrome schermpjes tot kleuren schermen en de grootte nam toe van 14" tot 21" en meer.

Toetsenbord

Elke PC heeft ook een toetsenbord. Veel verschillen (behalve de kwaliteit) zijn er niet. Ten opzichte van vroeger zijn er meer functietoetsen en zijn er aparte "eilandjes" voor de getallen, pijltjes en speciale toetsen. Voorts zijn er nog wat verschillen in de posities van de toetsen vb. azerty, qwerty.

Printer

Indien u een presentatie op papier wilt van uw gegevens, heeft u een printer nodig. Die printer kan serieel, parallel of via een netwerk aan uw PC gekoppeld zijn. Er zijn zwart-wit printers en kleuren printers. Verder kunnen ze nog opgedeeld worden volgens afdruk techniek. Zo zijn er de impact printers, de inkjet printers, de thermische printers en de laser printers.

Overige

Verder zijn er nog andere aparaten die aan de PC kunnen gekoppeld worden, die niet altijd door basic ondersteund worden. Zo zijn er:

- de muis: voornamelijk gebruikt in een grafische omgeving.
- de tapestreamer: gebruikt als backup-medium.
- de zip/jazz-drives: gebruikt als grote "diskettes", gaande van 100 MB tot 1 GB.
- de cd-rom: met een capaciteit van 640 MB, is dit goedkoop schijfje zeer geschikt als massa-geheugen drager.
- de scanner: indien u teksten of tekeningen wilt inscannen vanaf papier hebt u een scanner nodig. Wordt geleverd met tekstherkennings-software.
- de modem: internet is tegenwoordig het van het, maar om te communiceren met de buitenwereld heeft u een modem nodig. Deze vertaalt de digitale signalen van uw computer naar analoge signalen van de telefoonlijn en vice versa.
- de geluidskaart met speakers: de tijd van de eenvoudige beeps is ook achter de rug en PC's kunnen nu gebruikt worden om muziek weer te geven en ook te maken.

Opmerking

Vorige lijst is waarschijnlijk niet compleet en binnen de kortste keren weer achterhaald. De technologie in computerland is gewoon niet bij te benen.

Herhaling DOS

Directories

DOS heeft een directory-structuur.
Schijven worden aangeduid met een letter, gevolgd door een dubbele punt.
Elke schijf bevat een hoofd-directory (root-directory), aangeduid door een backslash.
Directories kunnen sub-directories bevatten.
Directories worden gescheiden door een backslash.
Individuele files (in 8.3 formaat) worden in een directory geplaatst.
Vb. C:\DATA\BRIEF.TXT : schijf C: bevat een hoofddirectory \, deze bevat een subdirectory DATA en deze bevat een file met naam BRIEF.TXT.

Er bestaan 2 speciale directories, nl.
. : dit is de huidige directory
.. : dit is de directory 1 stap hoger in de directory
Deze 2 directories zijn overal te gebruiken, behalve in de hoofd-directory.
Vb. C:\DATA\BRIEF.TXT en C:\PROGRAMS\START.BAT zijn 2 bestaande files.
U bevindt zich in directory C:\DATA .
.\BRIEF.TXT betekent dan C:\DATA\BRIEF.TXT en ..\PROGRAMS\START.BAT verwijst dan naar C:\PROGRAMS\START.BAT.

Files

DOS beheert files met een 8.3 formaat. Dit wil zeggen dat de naam begint met maximum 8 letters en/of cijfers en/of enkele speciale tekens, eventueel gevolgd door een punt en terug maximum 3 letters en/of cijfers en/of enkele speciale tekens.
Vb. BRIEF.TXT, TEKST.DAT, PROG1.BAS, PROG2.C, PROEF
Enkele speciale namen zijn gereserveerd voor DOS en worden gebruikt om devices mee aan te duiden.

- AUX : Auxilary devices
- COMx : Seriële poorten
- CON : Console
- LPTx : Parallelle poorten
- NUL : NUL device (het zwarte gat)
- PRN : printer

Enkele commando's

DIR opvragen schijfinhoud, wildcards zijn toegestaan
parameters zijn o.a.
/W (wide) breed formaat
/P (page) wacht na elk scherm
/S (subdirectories) toont ook de sub-directories
Vb: `DIR *.* /P`

MKDIR maak directory
Vb: `MKDIR DATA`

CD verander van directory
Vb: `CD DATA`

RMDIR wis directory (directory moet eerst leeg zijn)
Vb: `RMDIR DATA`

TYPE toon inhoud van file
Vb: `TYPE BRIEF.TXT`

COPY	copieer file, wildcards zijn toegestaan Vb: `COPY BRIEF1.TXT BRIEF2.TXT` copieert BRIEF1.TXT naar BRIEF2.TXT
REN	hernoemt file, wildcards zijn toegestaan Vb: `REN BRIEF.TXT BRIEF.DAT`
DEL	wist file, wildcards zijn toegestaan Vb: `DEL BRIEF.TXT`

Oefening 1

1. Ga naar de hoofd-directory.
2. Kijk of de directory QBASIC bestaat.
3. Indien deze directory niet bestaat, maak ze dan.

De editor

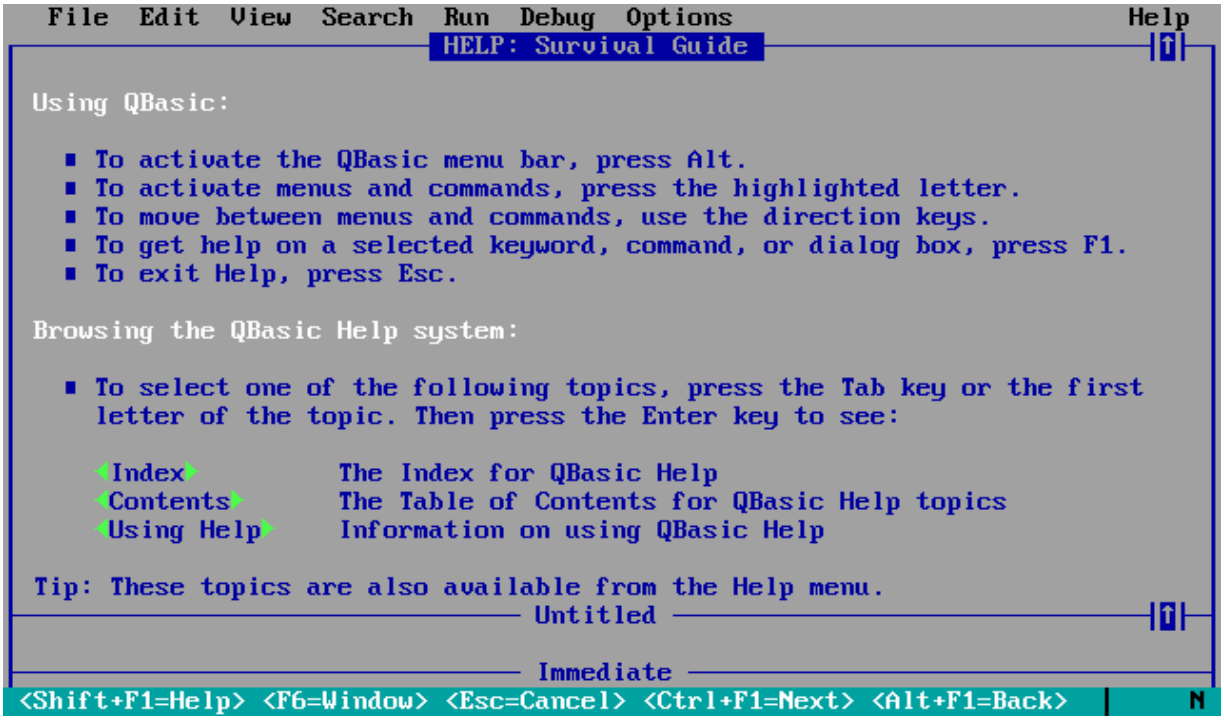
Starten editor

De QBASIC-editor lijkt goed op de EDIT-editor. Dit komt eigenlijk omdat het hetzelfde programma is. EDIT is QBASIC, maar met een aantal opties en menu's uitgeschakeld. EDIT wordt opgestart door QBASIC /ED uit te voeren. QBASIC heeft 2 belangrijke opties:

- /H start QBASIC met het hoogst mogelijk aantal lijnen beschikbaar;
- /RUN <prog> laadt het programma <prog.bas> en voert het onmiddellijk uit.

Start nu QBASIC door op de dos-prompt `QBASIC` te typen. U krijgt dan het volgende scherm:

Druk op Enter om de "survival guide" te bekijken.



Eerste programma

Druk nu op Escape. U gaat nu uw eerste programma maken. Typ het volgende over:

```
rem program      : qb0401.bas
rem author       : michel gielens
rem date written : 2000-09-18
rem description  : programma 01

cls
input "Wat is uw naam "; naam$
print "Dag "; naam$
end
```

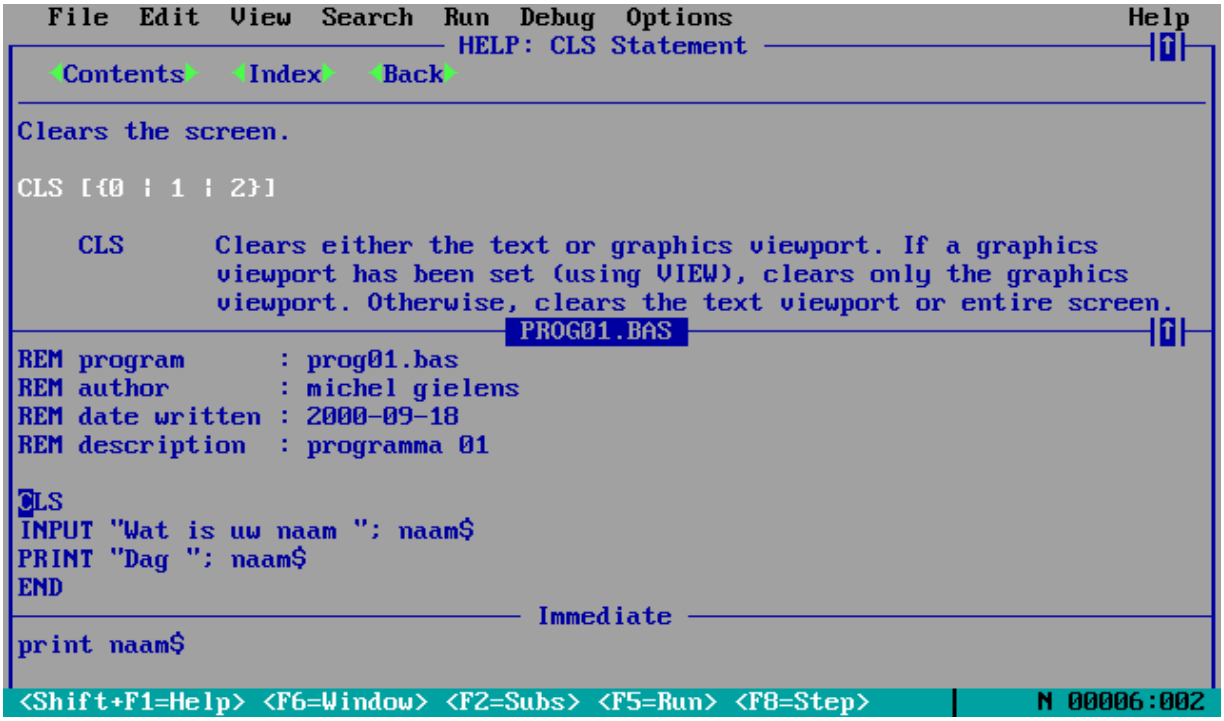
Tijdens het typen zult u gemerkt hebben dan sommige woorden automatisch in hoofdletters worden gezet. Dit zijn de instructies en functies die QBASIC herkent (`REM`, `CLS`, `INPUT`, `PRINT`, `END`,...) Dit omzetten naar hoofdletters zorgt er voor dat de programma's beter leesbaar zijn en dat de programmeur kan zien ofdat er tijdens het typen niets verkeerds getypt werd.

Ga nu met de cursor op de instructie "`CLS`" staan en druk op F1. QBASIC geeft help over de instructie "`CLS`". Deze help is beschikbaar voor alles wat QBASIC herkent.

Druk nu op F5. Het programma wordt uitgevoerd.

`REM` zijn opmerkingen (remarks) en worden door het programma genegeerd
`CLS` wist het scherm
`INPUT "Wat is uw naam "; naam$` vraagt om input vanaf het toetsenbord en slaat deze input op in de variabele "`naam$`"
`PRINT "Dag "; naam$` drukt de constante "`Dag `" en de variabele "`naam$`" af op het scherm
`END` beëindigt het programma

Hier ziet u de editor met in het bovenste panel de help-functie, in het middelste panel uw programma en het het onderste panel de instructies die onmiddellijk (buiten het programma om) uitgevoerd worden.



We gaan nu het programma bewaren op disk voor later gebruik.
Druk Alt-F om het File-menu te bereiken en druk op de S van Safe om het programma te bewaren.
Vermits het een nieuw programma was, dient u er nog een naam aan te geven.

Dit programma kan later terug opgeroepen worden door Alt-F & Load te gebruiken.

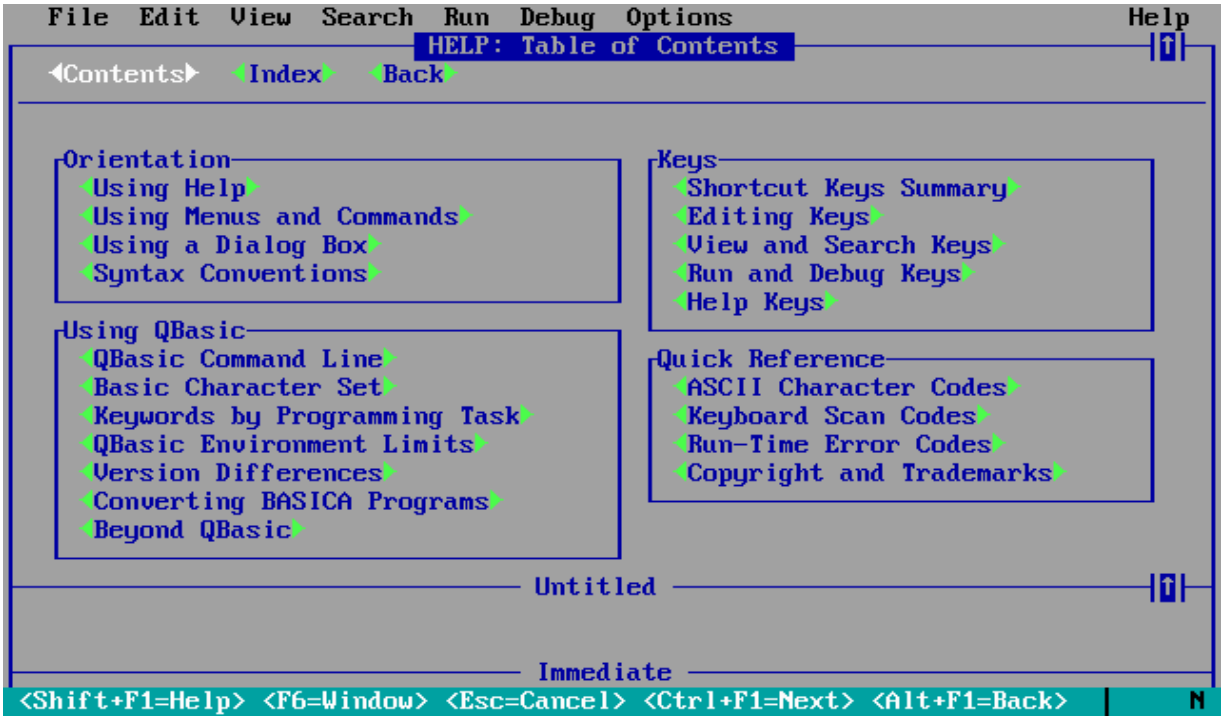
Overzicht menu's

menu-optie	sneltoets	omschrijving
Menu File		
New		Wist het programmavenster
Open...		Laadt een nieuw programma ter bewerking.
Save		Slaat het actuele programma op.
Save As...		Slaat het actuele programma onder een andere naam op.
Print...		Print het actuele programma.
Exit		Beëindigt QBASIC en keert terug naar DOS.
Menu Edit		
Cut	Shift-Del	Wist de gemarkeerde tekst uit het programmavenster en plaatst die in het klembord.
Copy	Ctrl-Ins	Copieert de gemarkeerde tekst uit het programmavenster en plaatst die in het klembord.
Paste	Shift-Ins	Plaatst de inhoud van het klembord in het programmavenster op de actuele cursorpositie.
Clear	Del	Wist de gemarkeerde tekst zonder die naar het klembord te copiëren.
New Sub...		Maakt een nieuwe subroutine aan.
New Function...		Maakt een nieuwe functie aan.
Menu View		
SUBs	F2	U kunt kiezen welk onderdeel van het programma wordt getoond.
Split		Verdeelt het scherm in 2 programmavensters.
Output Screen	F4	Toont het uitvoerscherm.
Menu Search		
Find...		Zoekt naar de opgegeven tekst.
Repeat Last Find	F3	Zoekt naar de volgende plaats waar de tekst die met Find is opgegeven voorkomt.
Change...		Zoekt en vervangt de opgegeven tekst.
Menu Run		
Start	Shift-F5	Start het actuele programma.
Restart		Wist de variabelen en start het actuele programma opnieuw.
Continue	F5	Gaat verder met het actuele programma nadat er was gestopt.
Menu Debug		
Step	F8	Voert de volgende opdracht van het programma uit.
Procedure Step	F10	Voert de volgende opdracht van het programma uit maar slaat oproepen van procedures over.
Trace on/off		De trace-modus wordt in- of uitgeschakeld.

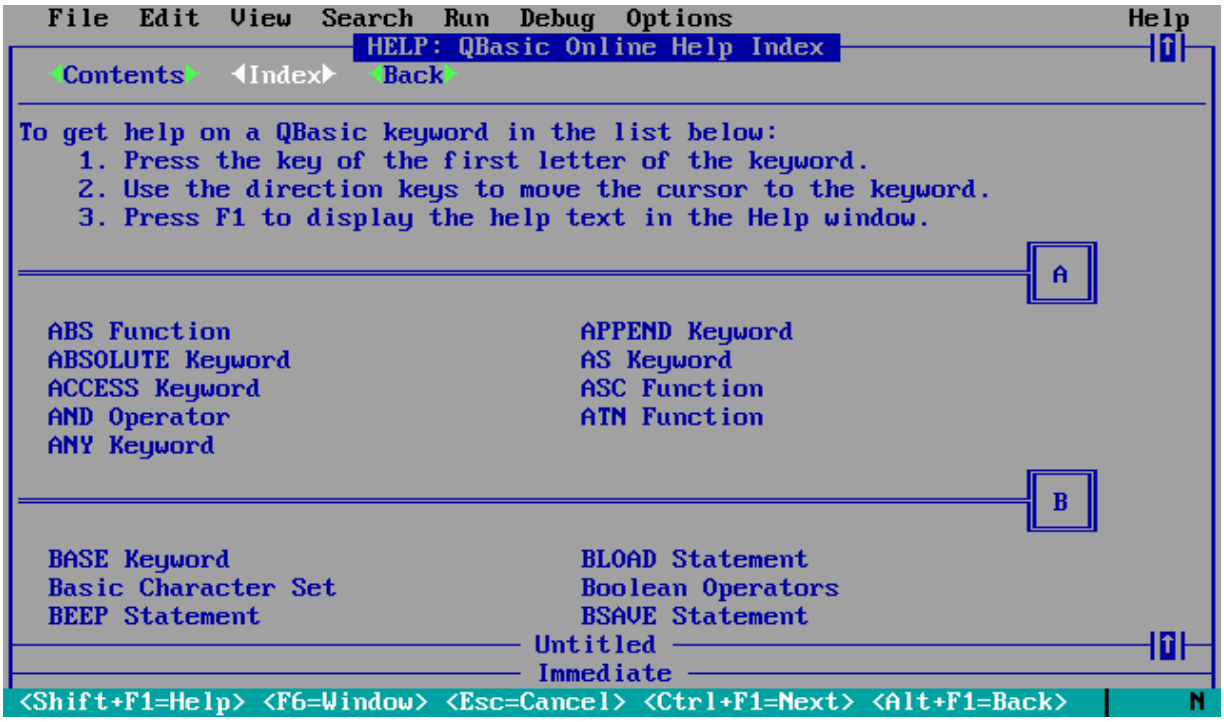
menu-optie	sneltoets	omschrijving
Toggle Breakpoint	F9	Stelt een breakpoint in op de actuele cursorpositie of verwijdert een bestaand breakpoint.
Clear all breakpoints		Verwijdert alle breakpoints uit het hele programma.
Set Next Statement		Zorgt ervoor dat de opdracht op de cursorpositie als volgende wordt uitgevoerd.
Menu Options		
Display...		Hiermee kunt u de schermkleuren en andere weergave-opties instellen.
Help Path...		Hier kunt u de plaats van de help-bestanden opgeven.
Syntax Checking		Schakelt de controle van de syntaxis in of uit.
Menu Help		
Index		Toont een index met informatie over Help.
Contents		Toont tabellen met de inhoud van het Help-systeem.
Topic:	F1	Geeft informatie over de QBasic-instructie waarop de cursor zich bevindt.
Using Help	Shift-F1	Geeft informatie over de help-functie zelf.
About...		Toont informatie over de QBasic-interpreter.

Overzicht help

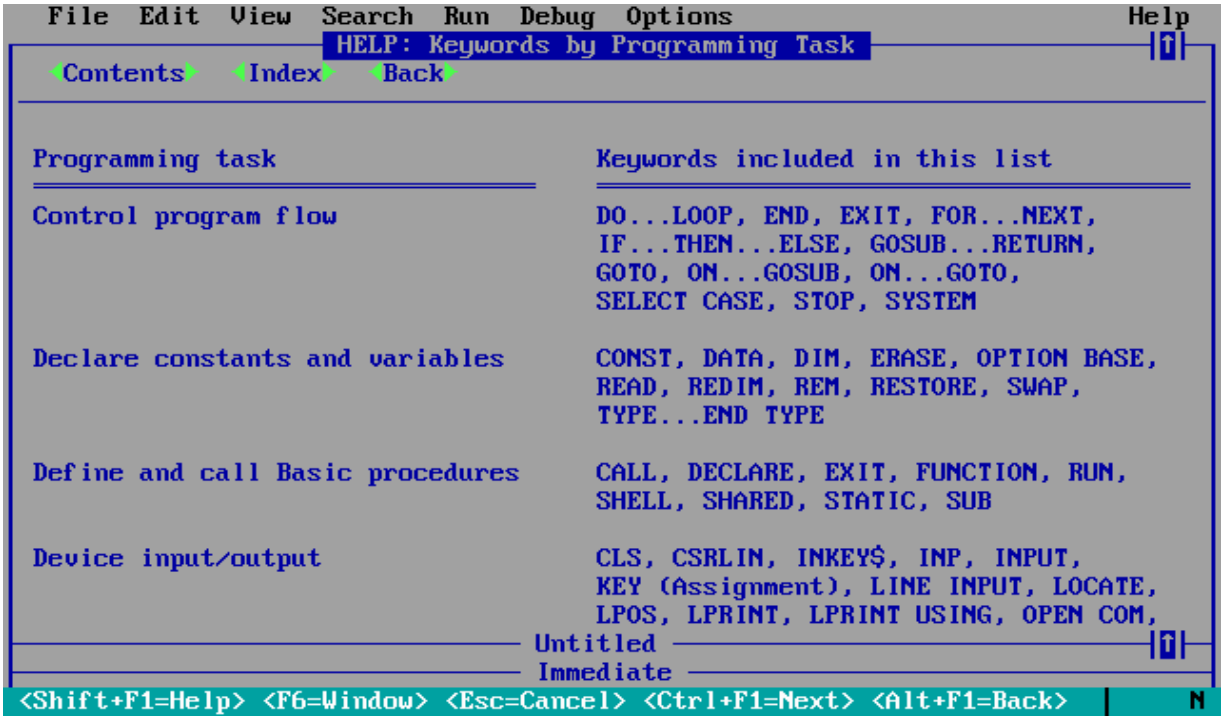
QBasic biedt verschillende soorten help aan. We zagen op het eerste scherm al de "survival guide". We konden help opvragen over een instructie door met de cursor op het betreffende woord te gaan staan en F1 te drukken.
QBasic verschaft ook help per onderwerp of gewoon door alle trefwoorden alfabetisch te rangschikken of door de instructies per programmeertaak te ordenen. QBasic bevat ook een "quick reference" over een aantal zaken: vb. alle ascii tekens.



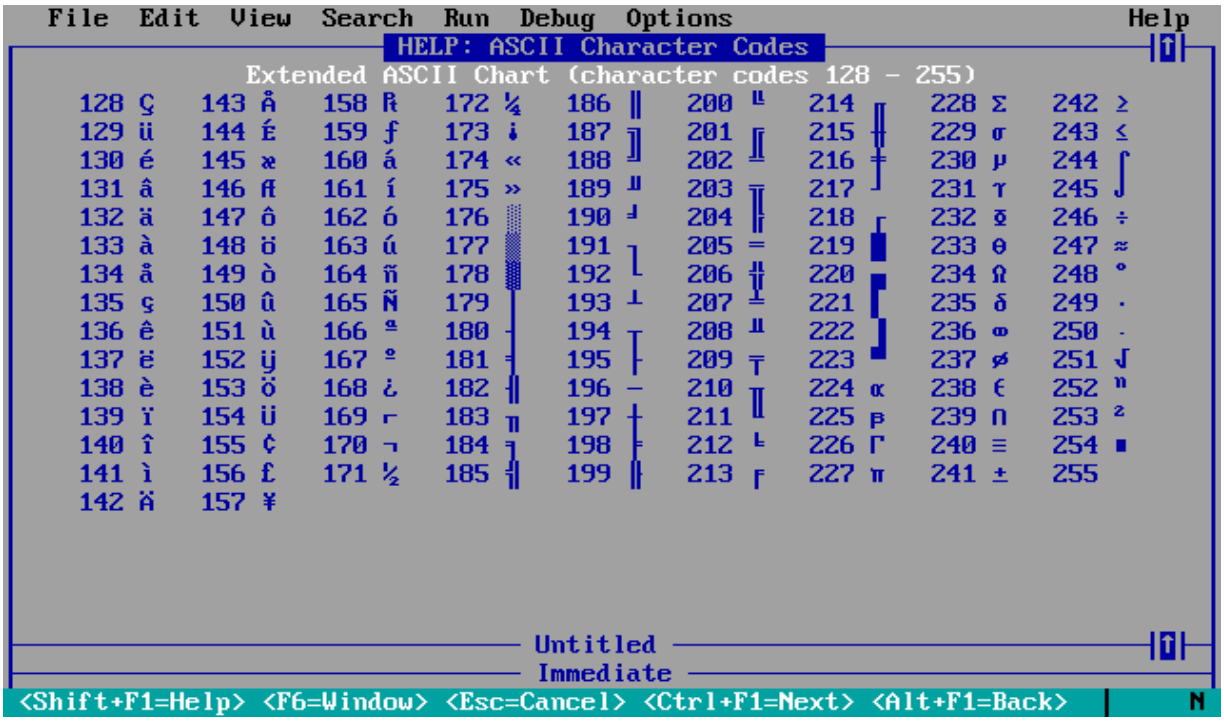
Help per onderwerp.



Help per trefwoord.



Help per programmeertaak.



Quick reference.

Debuggen programma

Indien u een programma gemaakt hebt, maar het levert niet het verwachte resultaat op, dan is er waarschijnlijk iets verkeerds geprogrammeerd. Een syntax fout is practisch uitgesloten omdat QBasic dit bij aanvang van het programma nog eens extra controleert. Hoogstwaarschijnlijk is een logische fout of redeneringsfout de oorzaak.
QBasic biedt een handige debugging om deze problemen op te sporen. U kunt met deze functies regel per regel het programma uitvoeren, waarbij na elke regel gewacht wordt. U kunt dan bijvoorbeeld variabelen ondervragen of eventueel nog veranderen.
U kunt ook met "breakpoints" werken. Het programma wordt dan full-speed uitgevoerd tot het de breakpoint tegenkomt, waarbij het dan onderbroken wordt. U kunt hier dan ook variabelen controleren en/of aanpassen.
Wanneer het programma onderbroken wordt of afgelopen is, kunt u de schermoutput nog bekijken door op F4 te drukken.

Oefening

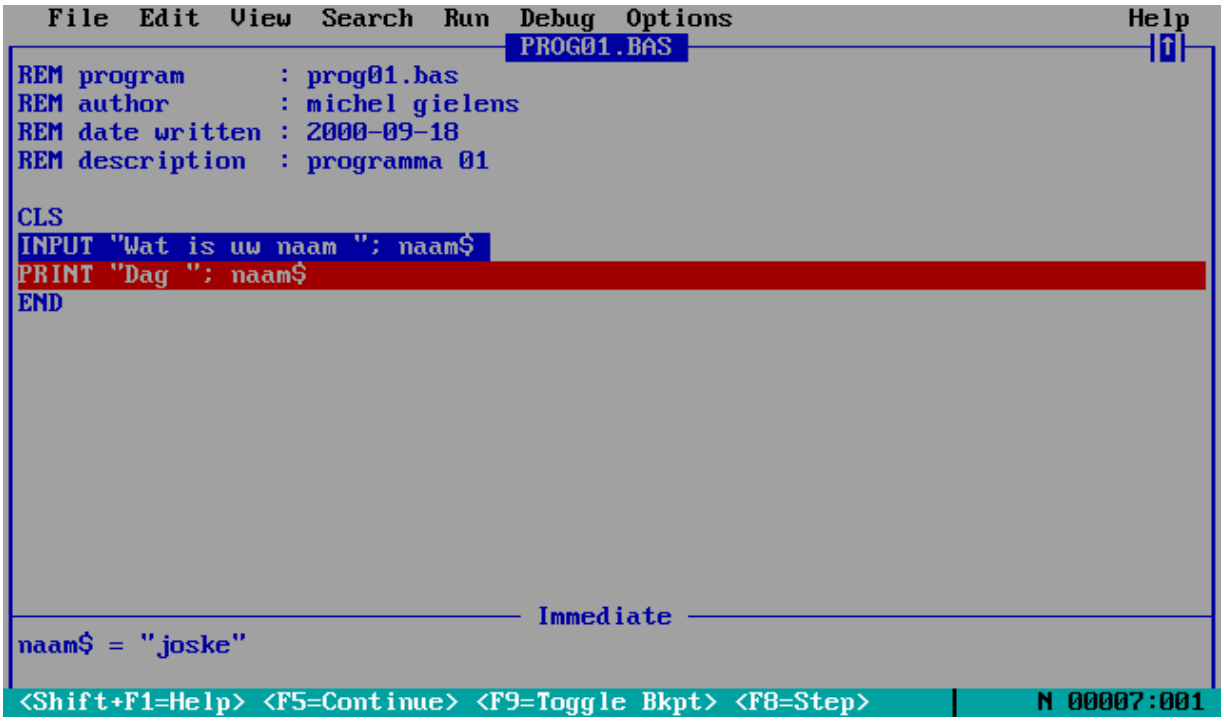
Laad programma 01.
Met behulp van F8 voert u regel per regel uit. (de regel die zal uitgevoerd worden, is weergegeven in het wit)
Voer `CLS` uit.
Voer `INPUT` uit.
Druk op F6 (om in het immediate venster te komen) en typ `print naam$`.
De inhoud van `naam$` wordt nu op scherm getoond.
Druk terug op F6 om in het programma te komen.
Voer `PRINT` uit.
Druk op F4 om de scherm-output nog eens te bekijken.
Voer `END` uit.

Oefening

Laad programma 01.
Ga met de cursor op de instructie `PRINT` staan en druk op F9.
Deze regel licht nu rood op: ze bevat een breakpoint.

Druk F5. Het programma wordt nu uitgevoerd tot de breakpoint.
Druk F6 en typ `naam$ = "joske"`.
Druk F5 om het programma verder uit te voeren.
U merkt dat de variabele `naam$` een andere inhoud heeft gekregen.

Debug
Rode regel = breakpoint
Blauwe regel = step by step uitvoering



Constanten

Constanten bevatten informatie die tijdens het uitvoeren van het programma niet verandert. Er zijn 2 soorten constanten: alfanumerieke en numerieke constanten.

Een **alfanumerieke constante** (of string-constante) kan bestaan uit letters, cijfers en andere tekens en staan tussen dubbele quotes. Vb. "Computer" en "2000". Hoewel het laatste voorbeeld enkel uit cijfers bestaat, is het niet bruikbaar voor berekeningen.

Een **numerieke constante** is een natuurlijk of reëel getal. De komma wordt in QBasic en vele andere talen weergegeven als een decimale punt. Gebruik dus niet 3,1416 maar 3.1416 .
QBasic kent 4 soorten numerieke constanten:
integers: zijn gehele getallen waarvan de waarde ligt tussen -32768 en +32767.
long integers: zijn gehele getallen waarvan de waarde ligt tussen -2147483648 en 2147483647.
single-precision getallen: zijn reële getallen, meestal weergegeven in E-notatie. De waarde ligt:
positieve getallen: tussen 2.802597E-45 en 3.402823E+38
negatieve getallen: tussen -3.402823E+38 en -2.802597E-45
double-precision getallen: zijn reële getallen, meestal weergegeven in D-notatie. De waarde ligt:
positieve getallen: tussen 4.940656458412465D-324 en 1.79769313486231D+308
negatieve getallen: tussen -1.79769313486231D+308 en -4.940656458412465D-324

Hoe definiëren?

```
CONST naam = "Els Joos"
```
: voor alfanumerische constanten.
```
CONST pi = 3.141593
```
: voor numerische constanten.

Variabelen

Een variabele is een grootheid waarvan de waarde kan veranderen tijdens het uitvoeren van het programma. Elke variabele krijgt een naam, die de computer in staat stelt de waarde van de variabele terug te vinden. Er zijn 2 soorten variabelen: alfanumerieke en numerieke variabelen. (alfanumerieke variabelen worden in QBasic ook "strings" genoemd.)

In QBasic moeten variabelen niet gedefinieerd worden, maar u doet dat beter wel. Het voordeel is dat QBasic niet moet "gokken" wat u bedoelt, de programmeur weet achteraf ook sneller terug wat bepaalde variabelen kunnen inhouden en het is een betere manier van programmeren.

Naamgeving variabelen:
- namen van variabelen moeten met een letter beginnen;
- namen mogen tot 40 tekens lang zijn;
- gereserveerde woorden mogen niet als naam voor variabele gebruikt worden.

Hoe definiëren?

strings:
`DIM naam AS STRING*30` : creëert de variabele "naam" van het type string en kan maximum 30 tekens bevatten.
`DIM adres AS STRING` : creëert de variabele "adres" van het type string en heeft een variabele lengte.
`naam$ = "Phoebe"` : creëert de variabele "naam" van het type string, heeft geen specifieke lengte en bevat hier de waarde "Phoebe".

integers:
`DIM aantal AS INTEGER` : creëert de variabele "aantal" van het type integer.
`aantal% = 13` : creëert de variabele "aantal" van het type integer en geeft er de waarde 13 aan.

long integers:
`DIM crc AS LONG` : creëert de variabele "crc" van het type long integer.
`crc& = 307200` : creëert de variabele "crc" van het type long integer en geeft er de waarde 307200 aan.

single-precision getallen:
`DIM prijs AS SINGLE` : creëert de variabele "prijs" van het type single precision.
`prijs! = 2.56` : creëert de variabele "prijs" van het type single precision en geeft er de waarde 2.56 aan.

double-precision getallen:
`DIM afstand AS DOUBLE` : creëert de variabele "afstand" van het type double precision.
`afstand# = 3.7685297133467E-7` : creëert de variabele "afstand" van het type double precision en geeft er de waarde 3.7685297133467E-7 aan.

Hint: definieer alle variabelen bovenaan het programma of bovenaan elke functie of procedure d.m.v. de `DIM` instructie.

Limieten

	Minimum	Maximum
Name, String, Number		
Variable-name length	1 character	40 characters
String length	0 characters	32,767 characters
Integers	-32,768	32,767
Long integers	-2,147,483,648	2,147,483,647
Single-precision numbers:		
Positive	2.802597E-45	3.402823E+38
Negative	-3.402823E+38	-2.282597E-45
Double-precision numbers:		
Positive	4.940656458412465D-324	1.79769313486231D+308
Negative	-1.79769313486231D+308	-4.940656458412465D-324
Arrays		
Array size (all elements):		
Static	1 byte	65,535 bytes (64K)
Dynamic		65,535 bytes (64K)
Number of dimensions allowed	1	60
Dimensions allowed if unspecified	1	8
Array subscript value	-32,768	32,767
Procedure and File		
Procedure size	0	64K
Number of arguments passed	0	60
Data file numbers	1	255
Data file record number	1	2,147,483,647
Data file record size	1 byte	32K
Data file size	0	Available disk space
Path names	1 character	127 characters
Error message numbers	1	255

Uitdrukkingen

Uitdrukkingen (alfanumeriek of numeriek) geven aan:
- welke *bewerkingen* dienen te gebeuren (optellen, aftrekken, delen, vermenigvuldigen, ...)
- op welke *variabelen of constanten* die bewerkingen dienen te worden toegepast
- in welke *volgorde* deze bewerkingen dienen te worden toegepast

In feite is een bewerking te beschouwen als een formule die we de computer opgeven. De uitdrukking noemen we alfanumeriek als het resultaat ervan alfanumeriek is. We noemen ze numeriek als het resultaat van de uitdrukking numeriek is.

Een uitdrukking kan volgende elementen bevatten: variabelen, constanten, bewerkingstekens, voorrangstekens (haakjes).

Algebraïsche bewerkingen

bewerking	betekenis
+	telt 2 numerieke uitdrukkingen bij elkaar op of plakt 2 alfanumerieke uitdrukkingen aan elkaar.
-	trekt de rechter numerieke uitdrukking van de linker af.
*	vermenigvuldigt 2 numerieke uitdrukkingen met elkaar.
/	deelt de linker numerieke uitdrukkingen door de rechter.
^	verheft de linker numerieke uitdrukking tot de macht die staat in de rechter uitdrukking.
\	rondt eerst de 2 numerieke uitdrukkingen af, deelt dan het linker gedeelte door het rechter en behoudt het quotiënt. (9 \ 2 = 4)
MOD	rondt eerst de 2 numerieke uitdrukkingen af, deelt dan het linker gedeelte door het rechter en behoudt de rest van de deling. (9 MOD 2 = 1)

Voorbeeld

```
REM program      : qb0801.bas
REM author       : michel gielens
REM date written : 2000-09-20
REM description  : test uitdrukkingen

DIM a AS SINGLE, b AS SINGLE
INPUT "Eerste waarde ", a
INPUT "Tweede waarde ", b
PRINT "a + b =", a + b
PRINT "a - b =", a - b
PRINT "a * b =", a * b
PRINT "a / b =", a / b
PRINT "a ^ b =", a ^ b
PRINT "a \ b =", a \ b
PRINT "a MOD b =", a MOD b
END
```

Functies

Functies zijn bewerkingen die 0 of meer argumenten aanvaarden en op basis daarvan een alfanumerieke of numerieke waarde teruggeven. QBasic heeft een aantal ingebouwde functies, maar de programmeur kan zijn specifieke functies eraan toevoegen. Het aantal argumenten is afhankelijk van de functie zelf.

Voorbeeld

```
REM program      : qb0802.bas
REM author       : michel gielens
REM date written : 2000-09-20
REM description  : test functies

PRINT INT(13.5)
PRINT LEFT$("Stokvis", 4)
PRINT SIN(3.1416)
END
```

Relationele bewerkingen

QBasic kan uitdrukking geven aan een relatie. De uitkomst van deze uitdrukking is altijd waar of onwaar. Waar wordt door -1 voorgesteld, onwaar door 0.

bewerking	betekenis
=	is gelijk aan
<	is kleiner dan
>	is groter dan
<>	is ongelijk aan
<=	is kleiner dan of gelijk aan
>=	is groter dan of gelijk aan

Voorbeeld

```
REM program      : qb0803.bas
REM author       : michel gielens
REM date written : 2000-09-20
REM description  : relationele bewerkingen

PRINT 13 = 13
PRINT (11 + 1) = (6 * 2)
PRINT (5 + 7) = 11
PRINT "CCMS" = "CC" + "MS"
PRINT "Bill" = "Billy"
END
```

Voorbeeld

```
REM program      : qb0804.bas
REM author       : michel gielens
REM date written : 2000-09-20
REM description  : relationele bewerkingen

DIM a AS INTEGER, b AS INTEGER
INPUT "Geef de waarde van a ", a
INPUT "Geef de waarde van b ", b
IF (a = b) THEN PRINT "a = b"
IF (a < b) THEN PRINT "a < b"
IF (a > b) THEN PRINT "a > b"
IF (a <> b) THEN PRINT "a <> b"
IF (a <= b) THEN PRINT "a <= b"
IF (a >= b) THEN PRINT "a >= b"
END
```

Logische bewerkingen

Een logische bewerking vergelijkt uitdrukkingen met elkaar en doet vervolgens een uitspraak in de zin van waar of onwaar. Volgende 'waarheidstabel' geeft 6 logische operatoren weer:

	Conditie 1	Conditie 2	Resultaat
AND	waar waar onwaar onwaar	waar onwaar waar onwaar	waar onwaar onwaar onwaar
OR	waar waar onwaar onwaar	waar onwaar waar onwaar	waar waar waar onwaar
XOR	waar waar onwaar onwaar	waar onwaar waar onwaar	onwaar waar waar onwaar
EQV	waar waar onwaar onwaar	waar onwaar waar onwaar	waar onwaar onwaar waar
IMP	waar waar onwaar onwaar	waar onwaar waar onwaar	waar onwaar waar waar
NOT	waar onwaar		onwaar waar

Volgend programma illustreert het gebruik van deze operatoren

```
REM program      : qb0805.bas
REM author       : michel gielens
REM date written : 2000-09-20
REM description  : logische bewerkingen

DIM a AS INTEGER, b AS INTEGER
INPUT "Hoeveel is 2+3 "; a
INPUT "Hoeveel is 4+5 "; b
IF (a = 5 AND b = 9) THEN PRINT "Beide sommen goed."
IF (a = 5 OR b = 9) THEN PRINT "1 of 2 sommen goed."
IF (a = 5 XOR b = 9) THEN PRINT "Slechts 1 som goed."
IF NOT (a = 5 OR b = 9) THEN PRINT "Beide sommen fout."
IF (a = 5 EQV b = 9) THEN PRINT "Beide sommen goed of beide sommen
fout."
IF NOT (a = 5 IMP b = 9) THEN PRINT "Eerste som goed, maar tweede
som fout."
END
```

Onthoud van deze logische bewerkingen enkel de basisbewerkingen AND, OR, NOT. Deze worden het meeste gebruikt.
De XOR, EQV, IMP bewerkingen komen zelden voor in logische bewerkingen. Veel mensen kennen ze niet en ik raad u aan deze dan ook niet te gebruiken. Ze kunnen trouwens ook met de basisbewerkingen weergegeven worden.

```
A XOR B = A AND NOT B OR NOT A AND B
A EQV B = A AND B OR NOT A AND NOT B
A IMP B = NOT A OR B
```

Volgorde van bewerkingen

Qbasic voert bewerkingen in volgende volgorde uit:

bewerkingen	opmerkingen
`()`	van binnen naar buiten
`^`	
`* / \ MOD`	van links naar rechts
`+ -`	van links naar rechts
`NOT`	
`AND`	
`OR XOR EQV IMP`	van links naar rechts

Voorbeeld: de bewerking '4 / 2 ^ 3 + 1 / 2 * 5 - 6' wordt als volgt uitgevoerd:

stap 1: 4 / 2 ^ 3 + 1 / 2 * 5 - 6 wordt 4 / 8 + 1 / 2 * 5 - 6
stap 2: 4 / 8 + 1 / 2 * 5 - 6 wordt 0.5 + 1 / 2 * 5 - 6
stap 3: 0.5 + 1 / 2 * 5 - 6 wordt 0.5 + 0.5 * 5 - 6
stap 4: 0.5 + 0.5 * 5 - 6 wordt 0.5 + 2.5 - 6
stap 5: 0.5 + 2.5 - 6 wordt 3 - 6
stap 6: 3 - 6 wordt -3

Voorbeeld: met variabelen:

```
( c + d ) * 4 - 8 ^ 2 * ( a * b - 3 )
  +-1-+                   +-2-+
                +-4-+        +--3--+
    +---5---+     +-----6------+
        +-------7-------+
```

Oefening 2: zet om in QBASIC notatie

x^2

y^6

x^{i+1}

$a(b-c)$

$a+b-bc+d^2$

$\frac{a}{b} - \frac{2b}{c} + \frac{4cd}{e}$

$\frac{a}{bc}$

$ax^3 + bx^2 + cx + d$

$(ab)^4$

Oefening 3: in welke volgorde worden volgende berekeningen uitgevoerd?

6+a-4

6+a*4

6^2-a*4

(a-4)-a*4

a-b*c/d^e

a-(b*c)/d^2

(a/(3*b))^3

(b+(b^2-4*a*c))/(2*a)

((a/b)^2+(c/d)^2)/((a-b)*(c+d))

CLS / PRINT / COLOR / INPUT

CLS

CLS is de afkorting van CLear Screen: scherm leegmaken.

PRINT

PRINT betekent afdrukken.

De uitvoer gemaakt door de PRINT instructie wordt naar de standaard uitvoer geleid. In de meeste gevallen is dit het scherm.
De PRINT instructie kan 0 of meer (numerische of alfanumerische) uitdrukkingen afdrukken. Deze uitdrukkingen worden gescheiden door een ';' of een ','. Na de eventuele uitdrukkingen kan nog een ';' of een ',' staan.
De ';' plakt uitdrukkingen aan elkaar. Indien de uitdrukking een getal is, wordt een negatief getal voorafgegaan door een '-' en een positief getal door een spatie. Achter elk getal wordt ook nog een spatie geplaatst.
De ',' plaatst uitdrukkingen in kolommen. Door de komma wordt de instructie gegeven om naar de eerstvolgende tabulatorstop te springen. Vanaf daar wordt verder afgedrukt.
Indien na de PRINT-instructie een ';' of een ',' staat, blijft de cursor op deze positie staan en wordt vanaf daar verder gedrukt bij volgende PRINT-instructie. Als er geen ';' of ',' staat, gaat de cursor naar het begin van de volgende regel.

Voorbeeld

```
REM program      : qb0901.bas
REM author       : michel gielens
REM date written : 2001-01-11
REM description  : test print instructies

DIM a AS INTEGER, b AS INTEGER
DIM c AS STRING*4, d AS STRING*3

a = -12
b = 34
c = "stok"
d = "vis"

CLS

PRINT a
PRINT b
PRINT c
PRINT d
PRINT
PRINT a; b
PRINT b; a
PRINT c; d
PRINT c; " "; d
PRINT
PRINT a, b
PRINT c, d
PRINT a, c, b, d
PRINT
PRINT "wijn"; c
PRINT c, "oud"
PRINT
PRINT c;
PRINT a

END
```

De positie waar iets afgedrukt moet worden, kan ook op volgende wijze beïnvloed worden:

- `TAB`: vertelt op welke positie van de regel het resultaat zal afgedrukt worden.
  `PRINT TAB(34); "TAB-functie"`
  drukt de string "TAB-functie" af vanaf positie 34 op de regel.
  Let op: indien de cursor al voorbij de opgegeven tabstop is, wordt eerst naar een nieuwe regel gegaan en daarna naar de opgegeven tabstop!
- `SPC`: drukt een aantal spaties af.
  `PRINT "appels"; SPC(20); "peren"`
  drukt "appels" en 20 spaties en "peren" af.
- `LOCATE regel, positie`: plaatst de cursor op een bepaalde plaats om vanaf daar verder te drukken.
  `LOCATE 12, 39 : PRINT "x"`
  drukt een kruisje ongeveer in het midden van een 80x25 scherm.

Oefening 4

1. Bereken de omtrek van een cirkel met R = 13 cm. (Formule = 2 * PI * R)

2. Bereken de tijd Diest - Brussel indien men aan een gemiddelde snelheid rijdt van 90 km/u. Afstand = 70 km.

3. Bereken het gemiddelde van 3 zelfgekozen constanten.

4. Zet 30° Celsius om in ° Fahrenheit. (Formule: F = 9/5 * C + 35)

COLOR

COLOR betekent kleur.

De instructie color wijzigt de kleuren die op het scherm getoond worden. Er is een verschil in het tonen van kleuren in tekst-mode en in grafische mode. We beginnen hier met de kleuren in tekstmode.

Deze instructie kan de voorgrondkleur of de achtergrondkleur of beide veranderen. Ze werkt als volgt:
`COLOR voorgrond, achtergrond`
waarbij voorgrond de voorgrondkleur is en achtergrond de achtergrondkleur.

Men moet minstens de voorgrond of de achtergrondkleur of beide opgeven. Indien men de achtergrond kleur opgeeft, moet men deze laten voorafgaan door een komma.

Vergeet niet dat een opgegeven kleur blijft gelden tot die kleur weer veranderd wordt.

Voorbeeld

```
REM program      : qb0906.bas
REM author       : michel gielens
REM date written : 2001-03-08
REM description  : test color

CONST black = 0
CONST blue = 1
CONST green = 2
CONST cyan = 3
CONST red = 4
CONST magenta = 5
CONST brown = 6
CONST lightgray = 7
CONST darkgray = 8
```

```
CONST lightblue = 9
CONST lightgreen = 10
CONST lightcyan = 11
CONST lightred = 12
CONST lightmagenta = 13
CONST yellow = 14
CONST white = 15

CLS
COLOR yellow, blue
LOCATE , 35
PRINT "+--------+"
LOCATE , 35
PRINT "| Welkom |"
LOCATE , 35
PRINT "+--------+"
COLOR lightgray, black

END
```

INPUT

INPUT betekent invoer / ingave.

Met de INPUT instructie kunnen tijdens het uitvoeren van het programma gegevens worden aangeboden via het toetsenbord of via de standaard input. Deze gegevens worden in variabelen opgeslagen. De gegevens kunnen zowel numerisch als alfanumerisch zijn.
Er kunnen meerdere gegevens achter elkaar ingegeven worden, maar dan dienen ze gescheiden te worden door een komma.

Voorbeelden

```
INPUT a%
PRINT a%
```

geeft bij uitvoeren:

```
? 13 (Computer plaatst een vraagteken)
 13
```

```
INPUT a$
PRINT "Hallo "; a$
```

geeft bij uitvoeren:

```
? Joos
Hallo Joos
```

```
INPUT a%, b%, c%
PRINT a%; b%; c%
```

geeft bij uitvoeren:

```
? 1,2,3
 1   2   3
```

Bij het INPUT statement kan ook een prompt geplaatst worden. Deze wordt dan eerst afgedrukt waarna de gebruiker de variabele(n) kan ingeven.
Indien na de prompt een ';' staat, wordt er na de prompt een vraagteken afgedrukt.
Indien na de prompt een ',' staat, wordt er geen vraagteken na de prompt afgedrukt.

Voorbeelden

```
INPUT "Wat is uw naam "; naam$
PRINT "Hallo "; naam$
```

geeft bij uitvoeren:

```
Wat is uw naam ? Michel
Hallo Michel
```

```
INPUT "Geef uw naam : ", naam$
PRINT "Hallo "; naam$
```

geeft bij uitvoeren:

```
Geef uw naam : Michel
Hallo Michel
```

Na het INPUT statement gaat QBASIC automatisch naar een nieuwe regel. U kunt dit onderdrukken door voor de prompt een ';' te plaatsen.

Voorbeeld

```
INPUT ;"Wat is uw naam "; naam$
PRINT "  Hallo "; naam$
```

geeft bij uitvoeren:

```
Wat is uw naam ? Michel  Hallo Michel
```

Mogelijke fouten bij het INPUT statement:

- De gebruiker geeft te weinig gegevens.
  De computer reageert met "`Redo from start`" waarna de gebruiker opnieuw de gegevens moet invoeren.
- De gebruiker geeft te veel gegevens.
  De computer reageert met "`Extra ignored`" waarbij de extra gegevens genegeerd worden.
- De gebruiker geeft een alfanumerische waarde als er een numerische verwacht werd.
  De computer reageert met "`Redo from start`" waarna de gebruiker opnieuw de gegevens moet invoeren.

Opgelet: deze foutmeldingen kunnen per BASIC-dialect verschillen.

Voorbeeld

```
REM program      : qb0907.bas
REM author       : michel gielens
REM date written : 2001-01-11
REM description  : inoefening cls, print, input instructies

DIM straal AS SINGLE, omtrek AS SINGLE
CONST PI = 3.1416

CLS
INPUT "Geef de straal van de cirkel in cm : ", straal
omtrek = 2 * PI * straal
PRINT
PRINT "De omtrek van een cirkel met een straal van "; straal; "cm is
"; omtrek; "cm."

END
```

Oefening 5

Je hebt 2 testen afgelegd.
Je geeft beide vakken en respectievelijke resultaten op. (input instructie)
Je telt de resultaten op en drukt de vakken, resultaten en het totaal af. (print instructie)
Layout:

```
+-----------------------------------------+
|                 rapport                 |
|                 -------                 |
| vak                           resultaat |
| ---                           --------- |
| ..............................    ...    |
| ..............................    ...    |
|                                  -----  |
| totaal                            ...   |
| ------                                  |
+-----------------------------------------+
```

Oefening 6

Een kapitaal K brengt jaarlijks een intrest op van I procent. Hoe groot zal dat kapitaal zijn na J jaren als er tussentijds geen geld afgehaald wordt? Formule samengestelde intrest: K*(1+I)^J
Layout:

```
+-------------------------------------------------+
|               kapitaalberekening                |
|               ------------------                |
|                                                 |
| kapitaal: .........   intrest: ... %   jaren: ... |
|                                                 |
| beginkapitaal                       eindkapitaal |
| -------------                       ------------ |
|   .........                          .........  |
+-------------------------------------------------+
```

IF / SELECT

IF THEN ELSE

IF betekent indien.
THEN betekent dan.
ELSE betekent anders.

Deze instructie kan in 2 vormen gebruikt worden:

```
IF (voorwaarde) THEN instructieblok1 ELSE instructieblok2
```

Indien aan de voorwaarde voldaan wordt, dan wordt instructieblok1 uitgevoerd, anders wordt instructieblok2 uitgevoerd.
Instructieblok 1 en 2 kunnen 0 of meer instructies bevatten, gescheiden door een ':', maar dit wordt zelden gedaan omdat dit de leesbaarheid negatief beïnvloed. Deze constructie wordt gebruikt om korte instructies op 1 regel te krijgen om alzo een compacter programma te verkrijgen.
De 'ELSE instructieblok2' is trouwens niet verplicht. Indien afwezig wordt enkel getest of de voorwaarde voldaan wordt.

```
IF (voorwaarde) THEN
  instructieblok1
ELSE
  instructieblok2
ENDIF
```

Indien aan de voorwaarde voldaan wordt, dan wordt instructieblok1 uitgevoerd, anders wordt instructieblok2 uitgevoerd.
Instructieblok 1 en 2 kunnen 0 of meer regels code bevatten.
Deze constructie wordt veel gebruikt om programma's zo overzichtelijk mogelijk te maken en omdat men op deze manier meerdere instructies in 1 blok kwijt kan.
De 'ELSE instructieblok2' is trouwens niet verplicht. Indien afwezig wordt enkel getest of de voorwaarde voldaan wordt. Bemerk als laatste instructie de `ENDIF`.

Opmerking: de voorwaarde moet altijd zo gesteld worden dat de uitkomst ervan steeds 'waar' of 'onwaar' is. (True of False in het engels). De uitkomst kan eventueel ook een getal zijn waarbij wordt aangenomen dat 0 onwaar is en alle andere getallen - verschillend van 0 - waar.

Voorbeeld

```
REM program      : qb1001.bas
REM author       : michel gielens
REM date written : 2001-01-14
REM description  : test if instructie

DIM getal AS INTEGER
CLS
INPUT "Geef een geheel getal : ", getal

REM 1e constructie
IF (getal > 100) THEN PRINT "Getal > 100" ELSE PRINT "Getal <= 100"

REM 2e constructie
IF (getal > 100) THEN
  PRINT "Getal > 100"
ELSE
  PRINT "Getal <= 100"
ENDIF

END
```

Oefening 7

Bereken het loon van iemand als men volgende gegevens krijgt:
- uurloon
- aantal gewerkte uren
- aantal werkuren in een week
De overuren worden altijd aan 150 % betaald van het normale uurloon.

SELECT CASE

SELECT betekent selecteer.
CASE betekent geval.

De SELECT-CASE is een veredelde IF-instructie.
Met deze constructie kunnen op een simpele, overzichtelijke manier verschillende tests uitgevoerd worden.

```
SELECT CASE (testexpressie)
  CASE expressielijst1
    statementblok-1
  CASE expressielijst2
    statementblok-2
  ...
  CASE ELSE
    statementblok-n
END SELECT
```

testexpressie	elke numerisch of alfanumerisch expressie
expressielijst1	1 of meer expressies om te vergelijken met de testexpressie
expressielijst2	
statementblok-1	0 of meer instructies op 1 of meerdere regels
statementblok-2	
statementblok-n	

De expressielijsten hebben volgend formaat of een combinatie van volgende formaten, gescheiden door een komma:
- expressie[,expressie]...
- expressie TO expressie
- IS relationele-operator expressie

expressie = elke numerisch of alfanumerisch expressie vergelijkbaar met de testexpressie.
relationele-operator = 1 van de volgende relationele operatoren: <, <=, >, >=, <>, of =.

Minstens 1 case is men verplicht te gebruiken.
De CASE ELSE constuctie is facultatief en wordt uitgevoerd indien geen enkele CASE aan de testexpressie voldoet.

Voorbeeld

```
REM program      : qb1003.bas
REM author       : michel gielens
REM date written : 2001-03-09
REM description  : select - case

DIM getal AS INTEGER

CLS

INPUT "Geef een geheel getal van 1 tot en met 10 : ", getal
```

```
SELECT CASE (getal)
  CASE 1, 2, 3, 5, 7
    PRINT getal; "is een priemgetal."
  CASE 4, 6, 8, 10
    PRINT getal; "is deelbaar door 2."
  CASE 9
    PRINT getal; "is deelbaar door 3."
  CASE ELSE
    PRINT "U heeft het waarschijnlijk niet goed begrepen!"
END SELECT

END
```

Oefening 8

De toegangsprijs aan de ingang van een pretpark is afhankelijk van de leeftijd van de bezoeker.
- normale prijs = 500 BEF
- kinderen < 3 jaar : gratis
- kinderen tussen 3 en 8 jaar : 50% korting
- 65 plussers : 50% korting
- speciale actie : wie 25 jaar is, krijgt 25% korting

Maak een programma dat de leeftijd van de persoon vraagt en dat door middel van de select-case instructie de toegangsprijs voor die persoon geeft.

FOR / WHILE / LOOP

FOR NEXT

FOR betekent voor/gedurende.
NEXT betekent volgende.

Indien een aantal instructies een specifiek aantal keren moet uitgevoerd worden, kan men deze instructies binnen in een for-next lus plaatsen.

De instructie wordt zo gebruikt:

```
FOR teller = start TO einde STEP verhoging
  instructieblok
NEXT teller
```

teller = de numerische variabele die gebruikt wordt als lus-teller.
start = de beginwaarde van teller.
einde = de eindwaarde van teller.
verhoging = wordt na elke doorloop van de lus bij de teller opgeteld (afgetrokken indien negatief)
instructieblok = 0 of meer instructies die telkens uitgevoerd moeten worden.

Het gedeelte 'STEP verhoging' is men niet verplicht van op te geven. Doet men het niet, dan wordt aangenomen dat de verhoging telkens 1 is.

Dit zijn de stappen die bij een for-next lus worden ondernomen:
1. De teller wordt eerst de waarde van start gegeven.
2. Overschrijdt de teller de waarde van einde, dan wordt er verdergegaan met de instructie volgend op het next-gedeelte.
3. Het instructieblok uitgevoerd.
4. De teller wordt met de waarde van 'verhoging' verhoogd (of verlaagd indien negatief).
5. Ga verder met punt 2.

Het is de gewoonte om het gedeelte binnen een for-next lus een beetje te laten inspringen. Dit is voor de programmeur visueel veel beter te volgen.

Voorbeeld

```
REM program      : qb1101.bas
REM author       : michel gielens
REM date written : 2001-03-25
REM description  : for-next voorbeeld

DIM i AS INTEGER

CLS

PRINT "De even getallen van 1 tot 10 zijn :";
FOR i = 2 TO 10 STEP 2
  PRINT i;
NEXT i
PRINT

PRINT "De oneven getallen van 20 tot 11 zijn :";
FOR i = 19 TO 11 STEP -2
  PRINT i;
NEXT i
PRINT

END
```

Het is toegestaan dat for-next lussen genest worden.
Dit wil zeggen dat een for-next lus binnen een andere for-next lus gebruikt wordt.
Ze mogen enkel niet gekruist gebruikt worden.

```
REM dit is goed
FOR y = 1 TO 25
  FOR x = 1 TO 80
    REM de instructies
  NEXT x
NEXT y

REM dit is fout
FOR y = 1 TO 25
  FOR x = 1 TO 80
    REM de instructies
  NEXT y
NEXT x
```

Voorbeeld

```
REM program      : qb1102.bas
REM author       : michel gielens
REM date written : 2001-03-25
REM description  : for-next voorbeeld 2

DIM x AS INTEGER, y AS INTEGER

CLS

PRINT TAB(8); "Tafels van vermenigvuldiging"
PRINT TAB(8); "----------------------------"
PRINT

FOR y = 1 TO 10
  REM druk de titel als y = 1
  IF (y = 1) THEN
    PRINT " X |   1   2   3   4   5   6   7   8   9  10"
    PRINT "---+----------------------------------------"
  END IF
  PRINT USING "## |"; y;
  FOR x = 1 TO 10
    PRINT USING " ###"; x * y;
  NEXT x
  PRINT
NEXT y

END
```

De `PRINT USING` constructie is hier nieuw: ze dient om o.a. de getallen mooi gealigneerd (in tabellen) weer te geven. Elk #-teken wordt daarbij vervangen door een decimaal van het getal. Zie voor meer uitleg in de QBASIC help files.

Voorbeeld

```
REM program      : qb1103.bas
REM author       : michel gielens
REM date written : 2001-03-25
REM description  : for-next voorbeeld 3 : kadertje

DIM vanx AS INTEGER, totx AS INTEGER
DIM vany AS INTEGER, toty AS INTEGER
DIM x AS INTEGER, y AS INTEGER

CLS

REM linkerbovenhoek
vanx = 10: vany = 5
```

```
REM rechteronderhoek
totx = 70: toty = 20

REM bovenste regel
LOCATE vany, vanx
PRINT "+";
FOR x = vanx + 1 TO totx - 1
  PRINT "-";
NEXT x
PRINT "+";

REM middelste regels
FOR y = vany + 1 TO toty - 1
  LOCATE y, vanx: PRINT "|"
  LOCATE y, totx: PRINT "|"
NEXT y

REM onderste regel
LOCATE toty, vanx
PRINT "+";
FOR x = vanx + 1 TO totx - 1
  PRINT "-";
NEXT x
PRINT "+";

END
```

Oefening 9

Druk een gradentabel met in de linkerkolom de graden in Fahrenheit gaande van 0 tot en met 100 graden in stapjes van 5 en in de rechterkolom de graden in Celsius. (formule : F=9/5*C+35)

WHILE WEND

WHILE betekent terwijl.
WEND is de samentrekking van while end.

Indien instructies moeten uitgevoerd worden zolang aan een voorwaarde voldaan is, kunnen ze in een while-wend lus geplaatst worden.

De instructie wordt zo gebruikt:

```
WHILE (voorwaarde)
  instructieblok
WEND
```

voorwaarde = een expressie die als resultaat waar of onwaar moet geven.
instructieblok = 0 of meer instructies die telkens moeten uitgevoerd worden.

Dit zijn de stappen die bij een while-wend lus worden ondernomen:
1. Wordt aan de voorwaarde voldaan, ga dan verder met stap 2, anders ga verder met de instructie volgend op het wend-gedeelte.
2. Het instructieblok uitgevoerd.
3. Ga verder met punt 1.

Het is de gewoonte om het gedeelte binnen een while-wend lus een beetje te laten inspringen. Dit is voor de programmeur visueel veel beter te volgen.

Voorbeeld

```
REM program      : qb1105.bas
REM author       : michel gielens
REM date written : 2001-03-25
REM description  : while-wend voorbeeld

DIM getal AS INTEGER

CLS

INPUT "Raad een geheel getal tussen 1 en 10 : ", getal
WHILE (getal <> 6)
  PRINT "Fout geraden."
  INPUT "Raad een geheel getal tussen 1 en 10 : ", getal
WEND
PRINT "U heeft het geraden!"

END
```

DO LOOP

DO betekent doen.
LOOP betekent lus.

Indien instructies moeten uitgevoerd worden zolang of totdat aan een voorwaarde voldaan is, kunnen ze in een do-loop lus geplaatst worden.

De do-loop is de betere versie van de while-wend.

De instructie kan op 2 manieren gebruikt worden:

1e manier:

```
DO WHILE/UNTIL voorwaarde
  instructieblok
LOOP
```

voorwaarde = een expressie die als resultaat waar of onwaar moet geven.
instructieblok = 0 of meer instructies die telkens moeten uitgevoerd worden.

2e manier:

```
DO
  instructieblok
LOOP WHILE/UNTIL voorwaarde
```

voorwaarde = een expressie die als resultaat waar of onwaar moet geven.
instructieblok = 0 of meer instructies die telkens moeten uitgevoerd worden.

Ofwel wordt het woordje WHILE (=zolang) gebruikt, ofwel het woordje UNTIL (=totdat).
Bij gebruik van 'WHILE voorwaarde' wordt het instructieblok uitgevoerd zolang aan de voorwaarde voldaan wordt.
Bij gebruik van 'UNTIL voorwaarde' wordt het instructieblok uitgevoerd totdat aan de voorwaarde voldaan wordt.

Dit zijn de stappen die bij een do-loop lus worden ondernomen:

1e manier: WHILE voorwaarde
1. Wordt aan de voorwaarde voldaan, ga dan verder met stap 2, anders ga verder met de instructie volgend op het loop-gedeelte.
2. Het instructieblok wordt uitgevoerd.
3. Ga verder met punt 1.

1e manier: UNTIL voorwaarde
1. Wordt aan de voorwaarde voldaan, ga dan verder met de instructie volgend op het loop-gedeelte.
2. Het instructieblok wordt uitgevoerd.
3. Ga verder met punt 1.

2e manier: WHILE voorwaarde
1. Het instructieblok wordt uitgevoerd.
2. Wordt aan de voorwaarde voldaan, ga dan verder met stap 1.

2e manier: UNTIL voorwaarde
1. Het instructieblok wordt uitgevoerd.
2. Wordt aan de voorwaarde voldaan, ga dan verder met de instructie volgend op het loop-gedeelte, anders ga verder met stap 1.

Het is de gewoonte om het gedeelte binnen een do-loop lus een beetje te laten inspringen. Dit is voor de programmeur visueel veel beter te volgen.

Voorbeeld

```
REM program      : qb1106.bas
REM author       : michel gielens
REM date written : 2001-03-25
REM description  : do-loop voorbeeldjes

DIM i AS INTEGER

CLS

PRINT "DO WHILE ... LOOP"
i = 0
PRINT "begin lus: i ="; i
PRINT "in de lus: i =";
DO WHILE (i < 5)
  PRINT i;
  i = i + 1
LOOP
PRINT
PRINT "einde lus: i ="; i

PRINT
PRINT "DO UNTIL ... LOOP"
i = 0
PRINT "begin lus: i ="; i
PRINT "in de lus: i =";
DO UNTIL (i < 5)
  PRINT i;
  i = i + 1
LOOP
PRINT
PRINT "einde lus: i ="; i

PRINT
PRINT "DO ... LOOP WHILE"
i = 0
PRINT "begin lus: i ="; i
PRINT "in de lus: i =";
DO
  PRINT i;
  i = i + 1
```

```
    LOOP WHILE (i < 5)
    PRINT
    PRINT "einde lus: i ="; i
    PRINT
    PRINT "DO ... LOOP UNTIL"
    i = 0
    PRINT "begin lus: i ="; i
    PRINT "in de lus: i =";
    DO
      PRINT i;
      i = i + 1
    LOOP UNTIL (i < 5)
    PRINT
    PRINT "einde lus: i ="; i

    END
```

Het resultaat is:

```
DO WHILE ... LOOP
begin lus: i = 0
in de lus: i = 0  1  2  3  4
einde lus: i = 5

DO UNTIL ... LOOP
begin lus: i = 0
in de lus: i =
einde lus: i = 0

DO ... LOOP WHILE
begin lus: i = 0
in de lus: i = 0  1  2  3  4
einde lus: i = 5

DO ... LOOP UNTIL
begin lus: i = 0
in de lus: i = 0
einde lus: i = 1
```

Oefening 10

De gebruiker geeft een paswoord in.
Daarna wordt het scherm gewist en wordt er constant het paswoord gevraagd totdat het juiste paswoord ingegeven wordt.

DATA / READ / RESTORE

DATA READ RESTORE

DATA betekent gegevens.
READ betekent lezen.
RESTORE betekent terugzetten.

Indien in een programma heel wat variabelen moeten gedefinieerd worden, kan men dit doen door de variabelen te definiëren en er telkens een waarde aan toe te kennen.
Dit is nochtans wat eenvoudiger te realiseren met behulp van de DATA en READ instructies.
DATA definieert de waarden en READ stopt deze waarden in de variabelen.

Het gebruik van elk van deze instructies:

```
DATA constante, constante, constante, ...
```

Constante is een numerische of alfanumerische constante. De constanten worden elk gescheiden door een komma.
Alfanumerische constanten hoeven hier niet door aanhalingstekens (" ") omsloten te worden, maar indien ze komma's of dubbele punten bevatten of beginnen of eindigen met spaties, is dit wel verplicht.

```
READ variabele, variabele, variabele, ...
```

De READ instructie leest de constanten die opgegeven zijn in de DATA regels en stopt deze in de variabelen. De variabele moet wel overeenkomen met het soort constante dat is opgegeven. Dus numerische variabelen gebruiken als er een getal moet gelezen worden en alfanumerische variabelen gebruiken als er strings gelezen moeten worden. Indien daar tegen gezondigd wordt, krijg je een syntax error.

```
RESTORE label
```

De RESTORE instructie laat toe om de constanten opgegeven in de DATA regels opnieuw in te lezen vanaf een bepaalde plaats.
Is 'label' gebruikt, dan zal QBASIC het label zoeken in het programma om vanaf daar bij de eerstvolgende READ instructie terug DATA gegevens op te halen.
Is de RESTORE instructie gebruikt zonder opgave van een label, dan zal bij de eerstvolgende READ instructie terug de allereerste DATA gegevens op halen.

Het gebruik van labels is hier nieuw.
Bij vroegere versies van Basic moest elke lijn beginnen met een regelnummer.
Dit was nodig omdat veel gestructureerde instructies nog niet bestonden en er nogal (te) veel met GOTO's gewerkt werd.
Tegenwoordig bezit QBASIC veel gestructureerde instructies (FOR-NEXT, WHILE-WEND, DO-LOOP, CALL) zodat regelnummers overbodig werden. Nochtans zijn ze voor sommige instructies toch nog nodig (vb. RESTORE, GOSUB). Ze zijn daarom vervangen door de meer leesbare labels.
Het label is nu dus een niet-gereserveerd woordje gevolgd door een dubbele punt aan het begin van een programma-regel. Bijvoorbeeld

```
gegevens:
DATA boter,kaas,eieren
```

Voorbeeld

```
REM program      : qb1201.bas
REM author       : michel gielens
REM date written : 2001-03-28
REM description  : data-read voorbeeld

REM Het aantal constanten is op voorhand bekend.

DIM dag AS STRING * 9
DIM i AS INTEGER

DATA maandag,dinsdag,woensdag,donderdag,vrijdag,zaterdag,zondag

FOR i = 1 TO 7
  READ dag
  PRINT dag
NEXT i

END
```

Voorbeeld

```
REM program      : qb1202.bas
REM author       : michel gielens
REM date written : 2001-03-28
REM description  : data-read-restore voorbeeld

REM Het aantal constanten is op voorhand niet bekend.
REM Dit probleem wordt opgelost door gebruik te maken van een
REM speciale constante, ook wel eens 'sentinel' genoemd.

DIM n AS INTEGER

DATA 1,2,3,4,5
tag:
DATA 6,7,8,9,0

READ n
DO WHILE (n <> 0)
  PRINT n;
  READ n
LOOP
PRINT

RESTORE tag

READ n
DO WHILE (n <> 0)
  PRINT n;
  READ n
LOOP
PRINT

END
```

Oefening 11

Stop de ledenlijst (lidnummer + naam + eventueel andere gegevens) van de computerclub in een reeks data-regels. Lees deze lijst d.m.v. de read instructie en toon de lidnummers en de namen op het scherm.
Maak gebruik van de 'sentinel' zodat deze ledenlijst simpel uit te breiden is.

Arrays

Arrays

Arrays zijn lijsten of tabellen.

Indien in uw programma veel numerische of alfanumerische constanten worden gebruikt, dan kunnen deze in een lijst of tabel opgenomen worden. Tot nu toe hebben we hiervoor gebruik gemaakt van de DATA instructie. De constanten werden dan met behulp van de READ instructie uit de tabel in een variabele gelezen. Nadeel van deze methode is dat bij elke READ instructie de vorige gelezen variabele vergeten werd.

QBASIC heeft een manier om deze constanten of variabelen in een rij of tabel (=array) te stoppen, waarbij elk element in deze array onmiddellijk beschikbaar is en blijft.

```
DIM ar1(10) AS INTEGER
```

Definieert een tabel met naam ar1 van het type INTEGER bestaande uit 11 elementen (van 0 t/m 10).

```
DIM ar2!(11 TO 15)
```

Definieert een tabel met naam ar2 van het type SINGLE ('!') bestaande uit 5 elementen (van 11 t/m 15).

De algemene definitie is als volgt:

```
DIM variabele(subscript) AS type
```

variabele: is de zelfgekozen naam voor de variabele.
subscript: zijn de dimensies van de array:
van TO tot
Waarbij 'van' de ondergrens van de subscript is en 'tot' de bovengrens.
Indien 'van TO' niet vermeld wordt, is de ondergrens 0.
Voor meerdere dimensies wordt het 'van TO tot' gedeelte herhaald, telkens met een komma ertussen.
type: is het type van de array (INTEGER, SINGLE, ...). 'AS type' moet niet gedefinieerd worden, maar dan moet het type wel duidelijk gemaakt worden door de extensie van de variabele.

De array kan tijdens het uitvoeren van het programma geherdefinieerd worden door het REDIM statement. Let op: de waarden van de oude array gaan hierbij verloren.

De algemene definitie is als volgt:

```
REDIM variabele(subscript) AS type
```

variabele, subscript en type worden op dezelfde manier gebruikt als bij de `DIM` instructie.

De waarde van elk element van de array kan benaderd worden door zijn index.
Voorbeeld: `PRINT ar(5)` print array-element nummer 5.

Wanneer een array niet langer nodig is, kan ze gewist worden. Hiermee bespaart u geheugenruimte.

De instructie is als volgt:

```
ERASE arraynaam
```

arraynaam is hier de naam van de variabele die als array gedefinieerd werd door een vorige DIM of REDIM instructie.

Voorbeeld

```
REM program      : qb1301.bas
REM author       : michel gielens
REM date written : 2001-04-04
REM description  : arrays

DIM dag$(1 TO 7)
DIM i AS INTEGER

DATA maandag,dinsdag,woensdag,donderdag,vrijdag,zaterdag,zondag

CLS

REM inlezen van de dagen
FOR i = 1 TO 7
  READ dag$(i)
NEXT i

REM afdrukken van de dagen
FOR i = 1 TO 7
  PRINT dag$(i)
NEXT i

END
```

Let op:
Voor elk element uit de array wordt geheugenruimte gereserveerd. Grote arrays kunnen dus zeer veel geheugenruimte innemen. U kunt geheugen besparen door de volgende tips toe te passen:

- Reserveer niet meer elementen als hetgeen er echt nodig is. Indien 12 elementen nodig zijn (vb. voor elke maand van het jaar) dan is het voldoende om dit te schrijven:
  `DIM maand(1 TO 12) AS INTEGER`
- Houd rekening met het type variabele. Als de array-elementen enkel gehele getallen gaan bevatten, moet de array het type INTEGER hebben.
  `DIM ar(99) as INTEGER` neemt (99 - 0 + 1) * 2 = 200 bytes in.
  `DIM ar(99) as SINGLE` neemt (99 - 0 + 1) * 4 = 400 bytes in.
  `DIM ar(99) as DOUBLE` neemt (99 - 0 + 1) * 8 = 800 bytes in.
- Indien een tabel niet langer nodig is, verwijder ze dan met de ERASE instructie.

Meerdimensionale arrays

Arrays kunnen meerdere dimensies bevatten.
Om bijvoorbeeld een tweedimensionale array te maken, definieert u het volgende:

```
DIM ar(1, 3) AS INTEGER
```

De geheugenruimte die dan gereserveerd wordt, zou er zo uitzien:

	0	**1**	**2**	**3**
0	ar(0,0)	ar(0,1)	ar(0,2)	ar(0,3)
1	ar(1,0)	ar(1,1)	ar(1,2)	ar(1,3)

De definities die gelden voor ééndimensionale array gelden ook voor de meerdimensionale array. Enkel het aantal dimensies wordt opgegeven door middel van de subscript. (de verschillende groottes van de dimensies worden gescheiden door een komma).

Voorbeeld

```
REM program      : qb1302.bas
REM author       : michel gielens
REM date written : 2001-04-06
REM description  : meerdimensionale arrays

DIM karakter(9, 6) AS STRING * 5
DIM getal(2) AS INTEGER
DIM x AS INTEGER, y AS INTEGER
DIM tijd AS LONG

REM lees karakters in een array
FOR y = 0 TO 6
  FOR x = 0 TO 9
    READ karakter(x, y)
  NEXT x
NEXT y

REM initialiseer getal
FOR x = 0 TO 2
  getal(x) = 0
NEXT x

CLS

DO
  REM 1 seconde vertraging
  REM De variabele 'TIMER' geeft het aantal seconden
  REM dat verstreken is sinds middernacht.
  tijd = INT(TIMER)
  DO
  LOOP UNTIL (INT(TIMER) <> tijd)
  REM druk het getal bestaande uit 3 tekens
  LOCATE 1, 1
  FOR y = 0 TO 6
    FOR x = 0 TO 2
      PRINT karakter(getal(x), y); " ";
    NEXT x
    PRINT
  NEXT y
  PRINT
  REM verhoog het getal met 1
  x = 2
  DO
    getal(x) = getal(x) + 1
    IF (getal(x) > 9) THEN
      getal(x) = 0
    ELSE
      x = 0
    END IF
    x = x - 1
  LOOP UNTIL (x < 0)
LOOP WHILE (1)

END

DATA " xxx "," x "," xxx "," xxx "," x ","xxxxx"," xx ",
"xxxxx"," xxx "," xxx "
DATA "x x"," xx ","x x","x x"," xx ","x ","x ","x x",
"x x","x x","x x"
DATA "x xx","x x ","x ","x ","x x ","xxx ","x ","x x",
" x ","x x","x x"
DATA "x x x"," x ","x x ","xxxxx","x x ","xxxx ",
" x "," xxx "," xxxx"
DATA "xx x"," x "," xx "," x ","x x","x x",
" x ","x x","x "," x"
DATA "x x"," x ","x ","x x","x x","x x","x x",
" x ","x x","x "
DATA " xxx ","xxxxx","xxxxx"," xxx ","x x","xxx ","xxx ",
" x "," xxx "," xx "
```

Oefening 12

Gegeven:
Veronderstel dat er 6 leerlingen zijn en 4 vakken.
Dit zijn de DATA-regels in het programma:

```
DATA Anja,Bart,Carla,Dirk,Els,Fons
DATA wiskunde,9,7,5,6,9,6
DATA biologie,7,9,8,5,6,9
DATA fysica,9,7,5,9,5,7
DATA chemie,6,9,6,6,5,8
```

De eerste regel bevat de 6 namen van de leerlingen, de volgende regels bevatten telkens de naam van het vak en de resultaten voor elk van de leerlingen.

Gevraagd:
Druk een rapport met op elke regel de leerling met de resultaten per vak.
Druk ook het gemiddelde per leerling en het gemiddelde per vak.

Layout:

	wiskunde	biologie	fysica	chemie	(gemiddeld)
Anja	9.0	7.0	9.0	6.0	7.8
Bart	7.0	9.0	7.0	9.0	8.0
Carla	5.0	8.0	5.0	6.0	6.0
Dirk	6.0	5.0	9.0	6.0	6.5
Els	9.0	6.0	5.0	5.0	6.3
Fons	6.0	9.0	7.0	8.0	7.5
(gemiddeld)	7.0	7.3	7.0	6.7	

Grafisch

Opgelet: dit hoofdstuk handelt over de grafische capaciteiten van QBASIC.
- Uw PC heeft hiervoor minstens een grafische kaart nodig.
- Onder windows zult u normaal gezien full-screen moeten werken.

Laten we van de veronderstelling uitgaan dat iedereen tegenwoordig minstens een VGA-kaart heeft met 256K geheugen aan boord.
We gaan ook elke instructie niet in detail bekijken. Sommige parameters zal ik niet behandelen, omdat dit ons te lang zou bezig houden.

Eerst en vooral gaan we - nu de instructies ingewikkelder worden - de QBASIC syntax conventie bespreken. Deze beschrijft in feite de volledige instructie.

HOOFDLETTERS dit zijn sleutelwoorden van QBASIC en zijn zo verplicht in te geven (behalve als ze tussen vierkante haken staan).
kleine letters dit zijn de variabelen die nodig zijn voor de instructie. Het type van de variabele wordt vermeld door de suffix.
[optie] opties staan tussen vierkante haken en zijn dus niet verplicht te gebruiken.
{keuze1 | keuze2} accolades en een verticaal streepje duiden een verplichte keuze tussen 2 of meer items aan (behalve als dit tussen vierkante haken staat).
item, item, ... de 3 puntjes duiden aan dat meerdere items kunnen gebruikt worden.

SCREEN

Deze instructie verandert de scherm mode en wordt zo gebruikt:

```
SCREEN mode%
```

Deze modes zijn beschikbaar:

mode	type	resolution	colors	adaptor
0	text	40x25, 40x43, 40x50, 80x25, 80x50	16	MDPA, CGA, Hercules, Olivetti, EGA, VGA, MCGA
1	grafic	320x200, 40x25	4	CGA, EGA, VGA, MCGA
2	grafic	640x200, 80x25	2	CGA, EGA, VGA, MCGA
3	grafic	720x348, 80x25	1	Hercules, Olivetti, AT&T
4	grafic	640x400, 80x25	1	Hercules, Olivetti, AT&T
7	grafic	320x200, 40x25	16	EGA, VGA
8	grafic	640x200, 80x25	16	EGA, VGA
9	grafic	640x350, 80x25, 80x43	16	EGA, VGA
10	grafic	640x350, 80x25, 80x43	4	EGA, VGA, Monochrome
11	grafic	640x480, 80x30, 80x60	2	VGA, MCGA
12	grafic	640x480, 80x30, 80x60	16	VGA, MCGA
13	grafic	320x200, 40x25	256	VGA, MCGA

Voorbeeld

`SCREEN 12` schakelt over naar mode 12 met een resolutie van 640x480 pixels en 16 kleuren.

Coordinaten-systeem: het standaard assenstelsel heeft zijn oorsprong in de linkerbovenhoek van het scherm met coordinaat (0,0). De X-as loopt van linksboven naar rechtsboven; de Y-as loopt van linksboven naar linksonder.

Grafische cursor: in grafische mode wordt er ook een soort cursor bijgehouden. Deze is niet zichtbaar op scherm, maar houdt toch bij waar de laatste grafische actie plaatsgevonden had. De locatie van de laatste pixel die gewijzigd werd, is dus altijd bekend. Hiervan kunnen we bij sommige instructies gebruik maken.

COLOR

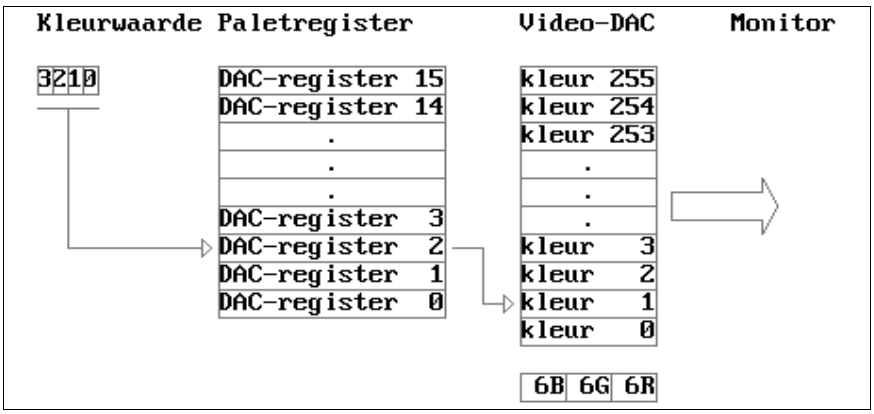
Wijzigt, zoals in tekstmode, de kleuren van voorgrond en achtergrond die getoond zullen worden.

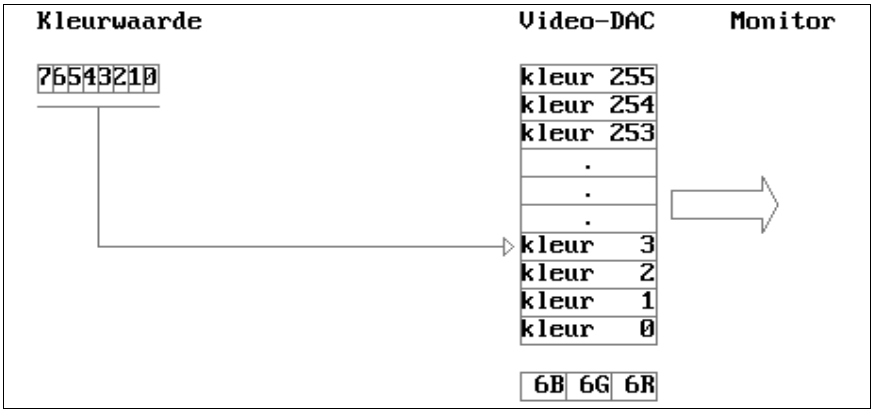
De instructie wordt zo gebruikt:

```
COLOR [voorgrond%] [,achtergrond%]
```

Men kan de voorgrondkleur, de achtergrondkleur of beide opgeven. Indien de achtergrondkleur wordt opgegeven, moet men een komma in de instructie plaatsen.
De achtergrond kan men enkel opgegeven indien deze ook effectief ondersteund wordt door de screen-mode.

In feite wordt niet de kleur opgegeven, maar het kleurattribuut. Er wordt namelijk intern een tabel bijgehouden met nummers van kleuren en het daarbij horende RGB kleurenpalet. De kleur die met `COLOR` gekozen wordt, is in principe een kleurnummer; de kleur die dat nummer representeert wordt via de PALETTE instructie veranderd.





PSET / PRESET

PSET is de afkorting van Point SET.
PRESET is de afkorting van Point RESET.

PSET wordt gebruikt om een puntje (een pixel) op het scherm te plaatsen.
PRESET wordt gebruikt om dat puntje terug te verwijderen.

De instructies worden zo gebruikt:

```
PSET [STEP] (x!,y!) [,kleur%]
PRESET [STEP] (x!,y!) [,kleur%]
```

PSET en PRESET kleuren beide de pixel op coordinaat (x,y) in de opgegeven kleur. Indien het facultatieve woordje 'STEP' gebruikt wordt, zal de coordinaat van de grafische cursor eerst bij de opgegeven coordinaat geteld worden.
De kleur is ook facultatief: wordt ze niet vermeld dan gebruikt PSET de huidige voorgrondkleur en PRESET de huidige achtergrondkleur.
x en y mogen eventueel buiten het scherm liggen. In dat geval wordt er niets gekleurd.

Voorbeeld

```
REM program      : qb1401.bas
REM author       : michel gielens
REM date written : 2001-04-10
REM description  : PSET-voorbeeldje

DIM x AS INTEGER, y AS INTEGER, kleur AS INTEGER

SCREEN 13

FOR y = 0 TO 199
  kleur = 0
  FOR x = 0 TO 319
    PSET (x, y), kleur
    kleur = kleur + 1
    IF (kleur > 255) THEN kleur = 0
  NEXT x
NEXT y

END
```

LINE

LINE tekent een lijn of een rechthoek op het scherm.

De instructie wordt zo gebruikt:

```
LINE [[STEP](x1!,y1!)]-[STEP](x2!,y2!) [,[kleur%] [,[B | BF] ]]
```

STEP betekent dat de coordinaten relatief zijn t.o.v. de grafische cursor.
(x1,y1) is de startcoordinaat en (x2,y2) is de eindcoordinaat van de lijn.
kleur is de kleur waarmee de lijn moet getekend worden. Indien afwezig, wordt de huidige voorgrondkleur genomen.
B : tekent een rechthoek i.p.v. een lijn.
BF : tekent een rechthoek i.p.v. een lijn en vult ze op met de gekozen kleur.

Voorbeeld

```
REM program      : qb1402.bas
REM author       : michel gielens
REM date written : 2001-04-10
REM description  : LINE-voorbeeldje

DIM x AS INTEGER, y AS INTEGER

SCREEN 13

FOR y = 0 TO 15
  FOR x = 0 TO 15
    LINE (20 * x, 10 * y)-(20 * x + 19, 10 * y + 9), 16 * y + x, BF
  NEXT x
NEXT y

END
```

CIRCLE

CIRCLE tekent cirkels of ellipsen op het scherm.

De instructie wordt zo gebruikt:

```
CIRCLE [STEP] (x!,y!),straal! [,[kleur%] [,[start!] [,[einde!]
[,afplatting!]]]]
```

STEP betekent dat de coordinaten relatief zijn t.o.v. de grafische cursor.
(x,y) zijn de coordinaten voor het middelpunt van de cirkel.
straal is de straal van de cirkel.
kleur is de kleur waarmee de cirkel moet getekend worden. Indien afwezig, wordt de huidige voorgrondkleur genomen.
start is de starthoek van de cirkel (in radialen).
einde is de eindhoek van de cirkel (in radialen).
afplatting is de verhouding van de x-as tot de y-as om ellipsen te tekenen.

Door de start- en/of de eindhoek van de cirkel op te geven kunnen we cirkelsegmenten tekenen. Een negatieve hoek zal hierbij nog een verbinding met het middelpunt van de cirkel tekenen.

Voorbeeld

```
REM program      : qb1403.bas
REM author       : michel gielens
REM date written : 2001-04-10
REM description  : CIRCLE voorbeeldje

DIM x AS INTEGER

CONST pi = 3.141593

SCREEN 12

COLOR 1
CIRCLE (50, 20), 20
CIRCLE (100, 20), 20, 2
CIRCLE (150, 20), 20, 3, pi * .25
CIRCLE (200, 20), 20, 4, , pi * .75
CIRCLE (250, 20), 20, 5, pi * .25, pi * .75
CIRCLE (300, 20), 20, 6, -(pi * .25), pi * .75
CIRCLE (350, 20), 20, 7, -(pi * .25), -(pi * .75)
CIRCLE (400, 20), 20, 8, pi * .75, pi * .25
CIRCLE (450, 20), 20, 9, -(pi * .75), -(pi * .25)
CIRCLE (500, 20), 20, 10, , , .5
```

```
CIRCLE (550, 20), 20, 11, , , 2

FOR x = 0 TO 10
  CIRCLE (320, 240), 200, 12, , , x / 10
  IF (x = 0) THEN ' voorkom deling door 0
    CIRCLE (320, 240), 200, 12, , , 1000
  ELSE
    CIRCLE (320, 240), 200, 12, , , 10 / x
  END IF
NEXT x

END
```

PAINT

PAINT betekent schilderen.
Door slechts 1 punt op te geven, en schilderkleur en een randkleur, kunt u ganse vlakken inkleuren.

De instructie wordt zo gebruikt:

```
PAINT [STEP] (x!,y!)[,[kleur%] [,[randkleur%]]]
```

STEP betekent dat de coordinaten relatief zijn t.o.v. de grafische cursor.
(x,y) zijn de coordinaten waar het inkleuren moet beginnen.
kleur is de kleur waarmee het vlak ingekleurd zal worden. Indien afwezig, wordt de huidige voorgrondkleur genomen.
randkleur is de kleur van de rand van het in te kleuren vlak. Van zodra QBASIC deze kleur tegenkomt, neemt het aan dat de rand van het te kleuren vlak bereikt is.

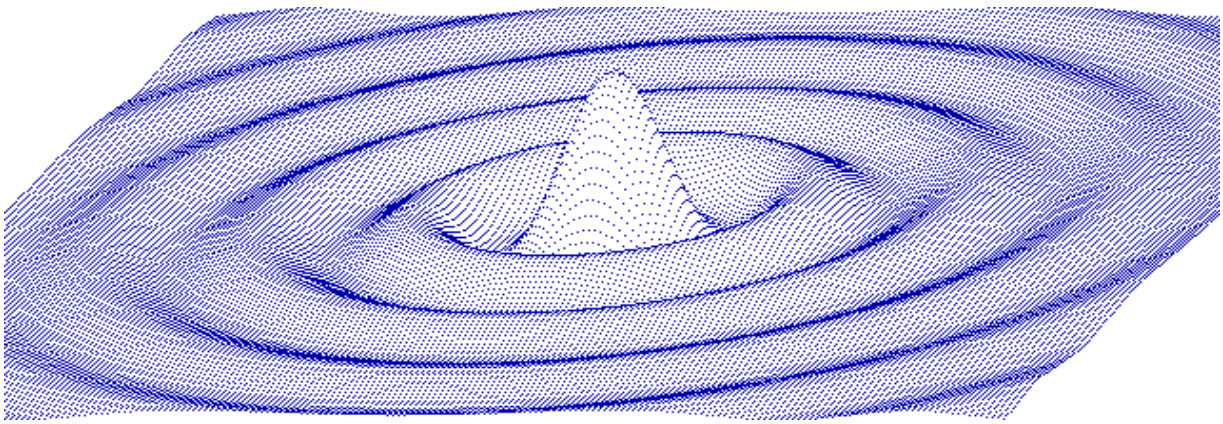
Voorbeeld

```
REM program      : qb1404.bas
REM author       : michel gielens
REM date written : 2001-04-10
REM description  : PAINT-voorbeeldje

CONST pi = 3.141593

SCREEN 13

PAINT (0, 0), 8

CIRCLE (160, 100), 100, 9
PAINT (160, 100), 9

CIRCLE (120, 75), 20, 0
PAINT (120, 75), 10, 0
CIRCLE (120, 75), 10, 0
PAINT (120, 75), 0

CIRCLE (200, 75), 20, 0
PAINT (200, 75), 10, 0
CIRCLE (200, 75), 10, 0
PAINT (200, 75), 0

CIRCLE (160, 100), 20, 11, , , 2
PAINT (160, 100), 11

CIRCLE (160, 100), 80, 12, pi * 1.2, pi * 1.8, .8
CIRCLE (160, 100), 80, 12, pi * 1.2, pi * 1.8, .6

END
```

Voorbeeld

Sommige functies zoals SIN (sinus), COS (cosinus), ATN (boogtangens) of SQR (vierkantswortel) zien we pas in deel 2 van de cursus.

```
REM program      : qb1405.bas
REM author       : michel gielens
REM date written : 2001-04-15
REM description  : sinus in 3D

CONST maxx = 639
CONST maxy = 349

DIM pi AS DOUBLE
DIM mult AS INTEGER
DIM x AS DOUBLE, y AS DOUBLE, z AS DOUBLE, r AS DOUBLE
DIM c AS DOUBLE, s AS DOUBLE
DIM sx AS INTEGER, sy AS INTEGER, px AS INTEGER, py AS INTEGER
DIM h(maxx) AS INTEGER

SCREEN 9                                ' 640x350 16 kleuren
PAINT (0, 0), 15                        ' witte achtergrond
COLOR 1                                 ' blauwe tekenkleur

pi = ATN(1) * 4                         ' definieer PI
c = .5 * COS(pi / 4)                    ' verschuiving x
s = .5 * SIN(pi / 4)                    ' verschuiving y
mult = 8                                ' vermenigvuldiger

FOR sx = 0 TO maxx                      ' initialiseer tabel met maxy
  h(sx) = maxy
NEXT sx

FOR sy = -maxx / 2 TO maxx / 2 STEP 3
  y = sy / (maxx / 2) * mult * pi       ' omrekenen schermcoordinaat y
  FOR sx = -maxx / 2 TO maxx / 2 STEP 3
    x = sx / (maxx / 2) * mult * pi     ' omrekenen schermcoordinaat x
    r = SQR(x * x + y * y)              ' bereken afstand tot oorsprong
    IF (r = 0) THEN                     ' vermijd deling door 0
      z = 0
    ELSE
      z = mult * SIN(r) / r             ' bereken coordinaat z
    END IF
    px = (maxx / 2) + sx + c * sy       ' terugrekenen x-coordinaat
    py = (maxy / 2) - s * sy - 10 * z   ' terugrekenen y-coordinaat
    IF (px >= 0 AND px < maxx) THEN     ' test coordinaat
      IF (py < h(px)) THEN              ' test overlappingen in y
        PSET (px, py)                   ' teken punt
        h(px) = py                      ' onthoud hoogte van y
      END IF
    END IF
  NEXT sx
NEXT sy

END
```



Oplossingen

oefening 1

cd \
dir
mkdir qbasic

oefening 2

x ^ 2

y ^ 6

x ^ (i + 1)

a * (b - c)

a + b - b * c + d ^ 2

a / b - 2 * b / c + 4 * c * d / e

a / (b * c)

a * x ^ 3 + b * x ^ 2 + c * x + d

(a * b) ^ 4

oefening 3

```
6 + a - 4
+-1-+
  +--2--+
```

```
6 + a * 4
    +-1-+
+--2--+
```

```
6 ^ 2 - a * 4
+-1-+
        +-2-+
  +---3---+
```

```
(a - 4) - a * 4
 +-1-+
          +-2-+
   +---3----+
```

```
a - b * c / d ^ e
             +-1-+
    +-2-+
      +---3---+
+----4----+
```

```
(b * c) / d ^ 2
 +-1-+
          +-2-+
   +---3----+


(a / (3 * b)) ^ 3
      +-1-+
 +--2---+
    +-----3-----+


(b + (b ^ 2 - 4 * a * c)) / (2 * a)
      +-1-+
               +-2-+
                 +--3--+
        +----4-----+
 +-----5-----+
                              +-6-+
       +-----------7-----------+

((a / b) ^ 2 + (c / d) ^ 2) / ((a - b) * (c + d))
  +-1-+
    +--2---+
                 +-3-+
                   +--4---+
       +------5------+
                                  +-6-+
                                        +-7-+
                                   +----8----+
               +-----------9------------+
```

oefening 4

1)

```
REM program      : oef04_1.bas
REM author       : michel gielens
REM date written : 2001-03-04
REM description  : bereken de omtrek van een cirkel met R = 13 cm

CONST pi = 3.141593

DIM straal AS SINGLE, omtrek AS SINGLE

straal = 13
omtrek = 2 * pi * straal
PRINT "De omtrek van een cirkel met een straal van"; straal; "cm
is"; omtrek; "cm."

END
```

2)

```
REM program      : oef04_2.bas
REM author       : michel gielens
REM date written : 2001-03-04
REM description  : bereken de tijd Diest - Brussel indien men aan
REM                een gemiddelde snelheid rijdt van 90 km/u.
REM                Afstand = 70 km.

DIM snelheid AS INTEGER, afstand AS INTEGER, tijd AS SINGLE

snelheid = 90                   ' km/u
afstand = 70                    ' km
tijd = afstand / snelheid       ' u

PRINT "Aan een gemiddelde snelheid van"; snelheid; "km/u ";
PRINT "kan men de afstand Diest-Brussel (="; afstand; "km) ";
PRINT "afleggen in"; tijd; "uur of"; tijd * 60; "minuten."

END
```

3)

```
REM program      : oef04_3.bas
REM author       : michel gielens
REM date written : 2001-03-04
REM description  : bereken gemiddelde van 3 zelfgekozen constanten

CONST c1 = 11
CONST c2 = 13
CONST c3 = 17

DIM gemiddelde AS SINGLE

gemiddelde = (c1 + c2 + c3) / 3

PRINT "Het gemiddelde van"; c1; ","; c2; "en"; c3; "is"; gemiddelde

END
```

4)

```
REM program      : oef04_4.bas
REM author       : michel gielens
REM date written : 2001-03-04
REM description  : zet 30 graden celsius om in graden fahrenheit
REM              : F = 9/5 * C + 35

DIM C AS SINGLE, F AS SINGLE

C = 30                  ' celsius
F = 9 / 5 * C + 35      ' fahrenheit

PRINT C; "graden celsius ="; F; "graden fahrenheit."

END
```

oefening 5

```
REM program      : oef05.bas
REM author       : michel gielens
REM date written : 2001-03-21
REM description  : drukt een rapport met 2 vakken en 2 resultaten

DIM vak1 AS STRING * 20, vak2 AS STRING * 20
DIM res1 AS INTEGER, res2 AS INTEGER
DIM totaal AS INTEGER

CLS

INPUT "Geef vak 1 en resultaat 1, gescheiden door een komma : ",
vak1, res1
INPUT "Geef vak 2 en resultaat 2, gescheiden door een komma : ",
vak2, res2

totaal = res1 + res2

PRINT
PRINT

PRINT "            rapport"
PRINT "            -------"
PRINT "vak                    resultaat"
PRINT "--------------------  ---------"
PRINT vak1; TAB(27); res1
PRINT vak2; TAB(27); res2
PRINT "                       ---------"
PRINT "totaal"; TAB(27); totaal

END
```

oefening 6

```
REM program      : oef06.bas
REM author       : michel gielens
REM date written : 2001-03-21
REM description  : kapitaalberekening samengestelde intrest

DIM beginkapitaal AS LONG, eindkapitaal AS LONG
DIM intrest AS SINGLE, jaren AS INTEGER

CLS

INPUT "Geef het beginkapitaal ... : ", beginkapitaal
INPUT "Geef het intrest % ..... : ", intrest
INPUT "Geef het aantal jaren .... : ", jaren

eindkapitaal = beginkapitaal * (1 + intrest / 100) ^ jaren

PRINT
PRINT

PRINT "              kapitaalberekening"
PRINT "              ------------------"
PRINT
PRINT "kapitaal :"; beginkapitaal; "  ";
PRINT "intrestvoet :"; intrest; "%  ";
PRINT "jaren :"; jaren
PRINT
PRINT TAB(10); "beginkapitaal"; TAB(25); "eindkapitaal"
PRINT TAB(10); "-------------"; TAB(25); "------------"
PRINT TAB(10); beginkapitaal; TAB(25); eindkapitaal

END
```

oefening 7

```
REM program      : oef07.bas
REM author       : michel gielens
REM date written : 2001-03-09
REM description  : berekening loon

DIM uurloon AS SINGLE, werkuren AS SINGLE, ureninweek AS SINGLE
DIM normloon AS SINGLE, overloon AS SINGLE, loon AS SINGLE

CLS

INPUT "Geef het uurloon ....................... : ", uurloon
INPUT "Geef het aantal gewerkte uren ........... : ", werkuren
INPUT "Geef het normale aantal uren per week ... : ", ureninweek

IF (werkuren <= ureninweek) THEN
  normloon = werkuren * uurloon
  overloon = 0
ELSE
  normloon = ureninweek * uurloon
  overloon = 1.5 * (werkuren - ureninweek) * uurloon
END IF
loon = normloon + overloon

PRINT
PRINT "Het loon bedraagt"; normloon; "+"; overloon; "="; loon

END
```

oefening 8

```
REM program      : oef08.bas
REM author       : michel gielens
REM date written : 2001-03-21
REM description  : inkom pretpark (select case)

DIM leeftijd AS INTEGER
DIM normaleprijs AS INTEGER, prijs AS INTEGER

normaleprijs = 500

CLS

PRINT TAB(32); "Welkom in Wabili"
PRINT TAB(32); "----------------"
PRINT
INPUT "Wat is uw leeftijd "; leeftijd
PRINT
SELECT CASE (leeftijd)
  CASE IS < 3
    PRINT "leeftijd < 3 jaar : inkom = gratis"
  CASE IS <= 8
    PRINT "leeftijd tss 3 en 8 jaar : inkom ="; normaleprijs * .5
  CASE IS >= 65
    PRINT "leeftijd > 65 : inkom ="; normaleprijs * .5
  CASE 25
    PRINT "leeftijd = 25 : inkom ="; normaleprijs * .75
  CASE ELSE
    PRINT "normale prijs : inkom ="; normaleprijs
END SELECT

END
```

oefening 9

```
REM program      : oef09.bas
REM author       : michel gielens
REM date written : 2001-04-01
REM description  : gradentabel

DIM F AS SINGLE, C AS SINGLE

CLS

PRINT "    Gradentabel"
PRINT "Fahrenheit   Celsius"
FOR F = 0 TO 100 STEP 5
  C = (F - 35) * 5 / 9
  PRINT USING "   ###        ###.##"; F; C
NEXT F

END
```

oefening 10

```
REM program      : oef10.bas
REM author       : michel gielens
REM date written : 2001-04-08
REM description  : User geeft een paswoord in.
REM                Daarna wordt scherm gewist en wordt het paswoord
REM                gevraagd totdat het juiste ingegeven wordt.

DIM pwd1 AS STRING, pwd2 AS STRING

INPUT "Geef het paswoord :", pwd1

CLS
```

```
DO
  INPUT "Wat is het paswoord "; pwd2
LOOP UNTIL (pwd1 = pwd2)

PRINT "U heeft het juiste paswoord ingegeven."

END
```

oefening 11

```
REM program      : oef11.bas
REM author       : michel gielens
REM date written : 2001-04-08
REM description  : Druk een ledenlijst.
REM                Maak gebruik van de sentinel om de lijst
REM                gemakkelijk uit te breiden.

DIM nr AS INTEGER
DIM naam AS STRING, email AS STRING, homepage AS STRING

CLS

PRINT "NR NAAM"
READ nr, naam, email, homepage
DO WHILE (nr <> -1)
  PRINT USING "## &"; nr; naam
  READ nr, naam, email, homepage
LOOP

DATA 1,"Karel Coopmans","Coopmans.karel@skynet.be",""
DATA 2,"Gerhard Mombers","eaglewarrior@skynet.be",""
DATA 3,"Raf Heselmans","rafhesel@yucom.be",""
DATA 4,"Marina Janssens","minajan@yucom.be",""
DATA 5,"Johan van Haarlem","",""
DATA 6,"Jasper Meyns","Jasper.meyns@advalvas.be",
"http://home.tiscalinet.be/jasperm"
DATA 7,"Gilbert Ignoul","",""
DATA 8,"Toon Van Nooten","vannooten@skynet.be",""
DATA 9,"Joseph Bauwens","Joseph.bauwens@belgacom.net",""
DATA 10,"Frans Van Hoeyvelt","On4bab@pi.be",""
DATA 11,"Sabine Sas","On4bab@pi.be",""
DATA 12,"Jean Nuyts","robernu@pi.be",""
DATA 13,"Edward Schoofs","Edward.schoofs@pi.be",""
DATA 15,"Jenny Smeyers","",""
DATA 16,"Rene Schroyen","rene.schroyen@yucom.be",""
DATA 17,"Michel Driesens","michel.driesens@advalvas.be",""
DATA 18,"Peter Vanderlinden","peter.vanderlinden2@pi.be",""
DATA 19,"Geert Conard","geert.conard@itconsult.be",
"http://www.itconsult.be"
DATA 20,"Freddy Dupont","f.dupont@pi.be",
"http://www.freddydupont.com"
DATA 21,"Leopold Van Rossem","",""
DATA 22,"Clement Myten","",""
DATA 23,"Dirk Blarinckx","dirk.blarinckx@pi.be",""
DATA 24,"Eddy Plackle","gamba@belgacom.net",""
DATA 25,"Michel Gielens","michel.gielens@skynet.be",
"http://users.skynet.be/michel.gielens"
DATA 26,"Davy Goris","davy.goris@advalvas.be",""
DATA 27,"Robert De Vroe","r.devroe@planetinternet.be",""
DATA 28,"Nick Haesevoets","dominique.haesevoets@vrt.be",""
DATA -1,"","",""

END
```

oefening 12

```
REM program      : oef12.bas
REM author       : michel gielens
REM date written : 2001-04-06
REM description  : rapport
```

```
REM Veronderstel dat er 6 leerlingen zijn en 4 vakken.
REM Druk een rapport met op elke regel de leerling met de
REM resultaten per vak.
REM Druk ook het gemiddelde per leerling en het gemiddelde per vak.

DIM lln(1 TO 6) AS STRING * 8
DIM vak(1 TO 4) AS STRING * 8
DIM res(1 TO 6, 1 TO 4) AS INTEGER
DIM gemlln(1 TO 6) AS SINGLE
DIM gemvak(1 TO 4) AS SINGLE
DIM illn AS INTEGER, ivak AS INTEGER

REM lees de namen van de leerlingen
FOR illn = 1 TO 6
  READ lln(illn)
NEXT illn

REM lees de naam van het vak + de resultaten per leerling
FOR ivak = 1 TO 4
  READ vak(ivak)
  FOR illn = 1 TO 6
    READ res(illn, ivak)
  NEXT illn
NEXT ivak

REM berekenen van de gemiddelden
FOR illn = 1 TO 6
  FOR ivak = 1 TO 4
    gemlln(illn) = gemlln(illn) + res(illn, ivak) / 4
    gemvak(ivak) = gemvak(ivak) + res(illn, ivak) / 6
  NEXT ivak
NEXT illn

REM afdruk rapport

CLS

REM hoofding
PRINT "",
FOR ivak = 1 TO 4
  PRINT USING "\       \ "; vak(ivak);
NEXT ivak
PRINT "(gemiddeld)"
PRINT

REM tabel
FOR illn = 1 TO 6
  PRINT lln(illn),
  FOR ivak = 1 TO 4
    PRINT USING " ##.#     "; res(illn, ivak);
  NEXT ivak
  PRINT USING " ##.#     "; gemlln(illn)
NEXT illn
PRINT

REM eindregel
PRINT "(gemiddeld)",
FOR ivak = 1 TO 4
  PRINT USING " ##.#     "; gemvak(ivak);
NEXT ivak
PRINT

END

DATA Anja,Bart,Carla,Dirk,Els,Fons
DATA wiskunde,9,7,5,6,9,6
DATA biologie,7,9,8,5,6,9
DATA fysica,9,7,5,9,5,7
DATA chemie,6,9,6,6,5,8
```