1097 ATE3 - **---**

blz 3

1 Deze samenvatting is alleen bruikbaar voor de opleiding van E3.
Men gebruikt in deze cursus ATE3 > dit staat voor AssembleerTaalvanE3, deze behoort tot de 2e generatie en staat verhoudingsgewijs dicht bij machinetaal.
Deze cursus is bedoeld om basisbeginselen/denkwijze van programmeren te leren.
Op de site bij dit vak is er ook een klein programma te vinden waarmee de programmacode te testen is. Het heet ATE3editor en zit ingepakt in zip vorm.

Daarnaast nog een opmerking bij de derde Casus, er staan nog apart enkele PSD diagrammen op de site die bij de opdracht horen, ook weer ingepakt als zip-file.

Daarnaast worden er ook PSD en PSS diagrammen in behandeld, dat is wel algemeen.

410

3.2 ATE3 Taalbeschrijving

Elke taal kan beschreven worden in de BNF vorm. BNF staat voor: Backus-Naur-Formalisatie, vastgelegd door John Backus en Peter Naur. De syntax van een programmeertaal wordt meestal vastgelegd in de ISO normering E-BNF (Extended BNF).
Zo ook ATE3. De beschrijving van de voorkomende notaties staan hierbeneden weergegeven. Een speciale opmerking wordt gemaakt ten aanzien van de [[]] in elke syntaxregel. Zowel binnen E-BNF als ATE3 heeft de notatie [] een bepaalde betekenis. Binnen E-BNF betekent [] dat de ertussen vermelde tekst optioneel is en binnen ATE3 betekent [] dat de tekst ertussen geen onderdeel van het programma is en genegeerd wordt bij de uitvoering ervan. De dubbele haken [[en]] betekenen dus niets meer dan dat de optionele ATE3-tekst, volgens E-BNF, optioneel is. Verwarrend, maar waar.

vrg:	trefwoord	trefwrd onderverdeling	omschrijving
1098	**Intern geheugen -**		*******

blz 4 1.1 **bits en bytes -**

BIT > Binary Digit = binair getal > kleinste geheugenelement in een computer en kan twee toestanden aannemen, we gaan er hier vanuit dat dit 0 of 1 is.
Het geheugen van een computer is te beschouwen als een doorlopende rij van bits.
Het intern geheugen ve computer is adresseerbaar.

BYTE > is een afkorting van "By Eight" > is een geheugenelement van 8 bits;
we gaan ervanuit dat ieder byte een uniek adres heeft.
De nummering v bits in een byte gaat van hoog naar laag

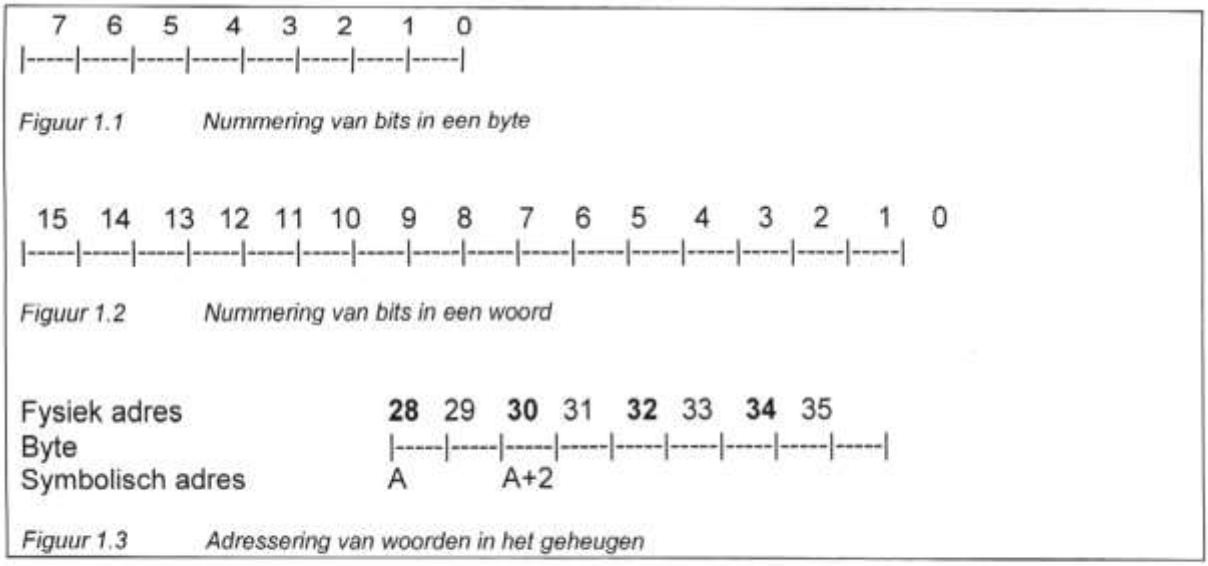
WOORD*** > is een samenvoeging van 2 Bytes > (volgende woord is 2 bytes verder). ***
het beginadres van een woord is een even getal.
Bv tabel TB1 van 5 woorden wordt gedeclareerd als TB1: DAB 10

Bits lezen we van rechts naar links en begint rechts bij 0
LEAST SIGNIFICANT BIT > is de meest rechtse bit 0
MOST SIGNIFICANT BIT > is de meest linkse bit 7

Plaats in het geheugen wordt als volgt weergegeven, drie manieren >
1. NUMERIEK ADRES > dit is mbv het adres vd byte die bedoeld wordt;
2. SYMBOLISCH ADRES of LABEL of ETIKET > dit kan met een symbolische naam die bekend gemaakt wordt dmv declareren.
3. Een symbolisch adres kan worden gebruikt voor verdere adressering.

ADDEND > is het getal dat aan een symbolisch adres kan worden toegevoegd.

370



Figuur 1.1 Nummering van bits in een byte

Figuur 1.2 Nummering van bits in een woord

Figuur 1.3 Adressering van woorden in het geheugen

Bij figuur 1.3 merken we nog het volgende op:
- een woord omvat de inhoud van twee adressen;
- het beginadres van een woord is een even getal.

1099	**Registers -**		

blz 6 1.2 **soorten -**

Om opdrachten te kunnen uitvoeren maakt een computer gebruik v registers.
REGISTERS > zijn speciaal daartoe aangewezen plaatsen in de CVE (centrale verwerkings eenheid). Deze kunnen per computer verschillen in grootte, de grootte is bepalend voor de functie vh register.

De volgende soorten zijn er >
- rekenregister;
- indexregister;
- conditieregister;
- instructiewijzer;
- instructieregister;
- stapelwijzer.

vrg:	trefwoord	trefwrd onderverdeling	omschrijving	
1100	**Registers -**			*******
blz 6 1.2.1	**rekeregister (RR) -**		We gaan er in deze cursus vanuit dat onze computer 7 rekenregisters heeft die elk 16 bits groot zijn. Elk rekenregister kan dus de inhoud van een geheugenwoord bevatten. De bits in elk rekenregister zijn genummer zoals de bits in een woord. Als bit 15 waarde 0 heeft > is het een positief getal.*** Als bit 15 de waarde 1 heeft > is het een negatief getal.*** De rekenregisters hebben vaste namen > R1, R2, ….R7 en gereserveerde ruimten. Deze namen mogen niet voor iets anders gebuikt worden bij variabelen.	
1101	**Registers -**			
blz 7 1.2.2	**indexregister -**		Rekenregisters kunnen worden gebruikt als een indexregister. INDEX > is een aanwijzer voor tabelelementen en men kan hiermee dus aangeven het hoeveelste element ve tabel wordt bedoelt.	
1102	**Registers -**			*******
blz 7 1.2.3	**conditieregister (CR) -**		CONDITIEREGISTER (CR) > bestaat uit 2 bits - bit 1 wordt NEGATIEVE BIT (N) genoemd - bit 2 wordt ZERO BIT (Z) genoemd CONDITIEREGISTER > heeft de volgende mogelijkheden: bit2 bit1 uitslag Z N 0 0 groter dan *** 0 1 kleinder dan 1 0 gelijk aan *** let op men begint links met bit 2 en "telt" weer terug	
1103	**Registers -**			
blz 8 1.2.4	**instructiewijzer (IW) -**		De INSTRUCTIEWIJZER (IW) is een register dat uit 16 bits bestaat en bevat steeds het adres vd volgende opdracht die uitgevoerd gaat worden.	
1104	**Registers -**			
blz 8 1.2.5	**instructieregister (IR) -**		Een programma bestaat uit programmaregels. Een programmaregel bevat een opdracht die uitgevoerd moet worden door de CVE. Elke opdracht wordt een instructie genoemd.	
1105	**Registers -**			**---**
blz 8 1.2.6	**stapelwijzer / stackpointer (SW) -**		STAPELWIJZER (SW) of STACKPOINTER > is een stapel v verwijzingen naar geheugenadressen en hier worden de adressen vd opdrachten bewaard. Door lussen (sprongopdrachten) kan het voorkomen dat we van de beschreven volgorde in een programma afwijken. Het SW zorgt er dan voor dat de instructies in de gewenste volgorde uitgevoerd worden ook al springen we hier en daar op en neer binnen onze instructies.	
1106	**Registers -**			
blz 8 1.2.7	**opdrachtencyclus -**		Een uit te voeren programma staat in het intern geheugen. Een OPDRACHTENCYSLUS bestaat uit de volgende acties > 1. haal de opdracht naar het IR (=instructieregister) 2. verhoog de IW (=instructiewijzer) met 4 3. analyseer de opdracht 4. voer de opdracht uit.	

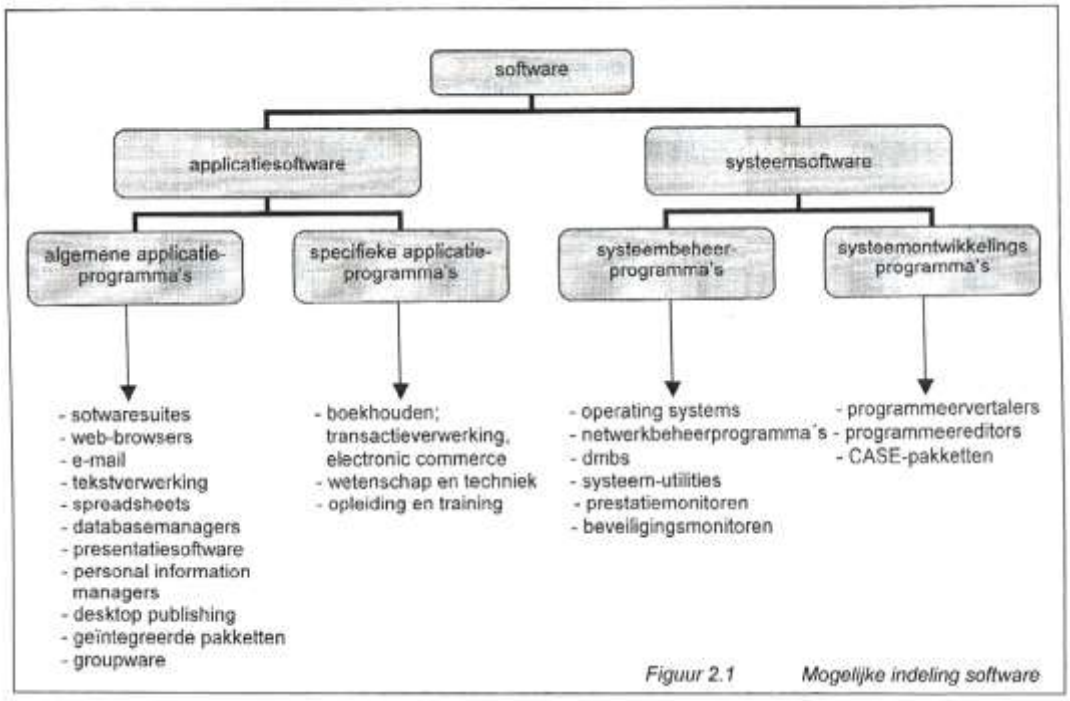
vrg:	trefwoord	trefwrd onderverdeling	omschrijving
1107	**Registers -**		
blz 9 1.2.8	**gereserveerde namen -**		(zoals oa. namen v registers) mogen niet voor andere aanduidingen gebruikt worden. De volgende namen zijn gereserveerd: - R1, R2…..R7 > dit zijn de rekenregisters (7 stuks binnen ATE3) - CR > conditieregister - IW > instructiewijzer - IR > instructieregister - SW > stapelwijzer of stackpointer en ook bv - LD > laden - SOM > optellen - OPS > opslaan e.d.
1108	**Soorten gegevens -**		
blz 10 1.3	**numerieke en alphanumerieke -**		NUMERIEKE GEGEVENS > zijn in het algemeen gehele getallen. Deze getallen worden binair weergegeven in 2 bytes, dit is dus gelijk aan 1 woord. De grootte v deze getallen zijn beperkt en hebben als grenswaarden: -32768 <= x <= 32767 dis is 2 tot de macht 16 en past precies in één woord (uiteraard zijn er ook oplossingen voor gebruik van grotere getallen, maar daar wordt in deze stof niet verder op ingegaan). ALFANUMERIEKE GEGEVENS of KARAKTER of CHARACTER > zijn letters, symbolen, leestekens en dergelijke. Deze tekens worden in ATE3 biniair weer gegeven mbv de ASCII-code. De binaire weergave bestaat uit 8 bits, is dus 1 byte.
1109	**Soorten gegevens -**		
blz 10 1.3.3	**declaratie -**		DECLARATIE > is een opdracht waarbij een gegeven in een geheugenadres geplaatst wordt. Door een declaratie krijgt een geheugenplaats een beginwaarde en dit kan een numeriek of alfanumeriek gegevens zijn. DECLAREREN > is tweeledig, men kent een naam én een waarde toe. In ATE3 heeft een declaratie de volgende opbouw > symbolisch adres : DC (alfa)numerieke waarde bv: variabele1: DC 241 of bv: variabele2: DC "Jantje" Hier wordt in variabele1 het getal 241 geplaatst en in variabele2 de tekst "Jantje" soorten declaratie opdrachten > DC > declareer een (alfa)numerieke waarde DG > declaratie van een aantal bytes DAB > declaratie van (een naam voor) een geheugenplaats
1111	**Bussen -**		
blz 12 1.5	**bussen -**		Voor de uitwisseling van gegevens maakt een computer gebruik van zgn bussen. BUSSEN > zijn zgn transportwegen voor gegevens tussen de op de bussen aangesloten apparaten en de volgende zitten normaal in een computer > - DATABUS > transportkanaal voor de gegevens zelf; - ADRESBUS > voor de adressering vd aangesloten locaties; - CONTROLEBUS > voor de besturing vd aangesloten apparaten (dit zijn hier muis, keyboard, disdrive, geheugens en CVE).
SE.1		**Hoofdstuk: 2 Programmeertalen**	

vrg:	trefwoord	trefwrd onderverdeling	omschrijving	
1112	**Generaties -**			**---**

blz 18 2.1.1 **basisprogrammatuur -**

hieronder verstaan we het besturingssysteem (operating system) en de datamangementsystemen. Een besturingssysteem bestaat uit een aantal programma's met de volgende functies > proces management / memory mangement / I/O mangement / data mangement / job mangement / library mangement / datacommunicatie mangement.

372



Figuur 2.1 Mogelijke indeling software

1113	**Generaties -**			**---**

blz 19 2.1.2 **databasemanagement programmatuur -**

Taak vh DBMS (DataBaseManagementSysteem) > besturing verzorgen tussen de DB en de toepassingen die vd de betreffende DB gebruik maken.

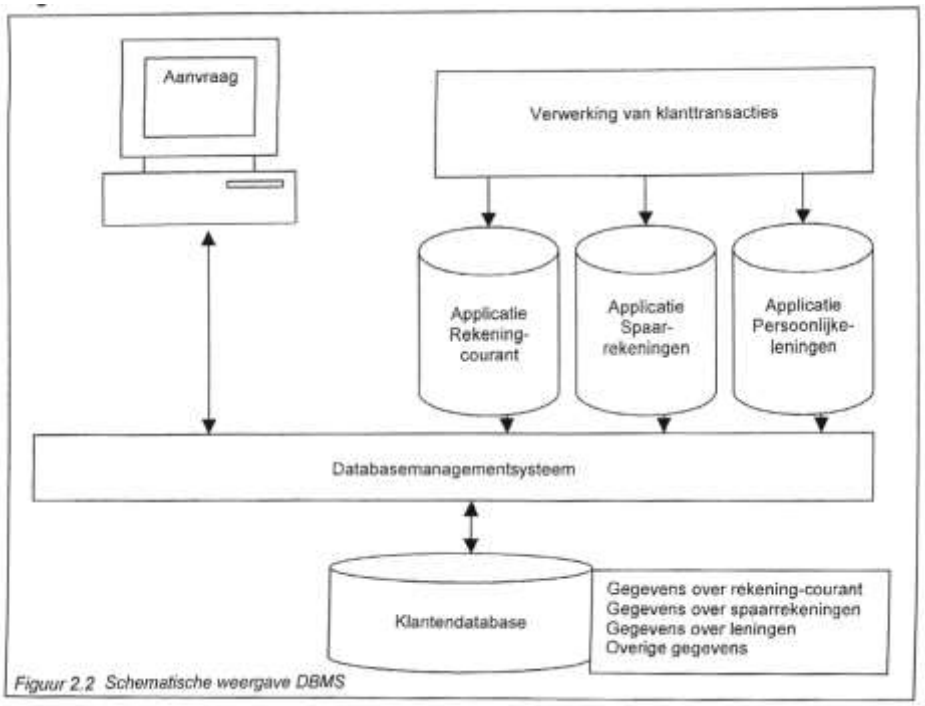
DDL = Data Definition Language > is de taal die gebruikt wordt om de structuur vd gegevensbank te definiëren.

DD = Data Dictionary > is de verzameling gebruikersgerichte beschrijvingen van karakteristieken van gegevens ter ondersteuning vh informatiemanagement.

Datadirectoy > is een verzameling computertechnische beschrijvingen van karakteristieken vd door het DBMS te beheren gegevens.

DML = Data Manupulation Language > is de taal voor het vullen vd DB met gegevens.

373

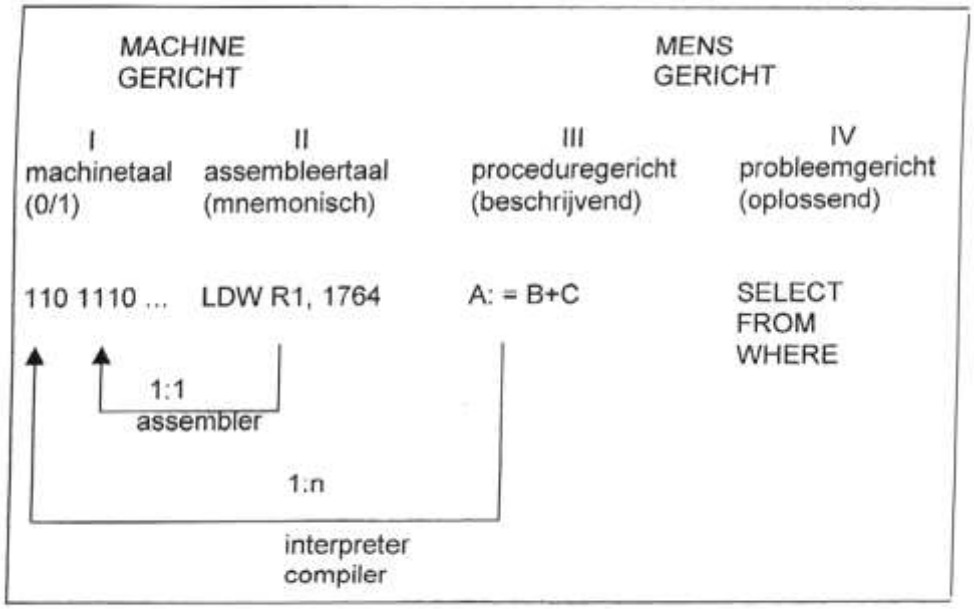


Figuur 2.2 Schematische weergave DBMS

1114 Generaties - **---**

blz 20 2.1.3 **generaties -** Software en hardware generaties lopen niet parallel.
De software generaties worden gebruikt om aan te geven in hoeverre mate de gebruikte software afstaat vd machine.

378



Figuur 2.4 Ontwikkeling van machinegericht naar mensgericht

377

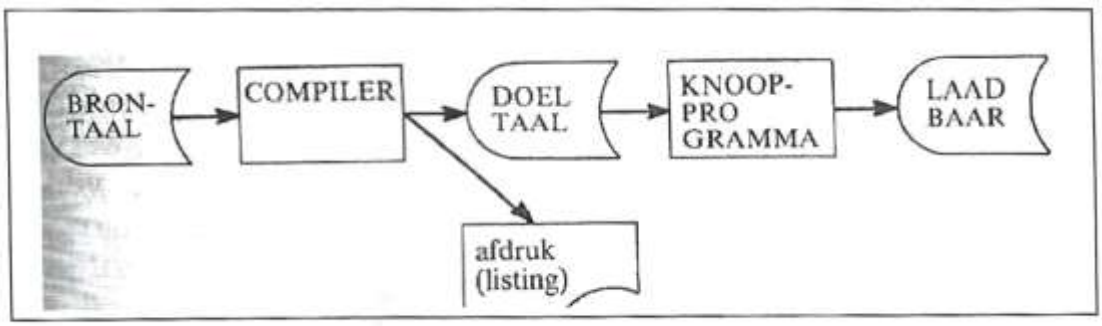
	Eerste generatie	Tweede generatie	Derde generatie	Vierde generatie	Vijfde generatie
Ontwikkeling: richting programmeertalen met natuurlijke conversatie					
Ontwikkeling in software	Door gebruikers geschreven programma's Machinetalen	Programma-pakketten Symbolische talen	Besturings-systemen Hogere programmeer-talen	Databasemanagement-systemen Vierde-generatietalen Pakketten voor de microcomputer	Natuurlijke talen Veelzijdige ondersteunende pakketten met grafische interface Expertondersteunende pakketten
Ontwikkeling: richting gebruikersvriendelijke veelzijdige applicatiepakketten voor samenwerking en productiviteitsverhoging					

Figuur 2.3 Softwaregeneraties

1115 Compilers en interpreters - **---**

blz 21 2.1.4.1 **compiler -** COMPILERS zijn programma's die de geschreven opdrachten in een bepaalde programmeertaal (de broncode) omzetten naar de objectcode, dat is een "opsomming van alle instructies" in machinetaal. Nadat de vertaling cq compilatiegang heeft plaatsgevonden, heeft men een zgn objectprogramma en zolang dit niet hoeft te worden aangepast hoeft het niet iedere keer gecompileerd te worden en kan dit herhaaldelijk gebruikt worden.

379



Figuur 2.5 Werking compiler

vrg:	trefwoord	trefwrd onderverdeling	omschrijving	
1116	**Compilers en interpreters -**			**---**
blz 21 2.1.4.2	**interpreters -**		zijn vertolkers. Dit zijn computerprogramma's die opdrachten in de hogere programmeertaal interpreteren. Dwz het programma dat de gewenste werking teweeg moet brengen, komt gelijktijdig met de vertolker in het geheugen te staan (interpretatieprogramma), daarna worden de opdrachten in de hogere programmeertaal één voor één uitgeveord. Dus hier wordt de vertaling naar machinetaal iedere keer opnieuw gebruikt als het programma wordt gebruikt/aangesproken.	
1117	**Voorbeelden van programma's -**			**---**
blz 22 2.2.1	**machinetaal (1e generatie) -**		Een opdracht in machinetaal ziet er in de ATE3 computer als volgt uit > 00 00101 1001 11 110 0011110010101011	
1118	**Voorbeelden van programma's -**			

blz 22
2.2.2
	assembleertaal (2e generatie) -		Een opdracht in een assembleertaal (ATE3) ziet er bv als volgt uit >
			LDC R4, CIJFER
			
			dit is al beter te lezen als de machinetaal. Voor alle stappen moet een instructie geschreven worden, dan kan er als voglt uitzien >
			
			LDW R3, DEELGETAL
			LUS: SPR LT, EIND
			MINW R3, DELER
			SPR AL, LUS
			EIND: SOM R3, DELER
			OPSW R3, REST	

1119	**Voorbeelden van programma's -**			**---**

blz 23
2.2.3
	hogere talen (oa basic, pascal ed) -		een klein programma voor het omrekenen van Farenheit naar Celsius temperaturen, kan er als volgt uitzien >
			
			program FarenheitCelsius(input,output);
			var i : integer;
			procedure ToCelsius(fahr: integer);
			var celcius: interger;
			begin
			celcius:= 5 * (fahr - 32)/9;
			wrtieln(fahr, ' ',celsius);
			end; {ToCelsius}
			
			begin {hoofdprogramma}
			for i := 50 to 100 do
			begin
			ToCelcius(i);
			end;
			end.	

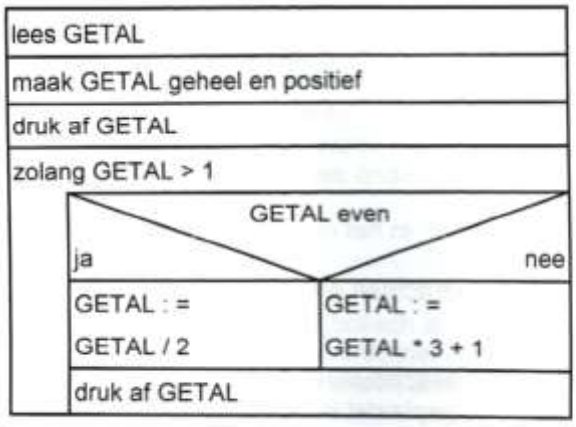
1120	**Voorbeelden van programma's -**			**---**
blz 23 2.2.4	**tools om het de programmeur makkelijk te maken (4e generatie) -**		bv SQL en de opdracht > SELECT n,a,w FROM persoon WHERE w="maastricht"	

vrg:	trefwoord	trefwrd onderverdeling	omschrijving
1121	**Voorbeelden van programma's -**		**---**
blz 23 2.2.5	**obeject oriented (5e generatie) -**	worden gekenmerkt door het feit dat objecten worden gedefinieerd met hun eigenschappen en met de activiteiten die door de objecten kunnen worden uitgevoerd. Kenmerken zijn > - eigenschappen - operations (methoden)	

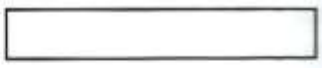
SE.1 Hoofdstuk: 3 Basisprincipes van programmeren

1122	**Basisprincipes -**		*******
blz 27 3.1	**PSD - Programma Structuur Diagram -**	De sequentie wordt uitgedrukt met een rechthoek, fig 3.2. De iteratie wordt uitgedrukt met een rechthoek met een liggende L erin en nog een rechthoek, zie fig 3.3. De selectie wordt uitgedrukt met de splitsing erin, zie figuur 3.4. Korte uitleg van figuur 3.1 > lees GETAL computer plaatst vanuit invoer een getal in zijn intern geheugen maak GETAL geheel en positief inhoud van GETAL naar register kopieren, daar wordt bepaald of het positief en geheel is. Als het niet positief is wordt het positief gemaakt. Daarna wordt het in de geheugenplaats van het symbolische adres GETAL geplaatst. De oorspronkelijke inhoud van GETAL is dan overschreven. druk af GETAL GETAL wordt naar uitvoerapparaat getransporteerd en afgedrukt op scherm of papier. Zolang GETAL > 1 blijf het volgende doen totdat het getal gelijk wordt of kleiner is dan 1. GETAL even controleer of het een even getal is. > ingeval van ja (het is een even getal) als het een even getal is doe dit: GETAL := GETAL/2 >> deel het GETAL door 2 en schrijf dit weg als GETAL > in geval van nee (het is een oneven getal) als het getal oneven doe dit: GETAL := GETAL * 3 + 1 >> vermenigvuldig het GETAL met 3 en tel hier 1 bij op en schrijf dit weer weg als GETAL druk af GETAL GETAL wordt naar uitvoerapparaat getransporteerd en afgedrukt op scherm of papier. Ga nu weer terua naar "Zolang GETAL > 1" je blijft in deze lus totdat het getal kleiner of gelijk is geworden aan 1.	

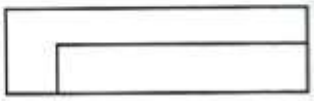
380



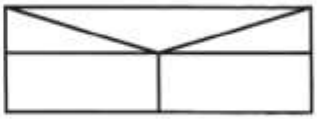
Figuur 3.1 Een ProgrammaStructuurDiagram (PSD)



Figuur 3.2 Het symbool voor sequentie in een PSD



Figuur 3.3 Het symbool voor iteratie in een PSD

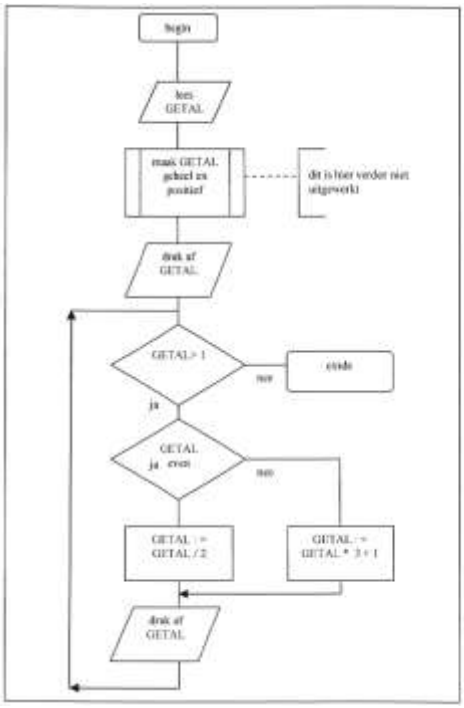


Figuur 3.4 Het symbool voor selectie in een PSD

vrg:	trefwoord	trefwrd onderverdeling	omschrijving
1123	**Basisprincipes -**		

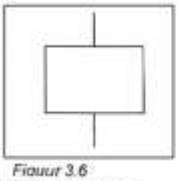
blz 30 3.2 **PSS = Programma Stroom Schema -** zie vorigaande PSD, dit PSS doet hetzelfde maar hier wordt het anders weergegeven.

398

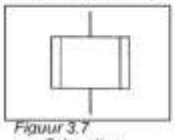


Figuur 3.5
Een ProgrammaStroomSchema (PSS)

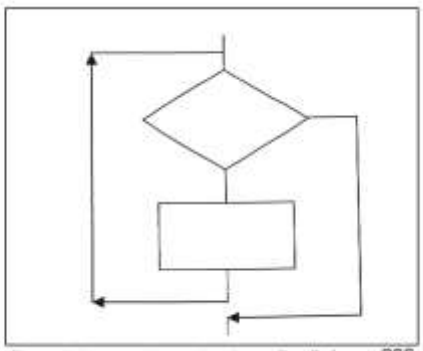
399



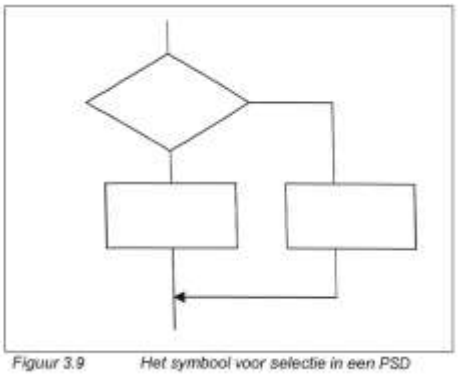
Figuur 3.6
Het symbool voor sequentie in een PSS



Figuur 3.7
Subroutine



Figuur 3.8 *Het symbool voor iteratie in een PSS*



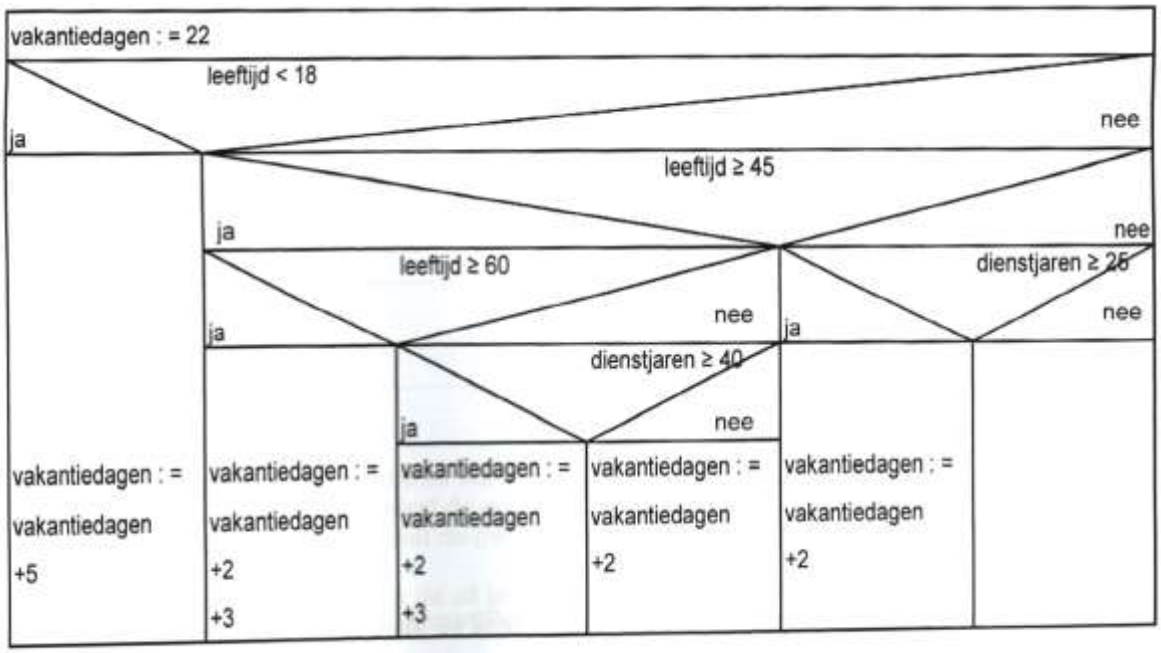
Figuur 3.9 *Het symbool voor selectie in een PSD*

vrg:	trefwoord	trefwrd onderverdeling	omschrijving
1124	**Basisprincipes -**		

blz 34 3.3 **beslissingstabel -** is een niet-procedurele procesbeschrijvingen en opgebouwd uit vier kwadranten (zie de dubbele lijnen).
De beslissingstabel is opgebouwd uit 4 kwadranten >
1. linksboven > bevat de condities;
2. rechtsboven > bevat de relevante uitkomstcombinaties voor de condities uit kwadrant 1;
3. linksonder > bevat de uit te voeren acties;
4. rechtsonder > bevat de acties die bij de conditie-uitslagen worden uitgevoerd.

In figuur 3.11 staat het uitgewerkte PSD van deze beslissingstabel.
Aan de ontvanger/lezer vragen in welke vorm hij de gegevens graag krijgt, PSS of PSD !!

401



Figuur 3.11 PSD van beslissingstabel van figuur 3.9

400

jonger dan 18 jaar	J	N	N	N	N	N
45 jaar of ouder (1)	-	J	J	J	N	N
25 dienstjaren of meer	-	-	-	(2) -	J	N
60 jaar of ouder	-	J	N	N	-	-
40 dienstjaren of meer	-	-	J	N	-	-
22 vakantiedagen	X	X	X	X	X	X
5 dagen extra (3)	X	-	-	(4) -	-	-
2 dagen extra	-	X	X	X	X	-
3 dagen extra	-	X	X	-	-	-

Figuur 3.10 Een beslissingstabel

SE.1 **Hoofdstuk: 4 Assembleertaal ATE3**

vrg:	trefwoord	trefwrd onderverdeling	omschrijving
1125	**Tabel Structuur -**		**---**
blz 42 4.1	**op een ATE3 computer -**		Een opdracht bestaat hier uit 32 bits (4 bytes = 2 woorden) en is als volgt opgebouwd > - de eerste 2 bits worden niet gebruikt; - de volgende 5 bits worden gebruikt voor de opdrachtcode; - de volgende 4 bits worden gebruikt voor de eerste operand; - de volgende 2 bits worden gebruikt voor de aanduiding C of W; - de volgende 3 bits worden gebruikt voor de aanduiding ve indexregister; - de volgende 16 bits worden gebruikt voor de tweede oprerand En ziet er dan bijvoorbeeld zo uit > SOM R1 W R2 SALDO hierin staat > SOM voor de opdrachtcode; register R1 voor de eerste operand; de aanduiding is W (W=inhoud en C=getal/fysieke cijfer); register R2 is het indexregister; het symbolische adres SALDO is de tweede operand. In ATE3 taal wordt dit: SOMW R1, SALDO(R2) en hier gebeurt dan > tel (SOM) inhoud vh register R1 op bij de inhoud van W deze W bevindt zich in een symbolische adres dat men SALDO in register 2, op de plek van R2 staat hoeveel stappen er verschoven moet worden binnen de tabel om dit SALDO te vinden. W = inhoud C = cijfer of de fysieke waarde
1126	**Gegevens verplaatsen -**		
blz 43 4.2	**drie richtingen -**		1. van register naar geheugen 2. van geheugen naar register 3. van register naar register
1127	**Gegevens verplaatsen -**		*******
blz 43 4.2.1	**van register naar geheugen -**		ATE3 commando daarvoor is > OPSW R4, A >> gaat van R4 -> A hiermee bedoelen we dat de inhoud van de waarde van R4 ook in het symbolische adres A wordt bewaard. De inhoud blijft ook in R4 staan. Het is dus meer kopiëren dan tranporteren.
1128	**Gegevens verplaatsen -**		*******
blz 43 4.2.2	**van geheugenplaats naar register -**		ATE3 commando daarvoor is > LDW R4, A >> gaat van R4 <- A (dus van achter naar voor) !!! Inhoud van A wordt in R4 gezet; Of LDC R5, 27 >> gaat van R5 <- 27 (dus van achter naar voor) !!! 27 wordt in R5 gezet W = inhoud C = cijfer of de fysieke waarde
1129	**Gegevens verplaatsen -**		*******
blz 44 4.2.3	**verplaatsen van register naar register -**		ATE3 commando daarvoor is > VRP R4, R2 >> hier gaat de wordt de inhoud van R2 ook in R4 gezet. Maw laden vanuit een ander rekenregister. Je kunt het ook copieren noemen.

vrg:	trefwoord	trefwrd onderverdeling	omschrijving

1130 Gegevens verplaatsen - ***

blz 44 4.2.4 **declaratie -**

DECLAREREN > om de naam en inhoud ve geheugenplaats te kunnen gebruiken moet een geheugenplaats zijn benoemd en er ook een waarde aan zijn toegekend. Dit ziet er in ATE3 als volgt uit voor een >

- declaratie ve naam voor een geheugenplaats :

  SALDO : DG 2 > naam: DG aantal bytes

  DG is declareer geheugenplaats
- declaratie van een waarde voor een gehuegenplaats

  SALDO: DC 18 > hiermee wordt bedoeld dat in geheugenplaats met symbolische adres SALDO de waarde/getal 18 wordt geplaatst.

  DC is hier declareer fysieke waarde/getal/cijfer

1131 Rekenen met gegevens - ***

blz 45 4.3 **rekenkundige opdrachten -**

De rekenkundige opdrachten zijn >

- SOMx > optellen
- MINx > aftrekken
- VRMx > vermenigvuldigen
- DELx > delen

waabij x staat voor een W van inhoud van een variabele of de C van fysieke (natuurlijk) getal.

408

	ATE3 opdracht	doet dus eigenlijk	resultaat wordt bewaard in	opmerking
optellen	SOMW R3 , B	= R3 + B	R3	B veranderd niet
optellen	SOMC R6 , 27	= R6 + 27	R6	
aftrekken	MINW R3 , B	= R3 - B	R3	B veranderd niet
aftrekken	MINC R6 , 27	= R3 - 27	R6	
vermenigvuldigen	VRMW R3 , B	= R3 * B	R2R3	Hierbij worden 2 registers gebruikt, In R2 komt te staan of het een positief (0000) of negatief (1000) getal is en in R3 de uitkomst. B veranderd niet
vermenigvuldigen	VRMC R6 , 27	= R6 * 27	R5R6	Hierbij worden 2 registers gebruikt, In R5 komt te staan of het een positief (0000) of negatief (1000) getal is en in R6 de uitkomst.
delen	DELW R3 , B	= R3 / B	in R3 en R2	Hierbij worden 2 registers gebruikt, In R2 komt te staan wat over is en het gehele getal in R3. Dus bij bv 25/4 > R3 wordt dan 6 en R2 wordt dan 1. B veranderd niet
delen	DELW R6 , 3	= R6 / 3	in R6 en R5	Hierbij worden 2 registers gebruikt, In R5 komt te staan wat over is en het gehele getal in R6. Dus bij bv 26/3 > R6 wordt dan 8 en R2 wordt dan 2.

1132 Vergelijken v gegevens - ***

blz 48 4.4 **vergelijkingen -**

vergelijkings opdrachten > kunnen alleen in combinatie met een "SPR" commando***, in deze sprongopdracht zien we pas de conditie terug wat er moet gebeuren als een vergelijking wel of niet klopt >

VRGx > vergelijken

waarbij x staat voor een W van inhoud of de C van fysieke getal.

Dus het kan zijn:

VRGW R3 , B > vergelijk de inhoud van register R3 met de inhoud van B, er veranderd niets;

of

VRGC R6 , 32 > vergelijk de inhoud van register R6 met 32, er veranderd niets.

vrg:	trefwoord	trefwrd onderverdeling	omschrijving

1133 Sprongen binnen een programma - ***

blz 49 4.5.1 **opbouw ve programma -**

Elk programma bestaat uit een verzameling opdrachten, door deze uit te voeren wordt een gewenst resultaat bereikt. De opdrachten worden na elkaar uitgevoerd. Soms wordt na het voldoen aan een bepaalde voorwaarde of conditie, verder gesprongen naar een ander deel in het programma, of wordt er juist niet gesprongen. De plaats waar men naar toe springt wordt aan gegeven met een "Label", dit is een symbolisch adres.
Een SPR-opdracht kan alleen in combinatie met een VRG-opdracht die ervoor staat!***
Een label kan voor elke regel in het programma staan.

Bijvoorbeeld >

```
        LDW   R4 ,  GETAL   [laad de inhoud van GETAL in register R4]
        VRGC  R4 , 10       [vergelijk de inhoud van R4 met het getal 10]
        SPR     GD , TIENA  [CONDITIE > als de inhoud van R4 groter dan 10 was, dan spring naar
TIENA]
         ……
         ……..
TIENA:  ……hier gaat je programma dan verder……
```

1134 Sprongen binnen een programma -

blz 49 4.5.2 **opslag van een programma -**

STORED PROGRAM PRINCIPE > Een programma wordt door het besturingssyteem in het interne geheugen geplaatst. De gegevens die voor de uitvoering vh programma nodig zijn, worden eveneens in het interne geheugen geplaatst. Daarna begint het besturingssyteem pas met het opdracht voor opdracht uitvoeren vh programma.
Is bedacht door de wiskundige Von Neumann.

1135 Sprongen binnen een programma -

blz 50 4.5.3 **uitvoering ve programma -**

Het adres waarop de eerste opdracht ve programma staat wordt vastgelegd in de instructiewijzer (=IW). De IW bevat tijdens de uitvoering vh programma steeds hetzelfde adres van de eerstvolgende opdracht die moet worden uitgevoerd. In een programma zonder sprongopdrachten zal de inhoud vd IW dus steeds met 4 worden verhoogd.

vrg:	trefwoord	trefwrd onderverdeling	omschrijving

1136 Sprongen binnen een programma - ***

blz 50 4.5.4 ***** voorbeeld *** -**

uitslag van een examen wordt met de volgende formule bepaald >
uitslag = ((9*score) - 70)/20
als de uitslag minimaal gelijk is aan de norm, dan is de kandidaat geslaagd.
als de uitslag minder is dan de norm, dan is de kandidaat gezakt.
Bij deze formule is een kandidaat pas geslaagd als hij/zij 22 vragen goed heeft.

Het programma in ATE3-taal wordt dan >

```
NORM:      DC      6            > declareer het gegeven NORM mer de waarde 6;
SCORE:     DG      2            > declareer een naam voor een geheugenplaats van 2 bytes;
UITSLAG:   DG      2            > declareer een naam voor een geheugenplaats van 2 bytes;
           LDW     R4 , SCORE   > laad de waarde v SCORE in register R4;
           VRMC    R4 , 9       > vermenigvuldig de inhoud van R4 met 9 en sla op in R3R4;
                                    de uitkomst word opgeslagen in R4, positief of negatief R3;
           MINC    R4 , 70      > trek 70 van de inhoud van register R4 af en sla op in R4;
           DELC    R4 , 20      > deel de inhoud van R4 door 20 en sla op in R3R4 >
                                    de hele getallen worden opgeslagen in R4, de rest in R3;
           VERGW   R4 , NORM    > vergelijk de inhoud van R4 met de inhoud van de NORM;
           SPR     KD , GEZAKT  > als de inhoud van R4 kleiner is dan de norm, spring door
                                    naar Label GEZAKT;
GESLAAGD:  ……..
           ……
           SPR     AL , EINDE   > EINDE is hier ook een Label en geen programma opdracht;
GEZAKT:    ……
           ……
EINDE:     ……
           STP                  > stoppen met de uitvoering van dit programma;
           EIND                 > einde van dit programma, LET OP is wat anders dan het
                                    de hele getallen worden opgeslagen in R4, de rest in Label
                                    dat einde heet.
```

1137 Sprongen binnen een programma - ***

blz 52 4.5.5 **soorten condities -**

- condities bij/na vergelijkingen en sprongen >

GD	>	is groter dan
KD	<	is kleiner dan
GA	=	is gelijk aan
GG	>=	is groter dan of gelijk aan
KG	<=	is kleiner dan of gelijk aan
NG	<>	is niet gelijk aan
AL		altijd

1138 Sprongen binnen een programma - ***

blz 52 4.5.5 **commentaar plaatsen -**

Wanneer een programmaregel begint met het karakter [, wordt de hele of de rest van de regel als opmerking opgevat en wordt de regel, als programmaregel genegeerd. Er wordt wel nog gecontroleerd op een afsluitende]. Dit kan bij ATE3 dmv de tekst zetten tussen vierkante haken []. Bijvoorbeeld >

```
  LDW  R4 , SCORE    [de inhoud van de SCORE naar R4]
of
  LDW  R4, SCORE     [hier mag gewone tekst staan]
  [maar hier mag dat ook]
  VRG  R4, SCORE
```

Lege programmaregels worden overgeslagen.

vrg:	trefwoord	trefwrd onderverdeling	omschrijving	
1148	**Soorten opdrachten -**			*******

blz 11 1.4 **opdrachten in ATE3 taal zijn -**

waabij x overal staat voor een W van inhoud of de C van fysieke getal.

- transportopdrachten >
  - LDx > laden <-
  - OPSx > opslaan ->
  - VRP > verplaatsen <- het is eigenlijk copieren
- rekenkundige opdrachten >
  - SOMx > optellen <-
  - MINx > aftrekken <-
  - VRMx > vermenigvuldigen <-
  - DELx > delen <-
- vergelijkings opdrachten > kunnen allen in combinatie met een "SPR" commando >
  - VRGx > vergelijken
- besturings opdrachten (GOTO) >
  - SPR > springen (naar een label)
  - SSR > springen naar subroutine
  - TSR > terugkomen ve subroutine (er moet een terugspring label bij zijn)
  - STP > stoppen met uitvoering ve programma / stoppen met de bewerking waar je mee bezig bent / betekent stop de uitvoering van dit STE3 programma.
- logische opdrachten >
  - EN > de logische en
  - OF > de logische of
  - UOF > de logische uitsluitende of (bv je bent man of je bent vrouw)
- speciale opdrachten >
  - DAB=DG > declareren ve aantal bytes, gebruiken bij declareren v tabellen of lege variabelen
  - DG=DAB > declareren van (een naam voor) een geheugenplaats (enkelvoudig)
  - DC > declareren van een waarde van een gegeven (de C staat hier voor cijfer)
  - EIND > de laatste regel vh programma !!!
- condities >
  - GD > is groter dan
  - KD < is kleiner dan
  - GA = is gelijk aan
  - GG >= is groter dan of gelijk aan
  - KG <= is kleiner dan of gelijk aan
  - NG <> is niet gelijk aan
  - AL is altijd doen
- opmerkingen >
  - [] staan in de regel altijd tussen vierkante haken

409

Syntax:

[LABELx:]	{LDW|LDC|OPSW|OPSC|MINW|MINC|VRMW|VRMC|DELW|DELC} Rx , [LABELy|z]		[[tekst]]
[LABELx:]	{VRGW|VRGC}	Rx , [LABELy|z|Ry]	[[tekst]]
[LABELx:]	VRP Rx , Ry		[[tekst]]
[LABELx:]	{SPR|SSR}	{GD|GA|GG|KD|KG|NG|AL} , LABELy	[[tekst]]
[LABELx:]	{TSR|STP|EIND}		[[tekst]]
LABELx:	{DAB|DC|DG}	z	[[tekst]]

Tot aan de definitieve versie van november 2009 bestaat er onduidelijk over het gebruik van de opdrachten: EN / OF / UOF / SHL / SHR. Later blijkt dat deze geen onderdeel zijn van de SE.1 module en daarmee niet van deze editor en haar documentatie. Zij worden wel genoemd, maar verder niet verklaard.

vrg:	trefwoord	trefwrd onderverdeling	omschrijving
1149	**Soorten opdrachten -**		
blz 11 9	**opmerkingen -**		Variabele of Labelnaamcontroles > 1. Hoofdletters gebruiken 2. Wanneer een lettergroep in de programmaregel voorkomt met een ":", dan wordt deze aangemerkt als een Variabele-of Labelnaam. Als deze Variabele-of Labelnaam niet aan het begin van de programma regel staat, volgt een foutmelding. Een ":" is uiteraard wel toegestaan in een opmerking, ver-derop in de programmaregel. Registerbewerking > 1. Onderkastnamen worden zonder verdere foutmelding omgezet in kapitalen; 2. Alle dubbele spaties worden uit de opdrachtregel verwijderd om spatiecontrole mogelijk te maken; 3. Voor en na de komma worden eventuele spaties verwijderd; 4. De registeraanduiding voor de komma kan alleen maar R1, R2, R3, R4, R5, R6 of R7 zijn. In alle andere gevallen volgt een foutmelding; 5. Alleen bij een VRGW en een VRP opdracht mag achter de komma een tweede register vermeld worden. In alle ander gevallen volgt een foutmelding. Het tweede register mag gelijk zijn aan het eerste; 6. Als het eerste karakter achter de komma een letter is, wordt er vanuit gegaan dat dit een geldige Symbolische naam is. Daarna wordt gecontroleerd of het betrekking heeft op een opdracht dat op een W eindigt (LDW, OPSW, SOMW, MINW, VRMW, DELW, VRGW); 7. Als achter de komma een getalswaarde staat, wordt gecontroleerd of het betrekking heeft op een opdracht die op een C eindigt (LDC, OPSC, SOMC, MINC, VERMC, DELC, VRGC). Commando > 1. Het commando moet een van de volgende commando's zijn: LDW, LDC, OPSW, OPSC, VRP, SOMW, SOMC, MINW, MINC, VRMW, VERMC, DELW, DELC, VRGW, VRGC, SPR, SSR, TSR, STP, EN4, OF4, UOF4, SHL4, SHR4, DAB, DC, DG, EIND; 2. Als het commando DAB, DCof DG is, moet het gevolgd worden door een getalswaarde; 3. Niet geïmplementeerd, is het controleren van de semantiek van een programmaregel. Ook de semantiek van het programmabestand in zijn geheel behoort niet tot de functionele wensen.

SE.1	Hoofdstuk: 5 Tabellen

1139	**Tabellen -**		**---**
blz 57 5.1	**ééndimensionale tabellen -**		Een tabel een een verzameling van gelijksoortige gegevens. Een eendimensionale tabel is een verzameling gelijksoortige elementen. De declaratie van een tabel gebeurt door het declareren van een aantal bytes, als volgt > TAB: DAB 400 men reserveert hier een geheugenruimte van 400 bytes > dus 200 woorden; Het eerst woord heeft als beginadres TAB of TAB+0; Het laatste woord heeft als adres TAB+398. Het aanwijzen ve element vd tabel gebeurt mbv een indexregister. Elk rekenregister kan ook als indexregister gebruikt worden. Als register R4 bv een waarde van 40 heeft, dan wordt met TAB (R4) het element aangewezen dat zich op adres TAB+40 bevindt.
1140	**Tabellen -**		
blz 60 5.2	**meerdimensionale tabellen -**		Een meerdimensionale tabel onstaat als we in een eendimensionale tabel meer dan een soort gegevens willen opslaan en bestaat uit rijen en kolommen.

SE.1	Hoofdstuk: 6 Zoeken in tabellen

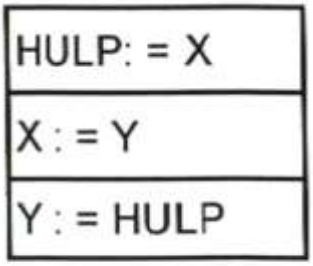
vrg:	trefwoord	trefwrd onderverdeling	omschrijving
1141	**Sorteren ve tabel -**		

blz 64 6.1.1 **verwisselen van tabelelementen -**

Een tabel heeft gesorteerd als de elementen in een oplopende (ascending) of dalende (descending) volgorde zijn gerangschikt.

Om te kunnen sorteren kan het zijn dat er 2 elementen omgewisseld moeten worden.

402



Figuur 6.1 De routine "verwissel"

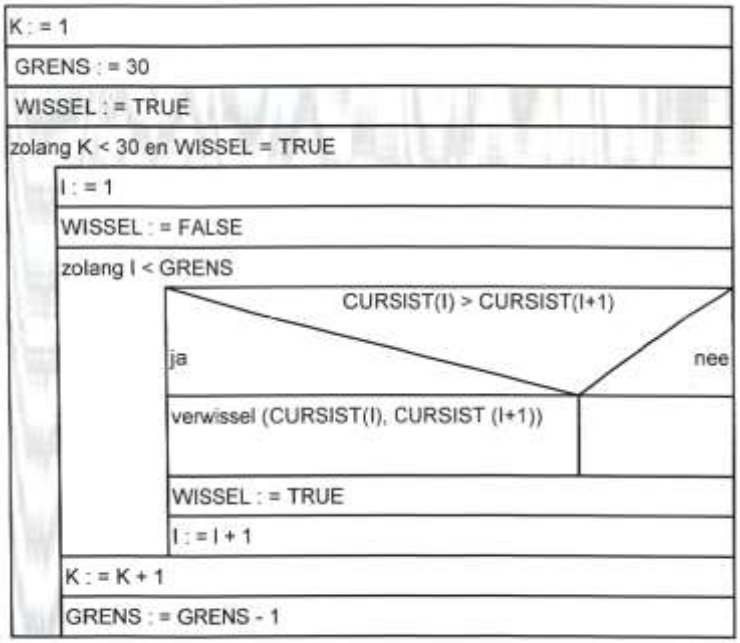
1142	**Sorteren ve tabel -**

blz 65 6.1.2 **bubble-sort -**

is gebasseerd op het volgende principe >
- doorloop de tabel en zorg ervoor dat het grootste element achteraan komt te staan;
- doorloop de tabel en zorg ervoor dat het een na grootste element voor het grootste element komt te staan;
- doorloop de tabel en zorg ervoor dat het twee na grootste element voor het een na grootste element komt te staan;
- doorloop de tabel en zorg ervoor dat het drie na grootste element voor het twee na grootste element komt te staan;
- en zo verder tot je alles hebt gehad….. Zie figuus 6.2

Deze wordt van achter naar voor opgebouwd.

403



Figuur 6.2 De sorteermethode bubble-sort

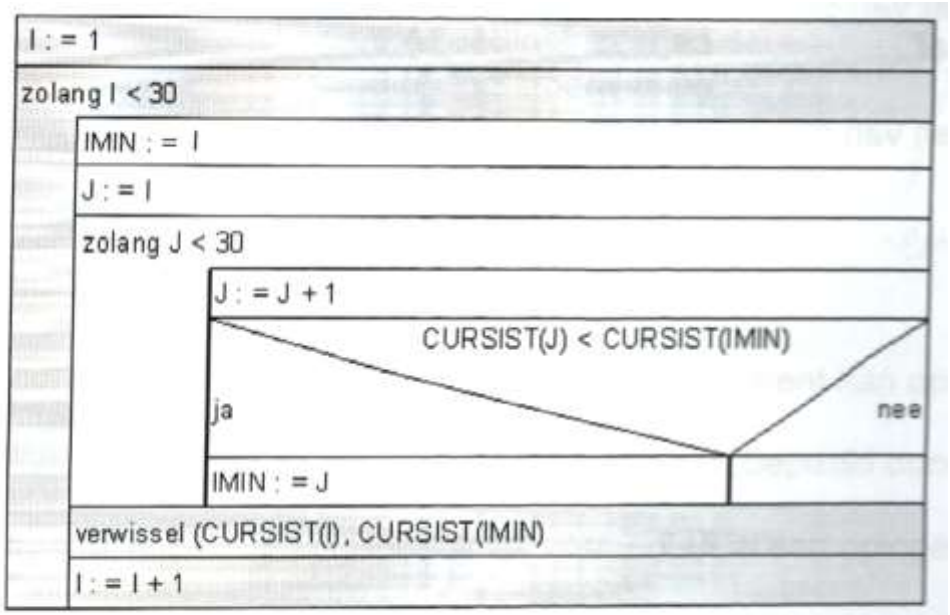
vrg:	trefwoord	trefwrd onderverdeling	omschrijving
1143	**Sorteren ve tabel -**		

blz 67 6.1.3 **sort-by-straight-selection -**

hier bouwen we de sortering van voren naar achteren op. Zie figuur 6,3.
Daar zijn we uitgegaan van 30 elementen. Model kan met kleine aanpassingen ook gebruikt worden bij een sorteervolgorde van klein naar groot.

Halveren > kijken of het erin zit > met dat deel verder > weer halveren > kijken > verder……en zo door / uiteraard geldt dat als het er niet in zit je dan met het andere deel verder gaan.

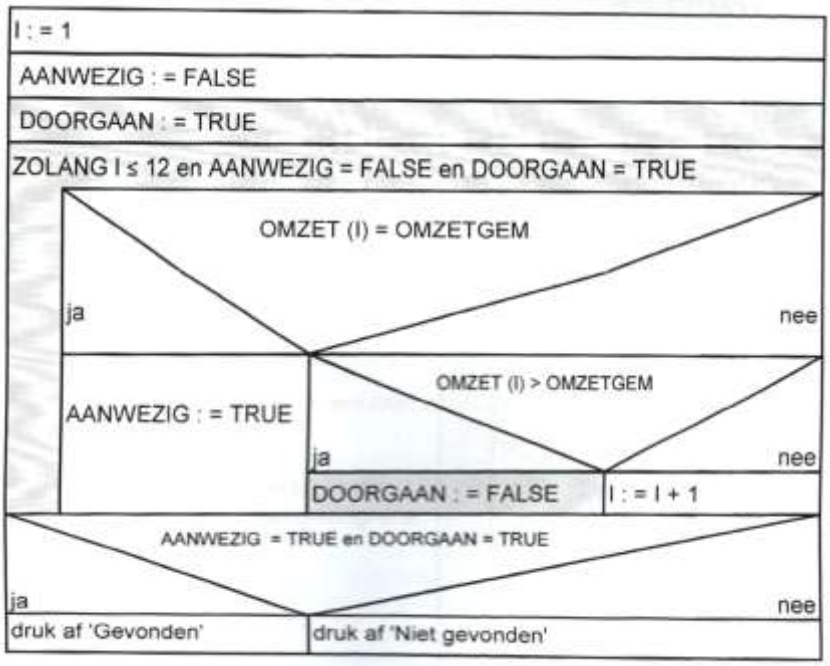
404



Figuur 6.3 De sorteermethode sort-by-straight-selection

vrg:	trefwoord	trefwrd onderverdeling	omschrijving
1144	**Sorteren ve tabel -**		
blz 67 6.2	**liniair zoeken -**		wil zeggen dat de tabel van voor tot achter wordt doorlopen om te bepalen of een vastgestelde waarde in de tabel voorkomt. In een ongesorteerde tabel moet de hele tabel doorlopen worden, in een gesorteerde tabel hoeft dat niet omdat > - als het element gevonden is, hoeft er niet meer doorgezocht te worden; - als er geconstateerd kan worden dat het element er niet in zit, hoeft de rest ook niet meer nagekeken te worden. Zie figuur 6.4 > - er zijn 12 elementen met een maandomzet als inhoud; - het gemiddelde v deze maandomzetten is berekend en vastgelegd in OMZETGEM; - er is een omzet die gelijk is aan de berekende OMZETGEM.

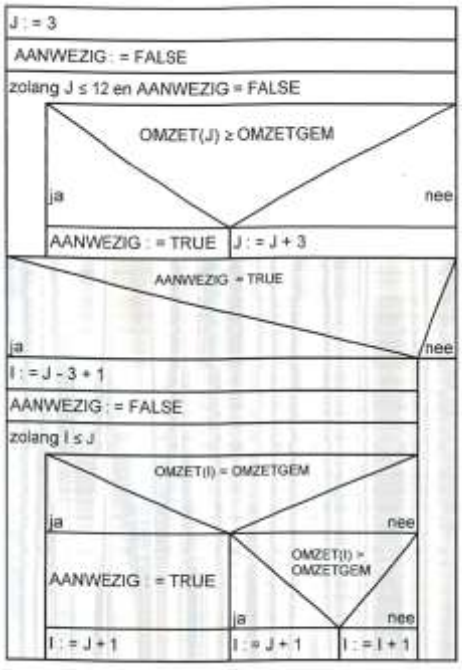
405



Figuur 6.4 *Lineair zoeken in een gesorteerde tabel met 12 elementen*

1145	**Sorteren ve tabel -**		
blz 68 6.3	**skip-sequentieel zoeken -**		hierbij wordt de tabel gesplitst in even grote delen. Vervolgens wordt bepaald in welk deel de gevraagde waarde moet liggen en als we dat weten wordt daarin liniair (sequential) gezocht. Zie fig. 6,5 > tabel v 12 elementen en wordt opgedeeld in 3 elementen.

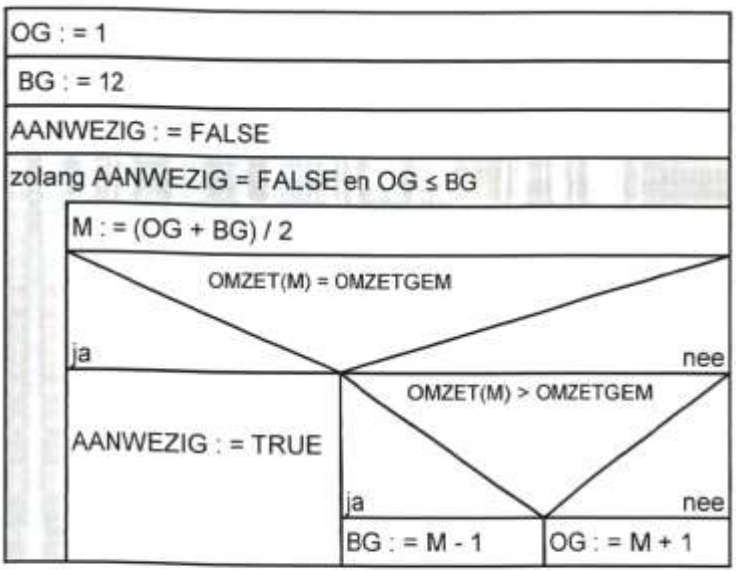
406



Figuur 6.5 *Skip-sequential zoeken in een tabel van 12 elementen*

vrg:	trefwoord	trefwrd onderverdeling	omschrijving
1146	**Sorteren ve tabel -**		
blz 69 6.4	**biniair zoeken -**		Dit gebeurt als volgt > tabel wordt in 2 delen verdeeld > bepalen in welk deel de gezochte waarde zit > dat deel ook weer in twee delen > en weer kijken in welk van die twee delen de waarde zit > dan weer in twee delen opsplitsen……en zo verder tot het element gevonden is. Zie fig. 6.6, hier is OG = ondergrens en BG = benendengrens

407



Figuur 6.6 *Binair zoeken in een tabel van 12 elementen*

OG staat voor: ondergrens
BG staat voor: benedengrens

SE.1	Hoofdstuk: 9	Overige info
1147	**divers -**	***
blz 99 9.9.9	**divers -**	Indien er gevraagd wordt welke zoekmethode het minst aantal vergelijkingen met betreffende waarde wordt uitgevoerd kun je concluderen dat > dit afhankelijk is van de gezochte waarde !!