RA

STICHTING
MATHEMATISCH CENTRUM
2e BOERHAAVESTRAAT 49
AMSTERDAM
REKENAFDELING

MR 81

Het MC-ALGOL 60-systeem voor de X8

Voorlopige programmeurshandleiding

door

F.E.J.Kruseman Aretz

RA

5e uitgave

1970

L.S.

Dit rapport is een poging, een zo volledig mogelijke handleiding te geven voor het vervaardigen van programma's en getallenbanden voor het ALGOL-systeem voor de X8 in zijn huidige vorm. Het is een zelfstandig geheel; er is geen beroep in gedaan op (voor-)kennis van de ALGOL-implementatie op de X1.

Hopelijk zal de programmeur, die wil spelen met alle mogelijkheden tot aan de grenzen die het ALGOL-systeem voor de X8 hem stelt, met dit rapport over een betrouwbaar naslagwerk beschikken.

Hopelijk zal ook de "occasional" programmeur uit deze leidraad de informatie kunnen putten die hij nodig heeft. Voor hem zijn de secties 3, 4, 7 en 8 de belangrijkste.

Het MC-ALGOL 60-systeem voor de X8 is geheel op het Mathematisch Centrum ontwikkeld. Een grote groep medewerkers van de Rekenafdeling heeft er aan bijgedragen. Het meeste werk is echter verzet door F.J.M.Barning, B.J.Mailloux, J.J.B.M.Nederkoorn en de auteur van deze handleiding. Het is de bedoeling, dat geleidelijk aan dit ALGOL-systeem tot in details beschreven zal worden.

FKA

Voorlopige programmeurshandleiding

1. Eisen te stellen aan ALGOL 60-programma's

Het ALGOL-systeem voor de X8 accepteert programma's, geschreven in ALGOL 60 als gedefinieerd in het Revised Report on the Algorithmic Language ALGOL 60, onder de voorlopige beperking, dat hierin
1.1. geen own <type> array's in gedeclareerd worden,
1.2. geen integer labels met een integer-waarde > 67108863 voorkomen,
1.3. het aantal gedeclareerde locale variabelen en locale labels binnen een procedure body zekere, zeer ruime grenzen niet overschrijdt,
1.4. het aantal entries in een switch declaration zekere, zeer ruime grenzen niet overschrijdt.

Een aantal (functie-) procedures mag, zonder dat zij in het programma gedeclareerd behoeven te worden, gebruikt worden. Naast de in 3.2.4. van het Revised Report opgenomen elementaire functies behoren hier onder meer alle in- en uitvoer-procedures toe. Voor een volledige opsomming zie 7.

Uitsluitend in commentaar (na comment en end) en binnen strings mogen, naast de basic symbols van ALGOL, ook de symbolen accent, apostrophe en vraagteken (', " en ?), alsmede willekeurige onderstreepte of doorbalkte combinaties voorkomen.

Formele parameters van procedures behoeven, voor zover ze niet in de value part voorkomen, niet gespecificeerd te worden. Het is gewoonlijk voordelig, zo veel mogelijk parameters te specificeren, daar de extra informatie, verschaft door de specificatie, mede in de test op overeenkomst formele/actuele parameters wordt betrokken, en bovendien de efficientie van de uitvoering van het programma ten goede kan komen. Het is evenwel mogelijk, procedures te schrijven die aan de afwezigheid van een of meer specificaties hun speciale betekenis ontlenen.

2. Enige interpretaties van ALGOL 60

2.1. De specificaties real en integer hebben de volgende betekenis:
2.1.1. als de bijbehorende formele parameter in de value part voorkomt, wordt in de procedure body een locale variabele gedeclareerd van het type volgens de specificatie, en aan die variabele wordt de waarde van de corresponderende actuele expressie geassigneerd, zonodig daarbij een real resultaat afrondend tot een integer waarde,
2.1.2. als de formele parameter niet in de value part voorkomt, worden beide specificaties geinterpreteerd als een mededeling "arithmetisch" en in hoofdzaak gebruikt voor de syntactische controle. Bij iedere aanroep van de procedure wordt overal in de body de formele parameter vervangen door de corresponderende actuele (zie Revised Report 4.7., in het bijzonder 4.7.3.2.), en het type van de actuele parameter dringt dus door tot in de body, zonder dat enige transferfunctie wordt ingelast, ongeacht de specificatie real of integer.

2.2. De specificatie array, zonder type-vermelding, wordt, als de bijbehorende formele parameter in de value part voorkomt, geidentificeerd met de specificatie real array. Anders wordt de specificatie array be-

schouwd als een mededeling, dat de formele identifier de identifier van een array voorstelt.

2.3. Bij formele parameters die in de value part voorkomen, worden de specificaties real procedure, integer procedure en Boolean procedure geidentificeerd met respectievelijk real, integer en Boolean.

2.4. Een formele parameter, voorkomend in de value part, mag gespecificeerd worden als label. De corresponderende actuele parameter moet dan een designational expression zijn; deze wordt bij het binnenkomen van de procedure, evenals alle andere parameters uit de value part, geevalueerd.

2.5. In tegenstelling tot 4.3.5. van het Revised Report wordt de evaluatie van een switch designator die niet tot een entry uit de switch list leidt, als een ongeoorloofde situatie beschouwd.

2.6. Bij de uitwerking van expressies worden de operanden geevalueerd in de volgorde van de ALGOL-tekst, van links naar rechts. Bij het binnenkomen van een procedure worden de formele parameters uit de value part geevalueerd in de volgorde van de formele-parameterlijst; echter worden eerst de simpele value parameters, en daarna de value arrays afgehandeld.

3. Ponsconventies voor ALGOL-programma's

De programmateksten dienen geponst te worden op 7-gats band in de z.g. MC-flexowriter-code. Enige conventies, waaraan de hand gehouden moet worden, zijn:

3.1. twee opeenvolgende word delimiters moeten gescheiden worden door tenminste een niet-onderstreept symbool. Dit laatste mag een spatie, tabulatie, of terug-wagen-nieuwe-regel zijn.

3.2. de laatste end van het programma moet afgesloten worden door ten minste een niet-onderstreept symbool (bij voorkeur een terug-wagen-nieuwe-regel).

3.3. het "wordt"-symbool (:=), voorkomend in assignment statements, moet geponst worden als een : gevolgd door een = niet gescheiden door enig ander symbool (ook geen spatie).

3.4. De symbolen $\dot{+}$, $\doteq$, $\supset$, $\neq$, $\leq$, $\geq$, 6 (quote) en 9 (unquote) worden in enige ponsingen geponst, en wel als:

$\dot{+}$	als	_	gevolgd door	:
$\doteq$	als	_	gevolgd door	=
$\supset$	als	_	gevolgd door	¬
$\neq$	als	|	gevolgd door	=
$\leq$	als	_	gevolgd door	<
$\geq$	als	_	gevolgd door	>
6	als	|	gevolgd door	<

9 als | gevolgd door >

3.5. Tape feed (blank), erase (127), stopcode (11) en backspace (42) worden overal in de programmaband genegeerd.

3.6. Het programma mag, zo nodig, over verschillende banden verdeeld zijn. Elke band moet beginnen en eindigen met ten minste 25 cm blank.

3.7. Eventueel getallenmateriaal, dat tijdens de uitvoering van programma's gelezen moet worden, mag (maar hoeft niet) direct achter het programma geponst worden op dezelfde band.

3.8. De bandlezers van de X8 zijn zeer kieskeurig met "plakjes" in banden.

4. De verwerking van ALGOL-programma's

4.1. Voor verwerking dienen ALGOL-programma's in de speciale enveloppen voor rekenopdrachten ingeleverd te worden aan de "balie". De enveloppe moet bevatten: de programma-band(en) en de getallen-band(en) met de bijbehorende teksten. De banden moeten duidelijke opschriften bevatten (ten minste "programma" of "getallen") en zo nodig genummerd zijn. Toegevoegd kunnen worden instructies voor de operateur als "na 10 seconden afbreken", "geen regelnummers" (zie 4.2.4.), "zonder protectie" (zie 6.2.).

4.2. De verwerking geschiedt hierna als volgt:

4.2.1. Het programma wordt ingelezen en in enige scans op syntactische fouten getoetst. Tijdens het inlezen wordt de tekst afgedrukt over de regeldrukker, waarbij iedere regel wordt voorafgegaan door het regelnummer, gevolgd door enige spaties. Zo nodig wordt het afdrukken van een regel onderbroken voor het afdrukken van een foutmelding voor een fout, reeds gedetecteerd tijdens de inleesfase. Voor syntactische fouten, gevonden in de latere scans, wordt een foutmelding afgedrukt volgend op de programmatekst. De representatie van de ALGOL-symbolen op de regeldrukker wordt behandeld in tabel III.

Iedere melding van een fout tegen de syntaxis heeft de volgende gedaante:

er <foutnummer> <regelnummer> <laatst gelezen symbool> <waarde laatst gelezen constante> <eerste 8 karakters van laatst verwerkte identifier>

De interpretatie van de foutnummers is te vinden in tabel I, de decodering van het laatst gelezen symbool geschiedt met behulp van tabel II. Het regelnummer slaat op de regel van de ALGOL-tekst, waarin de fout gedetecteerd is. Hoewel in de foutmelding slechts 8 karakters van de laatst verwerkte identifier gegeven worden, doen voor het ALGOL-systeem voor de X8 alle letters en cijfers van een identifier mee.

Sommige fouten in de ALGOL-tekst kunnen tot andere foutmeldingen leiden dan voor de hand zou liggen. Dit hangt samen met de wijze, waarop het syntactisch onderzoek van een tekst, die ten gevolge van een fout min of meer oninterpretabel is, verder wordt voortgezet.

Iedere pagina op de regeldrukker begint met een kopje, vermeldende de datum van verwerking en een serienummer, door het systeem aan

deze verwerking toegekend. Voor tekst en foutmeldingen staan 60 regels per pagina ter beschikking.

4.2.2. Slechts als bij bovengenoemd, syntactisch onderzoek geen fouten gevonden zijn, wordt het programma uitgevoerd. Output over de regeldrukker begint op een nieuwe pagina, output over de bandponsér wordt voorafgegaan door een standaardbegin en afgesloten door een standaardslot. Het standaardbegin heeft de vorm:

<stuk blank> <TWNR> '<datum> - <serienummer>' <TWNR> <stuk blank> <erase>

het standaardslot luidt:

<erase> <stuk blank> <stopcode> <stuk blank>

Bovendien wordt, iedere keer als de voorraad ongeponste band in de bandponser dreigt op te raken, het standaardslot geponst, op de nieuw ingelegde band gevolgd door het standaardbegin.

4.2.3. Bij het detecteren van een ongeoorloofde situatie in de executiefase van een programma wordt de uitvoering direct afgebroken; als laatste handelingen wordt een foutmelding gegeven over de regeldrukker en zonodig een standaardslot geponst. De normale vorm voor een foutmelding tijdens executie is:

er <foutnummer> <regelnummer> <waarde laatst ingevoerde decimale getal>

Voor de interpretatie van de foutnummers verwijzen we weer naar tabel I. Het regelnummer verwijst naar de regel van de ALGOL-tekst, waarin het laatst in executie genomen maar onvoltooid gebleven statement of de laatste in executie genomen maar nog niet voltooide array-declaratie begint.

4.2.4. Het is mogelijk, de operateur te vragen, het programma te doen uitvoeren zonder dat tijdens deze uitvoering het regelnummer wordt bijgehouden. Dit leidt als regel tot een verwaarloosbaar kortere executietijd, maar geeft enige winst in het beschikbare geheugen voor i.h.b. array-declaraties of recursieve aanroepen van procedures. In het bovengenoemde geval heeft een foutmelding tijdens executie de vorm:

er <foutnummer> <waarde laatst ingevoerde decimale getal>

4.2.5. De uitvoering van een programma wordt beeindigd:
a) door het "passeren" van de laatste end,
b) bij aanroep van de bibliotheek-procedure EXIT,
c) bij detectie van een fout,
d) door operateurs-ingreep.
In dit laatste geval wordt een pseudo-foutmelding gegeven met foutnummer 999.
In het algemeen zal de operateur een beeindigd programma niet herstarten (b.v. voor dezelfde berekeningen met een tweede getallenband). Indien men het programma enige malen doorlopen wil hebben, dient het programma zelf de vorm van een cyclus te hebben. Als dan b.v. de getallenbanden alle afgehandeld zijn en het programma om nieuwe invoer vraagt, zal de operateur het programma afbreken.

4.2.6. Er zijn in principe twee soorten "diensten":
a) twee-minuten-dienst,
b) dienst-volgens-afspraak, voor langer durende programma's.
In de twee-minuten-dienst worden programma's ten hoogste 2 minuten op de machine gelaten.
Alle programma's worden in closed-shop bedrijf door operateurs gedraaid.

5. De arithmetiek van het ALGOL-systeem

De arithmetiek van het ALGOL-systeem voor de X8 stemt overeen met de floating-point arithmetiek van de X8.

5.1. Variabelen van type integer bezetten in het geheugen een enkel woord. Als gevolg hiervan is het waardebereik van integer-variabelen beperkt van - 67 108 863 tot en met + 67 108 863, en mag, tijdens de uitvoering van het programma, geen waarde aan een integer-variabele toegekend worden, in absolute waarde groter (na eventuele afronding) dan 67 108 863. Dit geldt ook voor impliciete assignments aan integers (vergelijk Revised Report 3.1.4.2., 4.7.3.1., 5.2.4.1., en 5.4.4.).
Een uitzondering hierop vormt de integer procedure entier, waarvan het waardebereik niet in bovenstaande zin beperkt is.

5.2. Variabelen van type real bezetten in het geheugen twee woorden, en worden voorgesteld in floating-point representatie met een mantisse van 40 binaire posities (40 bits) en teken en een binaire exponent van 11 bits en teken. Hun waarde representeert in het algemeen een niet-geheel getal met een relatieve precisie van ongeveer 12 decimalen. Gehele getallen, in absolute waarde kleiner dan 1 099 511 627 776 kunnen exact door een real variable worden voorgesteld.

5.3. De uitwerking van expressies geschiedt steeds in floating-point arithmetiek. De arithmetische operaties +, -, ×, en / geven het best mogelijke, in bovengenoemde floating-point representatie voorstelbare resultaat, met behoud van de monotonie (dit impliceert b.v. dat als $a < b$ en $c \geq 0$, dan $a \times c \leq b \times c$).
Als de operanden bij de operaties +, -, en × gehele getallen zijn, in absolute waarde kleiner dan 1 099 511 627 776, en het resultaat van deze operaties ook in absolute waarde deze grens niet overschrijdt, is dit resultaat exact.
De arithmetische operatie /\ levert een resultaat met een relatieve nauwkeurigheid van ongeveer 12 decimalen. Als de exponent een geheel getal is, in absolute waarde kleiner dan 30, wordt het resultaat opgebouwd door herhaalde vermenigvuldiging.

5.4. De waarde van expressies en variabelen van type real ligt in absolute waarde tussen ongeveer $_{10}628$ en ongeveer $_{10}-616$, of is exact 0.
Als bij arithmetische operaties een resultaat zou ontstaan, dat in absolute waarde groter dan ongeveer $_{10}628$ zou zijn, wordt het grootst mogelijke, nog voorstelbare resultaat gevormd, ongeveer $_{10}628$ met het correcte teken.
Een resultaat 0 ontstaat bij additie of subtractie alleen, als beide operanden in absolute waarde "bit voor bit" gelijk zijn, bij vermenigvuldigen slechts, als ten minste een der beide operanden 0 is. In alle andere gevallen is het resultaat van een arithmetische operatie in absolute waarde tenminste ongeveer $_{10}-616$.
Bij deling door 0 ontstaat, als het deeltal $\neq 0$ is, het in absolute waarde grootst mogelijke resultaat. De waarde van 0/0 kan van geval tot geval verschillen.

5.5. De waarde van de relatie a = b is slechts true, als de twee operanden a en b "bit voor bit" gelijk zijn. De relaties a > b en a < b leveren zeker false, als de twee operanden a en b "bit voor bit" gelijk zijn.

6. Geheugenbezetting

6.1. Voor ALGOL-programma's en hun werkruimte (variabelen, arrays, en blokadministratie, bijgehouden door het ALGOL-systeem) staan voorlopig zeker 18 000 geheugenplaatsen ("woorden") ter beschikking.
Een variabele van type real beslaat in het geheugen twee woorden, variabelen van type integer en Boolean een woord.
real arrays beslaan twee woorden per element, integer arrays een woord per element, Boolean arrays een woord per 27 elementen.

6.2. Indien tijdens uitvoering van een programma de beschikbare geheugenruimte uitgeput is, wordt de uitvoering afgebroken met een foutmelding met foutnummer 609.
Het is mogelijk, in overleg met de contactpersoon aan de "balie", het programma nog eens aan te bieden, voorlopig met de instructie voor de operateur "zonder protectie". Tijdens de executiefase van een programma staan dan zeker 23 000 geheugenadressen ter beschikking.

7. Bibliotheek-procedures

De voorlopige bibliotheek van het ALGOL-systeem voor de X8 bevat de volgende groepen procedures, die niet in het programma gedeclareerd behoeven te worden:

7.1. Elementaire functies

7.1.1. real procedure abs (x); value x; real x;
abs:= if x ≥ 0 then x else - x;

7.1.2. integer procedure sign (x); value x; real x;
sign:= if x = 0 then 0 else if x > 0 then 1 else - 1;

7.1.3. real procedure sqrt (x); value x; real x;
Als de vierkantwortel van x exact representeerbaar is in de floating-point arithmetiek van de X8, wordt dit exacte resultaat afgeleverd. In het bijzonder geldt sqrt (i × i) = i voor i = 0 (1) 1 048 575. Voor x < 0 wordt voor sqrt (x) een resultaat 0 gegeven.

7.1.4. real procedure sin (x); value x; real x;
Er is voldaan aan de eisen: sin (0) = 0 en abs (sin (x)) ≤ 1.

7.1.5. real procedure cos (x); value x; real x;
Er is voldaan aan de eisen: cos (0) = 1 en abs (cos (x)) ≤ 1.

7.1.6. real procedure arctan (x); value x; real x;
Deze levert de hoofdwaarde van arctan (x); het resultaat ligt tussen - pi/2 en + pi/2. Er is voldaan aan arctan (0) = 0.

7.1.7. real procedure ln (x); value x; real x;
Voor x ≤ 0 wordt voor ln (x) het resultaat ca - $_{10}$628 afgeleverd.
Er is voldaan aan ln (1) = 0.

7.1.8. real procedure exp (x); value x; real x;
Er is voldaan aan exp (0) = 1. Voor x < – 1419 wordt ca $_{10}$–616, voor x > + 1447 wordt ca $_{10}$628 als resultaat van exp (x) afgeleverd.

7.1.9. integer procedure entier (x); value x; real x;
Bij wijze van uitzondering is het waardebereik van entier niet beperkt van – 67 108 863 tot en met + 67 108 863. Het afgeleverde resultaat is exact.

7.2. Input procedures

Als invoerapparaat staat ter beschikking een bandlezer, met een snelheid van 1000 ponsingen/seconde voor 5–, 7–, of 8–gats ponsband. Er zijn drie input procedures voor de bandlezer:

7.2.1. integer procedure REHEP;
De waarde van de function designator REHEP is gelijk aan de getalwaarde van de eerstvolgende ponsing (pentade bij 5–gats band, heptade bij 7–gats band, octade bij 8–gats band) op de band. Deze getalwaarde is minstens 0 en hoogstens 31 respectievelijk 127 respectievelijk 255. Alle ponsingen worden door REHEP gelijkelijk geaccepteerd, zij worden niet op pariteit gecontroleerd en hebben geen invloed op de door RESYM en READ bijgehouden laatste case–definitie.

7.2.2. integer procedure RESYM;
De function designator RESYM kan uitsluitend zinvol worden aangeroepen, als de invoerband een 7–gats band is, geponst in de z.g. MC–flexowriter–code. Alle ponsingen worden gecontroleerd op toelaatbaarheid als flexowriter–symbool. De waarde, die een aanroep van RESYM aflevert is de interne representatie van het eerste flexowriter–symbool op de band, dat verschilt van:
a) de ponsingen blank (0), erase (127), stopcode (11) en backspace (42), die door RESYM overal geskipt worden,
b) de ponsingen lower case (122) en upper case (124),
met inachtneming van de laatste, door RESYM of READ verwerkte case–definitie.
Aan het begin van de uitvoering van een programma wordt de gemeenschappelijke case van RESYM en READ door het systeem op lower case geinitialiseerd.
De interne representatie van flexowriter–symbolen wordt gegeven in tabel III.

7.2.3. real procedure READ;
De function designator READ heeft slechts betekenis als de invoerband geponst is volgens de ponsconventies voor getallenbanden, weergegeven in 8. De waarde van READ is de waarde van het eerstvolgende getal op de band, met een relatieve precisie van ongeveer 12 decimalen, in absolute waarde tussen de grenzen $_{10}$–616 en $_{10}$628 of exact 0. Integers op de band, in absolute waarde kleiner dan 1 099 511 627 776, worden exact ingevoerd.
De relatie <number> = READ is true als de decimale voorstelling van <number> in de ALGOL–tekst en van het door READ op de getallenband aangetroffen getal dezelfde is.
READ maakt voor het lezen uitsluitend gebruik van RESYM.

7.2.4. real procedure read; read:= READ;
read is slechts een andere naam voor READ.

7.2.5. Men zij voorzichtig met het afwisselend gebruiken van "REHEP", "RESYM", en "READ" (of "read"). We beschrijven hier de huidige werking, die zeker nog wel niet definitief zal zijn:

Na een aanroep van "READ" is het laatst gelezen:
a) als de getalscheider (zie 8.) bestaat uit een rij symbolen tussen twee accenten: deze gehele rij symbolen, inclusief de sluitings-accent,
b) als de getalscheider bestaat uit twee of meer spaties: twee spaties,
c) als de getalscheider een ander flexowriter-symbool is: dat symbool.

Een hieropvolgende aanroep van REHEP geeft de getalwaarde van de eerste nog niet gelezen ponsing, een aanroep van RESYM de interne representatie van het eerste nog niet gelezen flexowriter-symbool.

Als de getalscheider bestaat uit een min-teken, zal een volgende aanroep van READ een negatieve waarde afleveren, ongeacht tussengelaste aanroepen van RESYM of REHEP.

Een veilige manier om van READ naar RESYM of REHEP over te gaan is het afsluiten van het laatste door READ te lezen getal door de getalscheider <TWNR>. Bij overgang van REHEP naar RESYM of READ doet men er verstandig aan, het eerste te verwerken symbool te laten voorafgaan door een case-definitie.

7.3. Output procedures

Voorlopig staan ter beschikking als output apparaten:
a) een regeldrukker, met een snelheid van 7 tot 20 regels/seconde. Per pagina zijn beschikbaar 60 regels met een breedte van 144 posities,
b) een bandponser voor 7-gats band met een snelheid van 150 heptades/seconde.
Het gebruik van de bandponser als output apparaat is uitsluitend aan te bevelen voor het uitvoeren van gegevens, die
a) naderhand weer moeten worden ingelezen,
b) in een mooiere typografie dan die van de regeldrukker gepubliceerd moeten worden,
c) mechanisch vermenigvuldigd moeten worden.
Indien men slechts enige eind-antwoorden heeft, maar veel tussenresultaten ook wil uitvoeren om in geval van twijfel deze te kunnen controleren, kan men:
de tussenresultaten over de regeldrukker uitvoeren,
de eind-antwoorden ponsen.

7.3.1. Output procedures voor de regeldrukker

7.3.1.1. procedure PRSYM (n); value n; integer n;

Het effect van PRSYM is slechts gedefinieerd voor de waarden van n, vermeld in tabel III; afgebeeld wordt op de regeldrukker het in die tabel vermelde symbool.
PRSYM (93) is equivalent met SPACE (1),
PRSYM (118) is equivalent met TAB,
PRSYM (119) is equivalent met NLCR.

PRSYM, en via PRSYM alle andere outputprocedures voor de regeldrukker, houden de positie op de regel, en het regelnummer op de pagina bij. Als door een aanroep van PRSYM meer dan 144 symbolen op een regel dreigen te ontstaan, wordt door het systeem een NLCR ingelast; als door een aanroep van PRSYM meer dan 60 regels op een pagina dreigen te ontstaan, wordt overgegaan op een nieuwe pagina.

Aanroepen van PRSYM met n = 126 of n = 127 (onderstreping of doorbalking) verhogen de positie op de regel niet. Om een symbool te onderstrepen geve men eerst de onderstreping, daarna het symbool.

7.3.1.2. procedure SPACE (n); value n; integer n;

SPACE verhoogt in principe de positie op de regel met n. Als daarbij het aantal symbolen op de regel 144 dreigt te overschrijden, last het systeem zoveel regelopvoeren (daarbij telkens de positie op de regel verlagend met 144) in als nodig om de juiste regelbreedte te garanderen. SPACE (n) is equivalent met, maar sneller dan:
begin integer i; for i:= 1 step 1 until n do PRSYM (93) end;

7.3.1.3. procedure TAB;

TAB verhoogt in principe de positie op de regel met minstens 2 en ten hoogste 9, zodanig, dat een 8-voud bereikt wordt (de posities op de regel zijn genummerd van 0 t/m 143. De "tabulatorstoppen" bevinden zich dus op de posities, genummerd 8, 16, 24, ..., 136). In geval van overschrijding van de regelbreedte wordt een regelopvoer ingelast en de positie op de regel met 144 verminderd.

7.3.1.4. procedure NLCR;

NLCR geeft een regelopvoer en stelt de positie op de regel terug op 0. Als door het uitvoeren van NLCR het aantal regels op de pagina de 60 zou overschrijden, wordt in plaats van regelopvoer overgang naar een nieuwe pagina bewerkstelligd.

7.3.1.5. procedure CARRIAGE (n); value n; integer n;

Voor $n < -1$ of $n = 1$ of $n > 31$ is CARRIAGE equivalent met NLCR; voor $n = -1$ is CARRIAGE equivalent met NEW PAGE. Voor de andere waarden van n worden in principe n regelopvoeren gegeven en wordt de positie op de regel op 0 gesteld. Als door uitvoeren van CARRIAGE het aantal regels op de pagina de 60 zou overschrijden, wordt in plaats van de n regelopvoeren, overgang naar een nieuwe pagina bewerkstelligd.

7.3.1.6. procedure NEW PAGE;

NEW PAGE bewerkstelligt de overgang naar een nieuwe pagina en stelt de positie op de regel op 0. Iedere nieuwe pagina wordt door het systeem voorzien van een kopje, vermeldende de datum van verwerking en het serienummer, door het systeem aan deze verwerking toegekend.

7.3.1.7. integer procedure LINE NUMBER;

De waarde van de function designator LINE NUMBER is het nummer van de regel "in opbouw" op de heersende pagina. Deze waarde is minstens 1 en hoogstens 60; na overgang op een nieuwe pagina levert LINE NUMBER de waarde 1 af.

7.3.1.8. procedure ABSFIXT (n,m,x); value n,m,x; integer n,m; real x;

De absolute waarde van x wordt door de procedure ABSFIXT in het algemeen afgedrukt in vaste-komma-representatie met n cijfers voor de decimale punt, m cijfers erna, het geheel voorafgegaan en gevolgd door een spatie. Als $m = 0$, wordt het afdrukken van de decimale punt onderdrukt. In het gehele gedeelte van x worden non-significante nullen door spaties vervangen, uitgezonderd de nul op de eenhedenpositie in het geval $m = 0$ (d.w.z. als het breukgedeelte ontbreekt).

Het af te drukken getal wordt exact op de laatste af te drukken decimaal afgerond. Als hierna zijn absolute waarde $\geq 10 \uparrow n$ is, of als niet voldaan is aan de relaties $n \geq 0$, $m \geq 0$, $n + m \leq 21$, wordt ABSFIXT (n,m,x) door het systeem vervangen door FLOT (13,3,x).

Een aanroep van ABSFIXT (n,m,x) verhoogt in principe de positie op de regel met if $m = 0$ then $n + 2$ else $n + m + 3$. Als hierdoor het aantal symbolen op de regel meer dan 144 zou worden, wordt voor de aanvang van het afdrukken door het systeem een NLCR ingelast.

7.3.1.9. procedure FIXT (n,m,x); value n,m,x; integer n,m; real x;

De procedure FIXT verschilt slechts van de procedure ABSFIXT in dit opzicht, dat in plaats van de spatie die direct aan het eerste cijfer of aan de decimale punt voorafgaat, het teken van x (+ of −) wordt afgedrukt.

7.3.1.10. procedure FLOT (n,m,x); value n,m,x; integer n,m; real x;

De procedure FLOT drukt de waarde van x af in drijvende, decimale representatie. Na het teken van x en de decimale punt volgen een mantisse van n cijfers, het symbool "$_{10}$", het teken van de decimale exponent, de absolute waarde van die exponent in m cijfers (waarbij nonsignificante nullen, behalve op de eenhedenpositie, vervangen worden door spaties), en ten slotte een spatie.

Voor x = 0 worden een mantisse 0 en een decimale exponent 0, beide met het goede aantal cijfers, afgedrukt.

Voor $x \neq 0$ wordt de decimale exponent zo bepaald, dat de mantisse in absolute waarde $\geq .1$ en < 1 is. Als de zo verkregen decimale exponent niet in m cijfers kan worden afgedrukt, of als niet voldaan is aan $1 \leq n \leq 13$, $1 \leq m \leq 3$, wordt FLOT (n,m,x) vervangen door FLOT (13,3,x).

De mantisse wordt exact op de laatste decimaal afgerond.

Een aanroep van FLOT (n,m,x) verhoogt in principe de positie op de regel met n + m + 5. Als hierdoor het aantal symbolen op de regel meer dan 144 zou worden, wordt vooraf door het systeem een NLCR ingelast.

7.3.1.11. procedure PRINT (x); value x; real x;

Als de absolute waarde van x gelijk is aan een geheel getal, kleiner dan 1 099 511 627 776, wordt x afgedrukt volgens FIXT (13,0,x), gevolgd door 6 extra spaties. Zo niet, dan volgens FLOT (13,3,x). In beide gevallen wordt de positie op de regel verhoogd met 21, maar zonodig vooraf een NLCR door het systeem ingelast.

7.3.1.12. procedure print (x); PRINT (x);

print is slechts een andere naam voor PRINT.

7.3.1.13. procedure PRINTTEXT (s); string s;

De actuele parameter bij een aanroep van PRINTTEXT mag uitsluitend zijn: een string, of een formele identifier (het geval van een "doorgegeven" formele parameter), mits deze laatste uiteindelijk met een string correspondeert. De symbolen van de string, ontdaan van de buitenste string quotes, worden, symbool na symbool, afgedrukt. Telkens als daarbij de regelbreedte van 144 posities overschreden dreigt te worden, last het systeem een NLCR in.

7.3.2. Output procedures voor de bandponser

7.3.2.1. procedure PUHEP (n); value n; integer n;

De procedure PUHEP ponst de laatste 7 bits van de binaire representatie van n als 1 heptade op de band. Voor $0 \leq n \leq 127$ wordt dus n zelf als heptade geponst. Voor $n < 0$ is de binaire representatie uit die van abs (n) af te leiden door alle nullen door enen te vervangen en omgekeerd. (Een aanroep van PUHEP (n) met $n < 0$ heeft als neveneffect, dat de ponsbuffer, groot 150 heptades, leeggeponst wordt reeds voor dat hij geheel gevuld is).

Bij iedere aanroep van PUHEP wordt onderzocht, of de waarde van n toelaatbaar is als getalwaarde van een flexowriter-ponsing. Is dit het geval, dan wordt door PUHEP, ten behoeve van hogere ponsprocedures, de laatst geponste case-definitie, en ook de positie op de regel van de flexowriter bijgehouden. PUHEP (26) stelt de positie op de regel op 0, PUHEP (62) verhoogt de positie op de regel met minstens 2 en hoogstens 9,

zodanig dat een 8-voud bereikt wordt, PUHEP (14) wijzigt de positie op de regel niet, PUHEP (122) of PUHEP (124) zetten de laatst geponste case-definitie op lower, respectievelijk upper case. De door PUHEP bijgehouden laatst geponste case-definitie wordt aan het begin van de uitvoering van het programma geinitialiseerd op noch lower, noch upper case.

In alle gevallen wordt een en slechts een heptade geponst.

7.3.2.2. procedure PUSYM (n); value n; integer n;

PUSYM is het ponsende analogon van PRSYM (zie 7.3.1.1.). Het effect van PUSYM is slechts gedefinieerd voor de waarden van n, vermeld in tabel III; geponst wordt het in die tabel vermelde symbool in flexowriter-code, zonodig voorafgegaan door een case-definitie. Dit laatste geschiedt, als de flexowriter-case van het betreffende symbool niet overeenstemt met de door PUHEP bijgehouden laatst geponste case-definitie. Alle aanroepen van PUSYM lopen via PUHEP en beinvloeden derhalve de laatst geponste case-definitie en de positie op de regel als daar beschreven.
PUSYM (93) is equivalent met PUHEP (16) en met PUSPACE (1),
PUSYM (118) is equivalent met PUHEP (62),
PUSYM (119) is equivalent met PUHEP (26) en met PUNLCR;
bij deze symbolen wordt nimmer een case-definitie ingelast.

7.3.2.3. procedure PUSPACE (n); value n; integer n;

PUSPACE ponst in principe n (flexowriter-)spaties op de band.
PUSPACE (n) is equivalent met, maar sneller dan:
begin integer i; for i:= 1 step 1 until n do PUSYM (93) end;

7.3.2.4. procedure PUNLCR;

PUNLCR ponst de flexowriter-ponsing voor terug-wagen-nieuwe-regel. De door PUHEP bijgehouden positie op de regel wordt door PUNLCR op 0 gesteld.

7.3.2.5. procedure RUNOUT;

RUNOUT ponst een stuk blank band (80 ponsingen blank = 20 cm band).

7.3.2.6. procedure STOPCODE;

STOPCODE ponst de flexowriter-stopcode, gevolgd door een stuk blank band. STOPCODE is equivalent met begin PUHEP (11); RUNOUT end;

7.3.2.7. procedure ABSFIXP (n,m,x); value n,m,x; integer n,m; real x;

De procedure ABSFIXP is geheel het ponsende analogon van ABSFIXT (zie 7.3.1.8.). Evenals bij FIXP, FLOP, en PUNCH wordt, als na uitvoering van de procedure de positie op de regel 150 zou overschrijden, vooraf een aanroep van PUNLCR ingelast. Bovendien wordt, als de door PUHEP bijgehouden case-definitie van lower case verschilt, vooraf een lower case-ponsing ingelast.

7.3.2.8. procedure FIXP (n,m,x); value n,m,x; integer n,m; real x;

De procedure FIXP is geheel het ponsende analogon van FIXT (zie 7.3.1.9.). Verder gelden de opmerkingen, gegeven bij de beschrijving van ABSFIXP (zie 7.3.2.7.), ook voor FIXP.

7.3.2.9. procedure FLOP (n,m,x); value n,m,x; integer n,m; real x;

De procedure FLOP is geheel het ponsende analogon van FLOT (zie 7.3.1.10.). Verder gelden de opmerkingen, gegeven bij de beschrijving van ABSFIXP (zie 7.3.2.7.), ook voor FLOP.

7.3.2.10. procedure PUNCH (x); value x; real x;
De procedure PUNCH is geheel het ponsende analogon van PRINT (zie 7.3.1.11.) Verder gelden de opmerkingen, gegeven bij de beschrijving van ABSFIXP (zie 7.3.2.7.), ook voor PUNCH.

7.3.2.11. procedure PUTEXT (s); string s;
De procedure PUTEXT is het ponsende analogon van PRINTTEXT (zie 7.3.1.13.). Echter reageert PUTEXT nooit op een eventuele overschrijding van de regelbreedte van de flexowriter, al wordt wel de positie op de regel nauwkeurig bijgehouden. PUTEXT last zo min mogelijk case-ponsingen in en maakt hierbij gebruik van de door PUHEP bijgehouden laatst geponste case-definitie.

7.4. Berekening-sturende input procedures

Uitsluitend met toestemming van, en in overleg met de contactpersoon aan de "balie", mogen de procedures HAND en XEEN gebruikt worden, die de operateur in staat stellen, tijdens de uitvoering van programma's de loop van de berekening door manuele invoer van getallen te beinvloeden. Gebruik van HAND en XEEN sluit de programma's uit van de twee-minuten-dienst en zal, door de tijd, die de operateur nodig heeft voor zijn beslissingen en handelingen, uiterst kostbaar zijn.

7.4.1. real procedure HAND (n); value n; integer n;
Door HAND wordt, op een nieuwe regel van de bedienings-teleprinter, uitgetypt het woord HAND, gevolgd door de absolute waarde van n in ten hoogste 6 cijfers (dit laatste dient ter identificatie). De operateur kan nu, volgens de conventies van de procedure READ (zie 7.2.3. en 8.), een decimaal getal invoeren; de waarde hiervan wordt aan de function designator HAND toegekend. Bovendien geldt, voor zolang het duurt, deze waarde als het "laatst ingevoerde decimale getal" in foutmeldingen tijdens de executiefase van het programma. Het is aan te bevelen, voor een verslaglegging van de loop der berekeningen, de waarde van HAND meteen over de regeldrukker uit te voeren.

7.4.2. integer procedure XEEN (n); value n; integer n;
De waarde van de function designator XEEN is een functie van twee parameters; de eerste is niets anders dan het argument van XEEN, n, bij iedere aanroep van XEEN in principe verschillend, de tweede parameter daarentegen is een systeemgrootheid, ook integer, die in principe van aanroep tot aanroep van XEEN ongewijzigd blijft.
De waarde van XEEN wordt uit deze twee parameters gevormd door "bit voor bit colleren". In de binaire voorstelling van het resultaat van deze operatie staat dan en slechts dan een 1, als op de overeenkomstige positie in beide operanden ook een 1 staat. (In de X8 wordt de binaire representatie van een negatieve integer uit die van het tegenstelde getal verkregen door alle nullen door enen te vervangen en omgekeerd. Het meest significante bit fungeert hierdoor onder andere als tekenbit. Er zijn in principe twee representaties van de integer 0: een rij van 27 nullen, of een rij van 27 enen. Als het argument van XEEN gelijk 0 is, is de binaire voorstelling hiervan echter steeds een rij van 27 enen).
Bij de eerste aanroep van XEEN die in het programma wordt uitgevoerd, wordt, op een nieuwe regel van de bedienings-teleprinter, het woord XEEN uitgetypt. De operateur krijgt hiermee gelegenheid, de bovengenoemde systeemgrootheid, die verder ook bij iedere volgende aanroep van XEEN met het argument van XEEN gecolleerd zal worden, octaal in te voeren. Hiertoe denke men zich de 27 bits van een woord gesplitst in 9 groepen van 3 bits elk; iedere groep van 3 bits vatte men op als een octaal

cijfer (0 t/m 7). De operateur dient nu van de programmeur deze 9 octalen, als octaal getal geschreven, op te krijgen; eventuele nullen op non-significante posities mogen daarbij worden weggelaten.

Er zijn twee manieren, waarop een eenmaal aan het systeem opgegeven waarde nog kan worden gewijzigd:
a) door het programma, middels een aanroep van de procedure STOP,
b) door de operateur, op elk gewenst moment, door op de bedienings-teleprinter de toets X aan te slaan.
In beide gevallen zal het programma normaal doorrekenen. Echter zal bij de eerstvolgende aanroep van XEEN die in het programma wordt geeffectueerd, aan de operateur opnieuw de gelegenheid gegeven worden, aan het systeem een waarde op te geven. De gang van zaken is dan geheel als boven beschreven.

7.4.3. procedure STOP;

De procedure statement STOP heeft als enig effect, dat bij de eerstvolgende aanroep van XEEN niet langer de eventueel eerder opgegeven waarde van de systeemgrootheid, waarmee het argument van XEEN telkens gecolleerd wordt, gebruikt wordt, maar dat dan aan de operateur om een nieuwe waarde gevraagd wordt.

7.5. Diverse procedures

7.5.1.
```
real procedure SUM (i,a,b,x); value b; integer i,a,b; real x;
begin   real s; s:= 0;
        for i:= a step 1 until b do s:= s + x;
        SUM:= s
end;
```
De function designator SUM levert de som af van de waarden van de met x corresponderende actuele expressie, uitgerekend voor waarden van i = a (1) b. Voor b < a levert SUM de waarde 0 af.

7.5.2.
```
real procedure INPROD (i,a,b,x,y); value b; integer i,a,b; real
x,y;
begin   real s; s:= 0;
        for i:= a step 1 until b do s:= s + x × y;
        INPROD:= s
end;
```
De function designator INPROD levert af het inwendig product van twee vectoren van getalwaarden, verkregen door de met x, respectievelijk y corresponderende actuele expressies te evalueren voor i = a (1) b. Voor b < a levert INPROD de waarde 0 af.

7.5.3.
```
integer procedure EVEN (n); value n; integer n;
EVEN:= if n + 2 × 2 = n then 1 else - 1;
```
De function designator EVEN is equivalent met wat in mathematische notatie vaak geschreven wordt als $(-1)^n$.

7.5.4.
```
integer procedure REMAINDER (a,b); value a,b; integer a,b;
REMAINDER:= if b = 0 then a else a - a + b × b;
```
De function designator REMAINDER levert, tenzij b = 0, de rest af van de deling van a door b (de rest gedefinieerd als het in absolute waarde kleinste getal r met hetzelfde teken als a, waarvoor a = r (mod b) geldt).

7.5.5. real procedure RANDOM;

Opeenvolgende aanroepen van RANDOM geven min of meer homogeen verdeelde trekkingen uit het eenheidsinterval [0,1). Iets nauwkeuriger omschreven, iedere aanroep van RANDOM levert een waarde, ≥ 0 en < 1, gelijk

aan het eerstvolgende getal uit een pseudo-random rij, gegenereerd volgens een proces van D.H.Lehmer (zie b.v. M.Greenberger, MTAC 15 (1961) 383). De periode van dit proces is 2 ↑ 26.
Het is, in verband met de eis van reproduceerbaarheid van de resultaten van een programma, dringend gewenst, de eerste aanroep van RANDOM in het programma te laten voorafgaan door een aanroep van SETRANDOM (zie 7.5.6.).

7.5.6. procedure SETRANDOM (x); value x; real x;
De procedure SETRANDOM definieert met behulp van x de startwaarde voor het proces, waarmee de function designator RANDOM zijn pseudo-random trekkingen genereert. Het argument x van SETRANDOM moet voldoen aan $0 \leq x < 1 - 2 \uparrow (-27)$.

7.5.7. procedure EXIT;
De procedure EXIT beeindigt de executiefase van het programma, alsof de laatste end van het programma "gepasseerd" was.

8. Ponsconventies voor het ponsen van getallenbanden

Getallenbanden, die gelezen moeten worden door de function designator READ (zie 7.2.3.), moeten geponst worden op 7-gats band in de z.g. MC-flexowriter-code.

8.1. Een getal is een <unsigned number> in de zin van het Revised Report, al dan niet voorafgegaan door een teken. Elk getal op de band moet worden afgesloten door een getalscheider (zie 8.3.). Extra getalscheiders voor het begin van een getal worden door READ geskipt.

8.2. Als lay out symbolen binnen een getal zijn toegestaan:
a) na het teken van het getal, na $_{10}$, of na het teken van de exponent, maar voor het eerste cijfer van getal of exponent: willekeurig veel spaties of tabulaties,
b) op elke andere plaats: telkens hoogstens 1 spatie.

8.3. Als getalscheider fungeren:
a) het teken van het volgende getal,
b) twee of meer spaties of een tabulatie, behalve op de in 8.2. genoemde posities,
c) overgang op een nieuwe regel,
d) elke rij flexowriter-symbolen tussen twee accenten,
e) elk flexowriter-symbool dat verschilt van . of $_{10}$ of + of - of een cijfer.
N.B.: een enkele spatie bewerkstelligt geen getalscheiding.

Voorbeeld: als op een getallenband voorkomt de symbolenrij:
A[j3]:= - 3.14 $_{10}$- 3 × 15;
levert READ achtereenvolgens de waarden:
3, -.00314, en 15 af.

8.4. Het is toegestaan een getallenverzameling over meer dan een band te verdelen. Bandeinde heeft geen betekenis als getalscheider: na het laatste getal op de laatste getallenband moet dus nog een getalscheider staan.

8.5. Elke getallenband moet beginnen en eindigen met ten minste 25 cm blank. Op iedere plaats zijn verder, zonder enige betekenis, toegestaan de symbolen blank (0), erase (127), stopcode (11), en backspace (42). Na tape feed behoeft geen nieuwe case-definitie gegeven te worden. Aan het

begin van de executiefase van een programma wordt de gemeenschappelijk door RESYM en READ bijgehouden case-definitie door het systeem op lower case geinitialiseerd.

8.6. Als eerste getallenband functioneert het gedeelte van de (laatste) programmaband, dat volgt direct op het eerste symbool na de laatste end van het programma. Dit betekent onder meer het volgende:
a) getallenmateriaal mag direct na het programma geponst worden,
b) als de programmaband eindigt met tape feed en eventueel een stopcode, wordt dit stuk door READ (maar niet door REHEP) geskipt,
c) als het programma abusievelijk een end teveel bevat, zal de ALGOL-tekst gedeeltelijk als getallenband behandeld worden.

9. De tijdsduur van enige operaties

Daar de tijdsduur van een operatie in het ALGOL-systeem voor de X8 als regel van de specifieke syntactische constructie afhangt, zijn de volgende cijfers zeer globaal. In het bijzonder zal het evalueren van formele identifiers soms relatief lang duren. Echter maken de hier gegeven getallen een ruwe schatting van de benodigde tijd voor bepaalde programmaonderdelen mogelijk. Alle tijden zijn opgegeven in μs.

9.1. De diadische arithmetische operaties:
a) integer operanden:

+ of −	8	of	20
÷	190		

voorts als bij real operanden,
b) real operanden:

+ of −	13	of	25
×	40	of	52
/	65	of	77
↑ 2	290		
↑ 3.14	1500		

Als twee waarden opgegeven worden, geldt de kleinste waarde voor het geval, dat de tweede operand eenvoudig is (een constante of een simpele niet-formele identifier).

9.2. De logische operaties:

¬	5
∧, ∨, ≡, ⊃	21

9.3. Indiceren:

a) integer of real array	50 + 85 per indexpositie
b) Boolean array	150 + 85 per indexpositie

9.4. Assignments:

a) aan een real	15
b) aan een integer	16
c) aan een Boolean	14

9.5. For statements:

for i:= 1 step 1 until n do	80 per repetitie-slag

9.6. Blokingang en -verlating: 45

9.7. Array-declaratie: 150 + 100 per indexpositie

9.8. Procedure-ingang en -verlating:

110 + 70 per formele parameter

9.9.

abs	13
sign	13
entier	80
sqrt	340
sin	470
cos	450
arctan	725
ln	580
exp	735

Tabel I: Omschrijving van de betekenis van foutnummers

Ia) fouten tegen de syntaxis

100 in parameterscheider ontbreekt de colon
101 in parameterscheider ontbreekt de (
102 op de band staat een symbool van foute pariteit
103 op de band komt een onbekende ponsing voor
104 in declaratie wordt own niet gevolgd door <type>
105 in declaratie wordt own niet gevolgd door type- of arraydeclaratie
106 in declaratie wordt <type> gevolgd door switch
107 in specificatie wordt <type> gevolgd door label of switch
108 -
109 na $_{10}$ of . volgt geen getal
110 geen identifier waar vereist
111 programma begint met identifier, niet gevolgd door colon
112 programma begint met getal, niet gevolgd door colon
113 programma is geen compound statement of block
114 in procedure declaratie is de formele parameterlijst niet afgesloten met)
115 in procedure declaratie volgt op het <formal parameter part> geen semicolon
116 in value list staat een identifier die niet onder de formelen voorkomt
117 value list niet afgesloten door een semicolon
118 in specificatie staat een identifier die niet onder de formelen voorkomt
119 formele parameter wordt meer dan eens gespecificeerd
120 specificatie ontoelaatbaar voor parameter uit value list
121 specificatie niet door semicolon afgesloten
122 in typedeclaratie komt een reeds eerder in hetzelfde blok gedeclareerde identifier voor
123 in arraydeclaratie komt een reeds eerder in hetzelfde blok gedeclareerde identifier voor
124 in arraydeclaratie deugt de bound pair list niet
125 in arraydeclaratie ontbreekt de bound pair list
126 bij switchdeclaratie komt een reeds eerder in hetzelfde blok gedeclareerde identifier voor
127 bij proceduredeclaratie komt een reeds eerder in hetzelfde blok gedeclareerde identifier voor
128 declaratie niet door semicolon afgesloten
129 label in hetzelfde blok reeds eerder gedeclareerd of als label voor statement verschenen
130 numerieke label buiten de integercapaciteit of niet-integer

200 -
201 formele parameter uit value list niet gespecificeerd
202 te veel localen of labels in blok
203 in arraydeclaratie deugt de bound pair list niet
204 identifier onbekend

300 in arithmetische expressie is een ifclause niet besloten met then
301 in arithmetische expressie ontbreekt een else-deel
302 in arithmetische expressie ontbreekt een)
303 in arithmetische expressie begint een primary met ontoelaatbaar symbool

304 in arithmetische expressie staat een niet-arithmetische identifier
305 array- of switchidentifier wordt niet gevolgd door subscript list
306 subscript list niet afgesloten door]
307 in Boolean expressie is een ifclause niet besloten met then
308 in Boolean expressie ontbreekt een else-deel
309 in Boolean expressie ontbreekt een)
310 in Boolean expressie begint een Boolean primary met ontoelaatbaar symbool
311 in Boolean expressie wordt een arithmetisch gedeelte niet gevolgd door een relational operator
312 in Boolean expressie staat een niet-Boolean identifier
313 in arithmetische of Boolean expressie is een ifclause niet besloten met then
314 in arithmetische of Boolean expressie ontbreekt een else-deel
315 in arithmetische of Boolean expressie ontbreekt een)
316 arithmetische of Boolean expressie begint met ontoelaatbaar symbool
317 in arithmetische of Boolean expressie staat identifier van type string of van designational type
318 in stringexpressie is een ifclause niet besloten met then
319 in stringexpressie ontbreekt een else-deel
320 in stringexpressie ontbreekt een)
321 in stringexpressie komt ontoelaatbaar symbool voor
322 in stringexpressie staat een identifier, niet van type string
323 in designational expressie is een ifclause niet besloten door then
324 in designational expressie ontbreekt een else-deel
325 in designational expressie ontbreekt een)
326 in designational expressie komt een onbekende numerieke label voor
327 in designational expressie komt ontoelaatbaar symbool voor
328 in designational expressie staat een niet-designational identifier
329 Boolean- of stringexpressie in plaats van arithmetische of designational expressie
330 designational expressie in plaats van een <type>-expressie
331 in expressie is een ifclause niet besloten door then
332 in expressie ontbreekt een else-deel
333 in expressie ontbreekt een)
334 in expressie komt ontoelaatbaar symbool voor
335 statement begint met variabele, niet gevolgd door een colonequal
336 assignment aan formele type-procedure identifier
337 assignment aan functie-identifier buiten de declaratie
338 in left part list komt na een integer variabele een non-integer variabele voor
339 in left part list komt na een real variabele een non-real variabele voor
340 in left part list komt na een Boolean variabele een niet-logische variabele voor
341 in left part list komt na een stringvariabele een niet-stringvariabele voor
342/343 in left part list komt na een arithmetische variabele een niet-arithmetische variabele voor
344/345 in left part list komt een designational identifier voor
346 in expressie komt een non-type procedure identifier voor
347 statement begint met identifier op ontoelaatbare wijze
348 actuele parameter is een identifier maar geen array- of switch-identifier
349 actuele parameter is een identifier maar geen procedure identifier
350 actuele parameter is een identifier maar geen type-procedure identifier
351 actuele parameter is een array-, switch- of procedure identifier

352 actuele parameter is een expressie in plaats van een array-, switch- of procedure identifier
353 actuele parameter is geen variabele waaraan geassigneerd kan worden
354 actuele parameter is een expressie maar geen arithmetische
355 actuele parameter is een expressie maar geen logische
356 actuele parameter is een expressie maar geen stringexpressie
357 actuele parameter is een expressie maar geen designational
358 actuele parameter is een expressie maar designational
359 te veel actuele parameters
360 te weinig actuele parameters
361 actuele parameterlijst niet afgesloten met)
362 actuele parameterlijst ontbreekt
363 te veel of te weinig actuele parameters bij bibliotheek procedure
364 statement begint met een getal dat geen integer label is
365 in statement volgt na then een conditional statement
366 na for statement volgt een else-deel
367 statement niet correct
368 in statement is een ifclause niet besloten door then
369 for list niet met do besloten
370 van for statement is de controlled variable niet-arithmetisch
371 in for statement staat in plaats van een controlled variable een type-procedure identifier
372 in for statement volgt op de controlled variable geen colonequal
373 in step-until-element van for list ontbreekt de until
374 in for list volgt op arithmetische expressie geen step, while, do of comma
375 for wordt niet gevolgd door een identifier
376 switch list bevat een niet-designational identifier
377 switch list bevat een getal dat niet als integer label bekend is
378/379 switch list niet correct
380 in switch declaratie volgt op de switch identifier geen colonequal
381 in switch declaratie volgt op switch geen identifier
382 in array declaratie ontbreekt de bound pair list
383 in bound pair list volgt op een lower bound geen colon
384 bound pair list niet afgesloten door een]
385 declaratie niet door semicolon afgesloten
386/387 -
388 statement begint met switch identifier
389 op label volgt geen colon
390 in type-procedure declaratie komt geen assignment aan de procedure identifier voor
391 identifier reeds eerder gedeclareerd of als label bekend geworden
392 goto statement leidt binnen een for statement
393 subscript list bevat te veel of te weinig subscript expressies
394 actuele parameter is een identifier van een array, switch of procedure met onjuist aantal subscripts of parameters
395 actuele parameter is een identifier van onjuist type
396 in code body ontbreekt een comma als scheider tussen macronummer en bijbehorende parameter
397 in code body volgt na een minus geen getal
398 in code body begint een parameter niet met letter, cijfer of minus
399 in code body als macronummer geen getal
400 code body niet afgesloten met unquote
401 code body niet aangevangen met quote

490 wegens de reeds eerder gemelde fouten wordt het syntactisch onderzoek niet verder voortgezet
491 het programma bevat een switchdeclaratie met teveel entries in de switch list
492 het programma is te lang voor het beschikbare geheugen

Ib) fouten ontdekt tijdens de executiefase

500 actuele parameter is geen variabele waaraan geassigneerd kan worden
501 in arraydeclaratie is een bovengrens kleiner dan de bijbehorende ondergrens
502 van een arrayelement valt de (laatste) index buiten de range
503 van een arrayelement valt een index (maar niet de laatste) buiten de range
504 van een switch designator valt de index buiten de range
505 bij een assignment aan een arrayelement van integer type ligt de te assigneren waarde buiten de integercapaciteit
506 bij een assignment aan een arrayelement van een formeel array blijkt de corresponderende actuele parameter de naam van een switch te zijn
507/508/509 de formele, geindiceerde controlled variable in een for statement blijkt niet-arithmetisch te zijn
510 de integercapaciteit blijkt, bij afronding tot een integer, overschreden te zijn
511 bij de operatie ÷ zijn niet beide operanden van integer type
515 in een getallenband vindt "READ" of "RESYM" een symbool van foute pariteit
516 in een getallenband vindt "READ" of "RESYM" een onbekende ponsing

600/601/602/603/604 het programma bevat niet-geimplementeerde stringoperaties
605 het programma declareert own arrays
606/607/608 het programma bevat niet-geimplementeerde stringoperaties
609 de geheugenruimte is uitgeput
610 het programma bevat niet-geimplementeerde stringoperaties

999 het programma is door de operateur beeindigd

Tabel II: Interne codering van basic symbols

0	0	90	:
..	...	91	;
9	9	92	:=
10	a	93	space
..	...	94	if
35	z	95	then
37	A	96	else
..	...	97	comment
62	Z	98	(
		99	)
64	+	100	[
65	−	101	]
66	×	102	<
67	/	103	>
68	↑	104	begin
69	↑	105	end
70	=	106	own
71	≠	107	real
72	<	108	integer
73	≤	109	Boolean
74	≥	110	string of step
75	≥	111	array
76	¬	112	procedure
77	≡	113	switch
78	⊃	114	label
79	∨	115	value
80	∧	116	true
81	goto	117	false
82	for	118	tab
84	until	119	new line
85	while	120	'
86	do	121	"
87	,	122	?
88	.	126	_
89	10		
		127	|

160	| gevolgd door een niet toegestaan symbool
161	een niet geoorloofde onderstreping
162	een onbekende word delimiter

Tabel III: Waardebereik van RESYM, PRSYM, en PUSYM

waarde	flexowriter-representatie	regeldrukker-representatie
0	0	0
1	1	1
2	2	2
3	3	3
4	4	4
5	5	5
6	6	6
7	7	7
8	8	8
9	9	9
10	a	A
11	b	B
12	c	C
13	d	D
14	e	E
15	f	F
16	g	G
17	h	H
18	i	I
19	j	J
20	k	K
21	l	L
22	m	M
23	n	N
24	o	O
25	p	P
26	q	Q
27	r	R
28	s	S
29	t	T
30	u	U
31	v	V
32	w	W
33	x	X
34	y	Y
35	z	Z
37	A	𝕬
38	B	𝕭
39	C	𝕮
40	D	𝕯
41	E	𝕰
42	F	𝕱
43	G	𝕲
44	H	𝕳
45	I	𝕴
46	J	𝕵
47	K	𝕶
48	L	𝕷
49	M	𝕸
50	N	𝕹
51	O	𝕺
52	P	𝕻
53	Q	𝕼
54	R	𝕽
55	S	𝕾

waarde	flexowriter-representatie	regeldrukker-representatie
56	T	T
57	U	U
58	V	V
59	W	W
60	X	X
61	Y	Y
62	Z	Z
64	+	+
65	−	−
66	×	×
67	/	/
70	=	=
72	<	<
74	>	>
76	¬	¬
79	∨	∨
80	∧	∧
87	,	,
88	.	.
89	10	10
90	:	:
91	;	;
93	spatie	spatie
94	ongedefinieerd	0
95	ongedefinieerd	\
96	ongedefinieerd	#
98	(	(
99	)	)
100	[	[
101	]	]
118	tab	tab
119	twnr	twnr
120	'	'
121	"	"
122	?	?
126	←	←
127	|	|

Aanvullingen op MR 81

Met ingang van 20 februari 1966 is het MC-ALGOL 60-systeem voor de X8 op de volgende punten gewijzigd:

1 Aan de regeldrukker-output van elk programma wordt door het systeem een extra pagina toegevoegd, waarop de volgende gegevens vermeld worden:

1.1 Als het programma syntactisch incorrect is:
op regel 1 de tijd, besteed aan de syntactische controle, in milli-uren,

op regel 2 de som van de heptaden van de programmaband,

op regel 3 een schatting voor het aantal woorden in het geheugen van de X8, voor het programma zelf benodigd na verbetering van de fouten.

1.2 Als het programma syntactisch correct is, en ook in uitvoering genomen is:
op regel 1 de tijd, besteed aan de syntactische controle, gevolgd door de totale tijd, dat het programma op de X8 is geweest, beide tijden in milliuren,

op regel 2 de som van de heptaden van de programmaband, gevolgd door de som van alle heptaden, die bij de controle en de uitvoering verwerkt zijn,

op regel 3 het aantal woorden, in het geheugen van de X8 in beslag genomen door het programma zelf.

Indien een programma resultaten over de bandponser uitgevoerd heeft:
op regel 4 het aantal geponste heptaden, gevolgd door de som van de geponste heptaden (inclusief standaardkop en standaardafsluiting).

2 Aan de bibliotheek van standaardprocedures is toegevoegd:

2.1 real procedure time;
Deze levert de tijd af, die verlopen is sinds het in behandeling nemen van het programma (dus vanaf het begin van de syntactische controle), in seconden (dus niet in milliuren) met een nauwkeurigheid van 0.01 sec.

2.2 procedure TO DRUM (A, p); value p; array A; integer p;

procedure FROM DRUM (A, p); value p; array A; integer p;

Beide procedures nemen contact op met het trommelgeheugen, de eerste om de elementen van het array A op de trommel op te bergen, de tweede om het array A vanaf de trommel te vullen. Voor dit dumpen en weer ophalen van array-inhouden staat op de trommel een stuk van 81920 woorden ter beschikking. De parameter p geeft aan, welk van deze woorden met het eerste element van

het array moet corresponderen. Steeds moet $p \geq 0$ en p + aantal woorden, in beslag genomen door elementen van het array (zie MR 81 sectie 6) moet $\leq$ 81920 zijn.

De tijdsduur van het transport van of naar de trommel kan geschat worden als 50 milliseconden per 1000 woorden met een minimum van 25 m s. Het is dus niet efficient om arrays van kleine omvang te dumpen. Wel wordt gedurende het transport van een array reeds verder gerekend, zolang het programma maar geen gebruik maakt van het array in kwestie of het blok, waarin dat array gedeclareerd is, verlaat. Een uitvoeriger beschrijving van TO DRUM en FROM DRUM zal later gegeven worden.

3. Steeds staan tijdens de executiefase van een programma zeker 23.000 woorden ter beschikking voor programma en werkruimte. Een foutmelding met foutnummer 609 betekent, dat die ruimte te kort schiet. Een nieuwe aanbieding zonder wijzigingen van programma of invoer is dan zinloos.

4 Nieuwe foutmeldingen zijn:

402 procedure body is geen statement

520 bij een aanroep van TO DRUM of FROM DRUM is de eerste actuele parameter geen array identifier

521 bij een aanroep van TO DRUM of FROM DRUM is de tweede parameter te groot of te klein

997 het programma is beeindigd wegens overschrijding van de tijdslimiet

998 het programma is afgebroken wegens ontbreken van verdere input.

5 Bij aanbieden van een programma dient een schatting van de tijdsduur van uitvoering van het programma te worden opgegeven. hetzij in minuten (bij voorbeeld 15') hetzij in milliuren (bij voorbeeld 40 mh) De operateur heeft het recht het programma na de vermelde tijd af te breken, maar is daartoe niet verplicht. De Rekendienst draagt geen verantwoordelijkheid voor overschrijdingen van opgegeven tijden. De programmeur kan zelf voor limitering van de tijd zorgen door geregeld in zijn programma de bibliotheekprocedure "time" aan te roepen, en op de waarde hiervan passend te reageren.

6 Aan de bibliotheek van standaardprocedures zijn toegevoegd:

6.1 een aantal procedures voor het berekenen van inwendige produkten. Deze kunnen in plaats van INPROD gebruikt worden en zijn enige malen sneller. Ze berekenen:

vecvec (l,u,shift,a,b): $\sum_{k=l}^{u} a_k \times b_{k+shift}$

matvec (l,u,i,a,b): $\sum_{k=l}^{u} a_{ik} \times b_k$

tamvec (l,u,i,a,b): $\sum_{k=l}^{u} a_{ki} \times b_k$

matmat (l,u,i,j,a,b): $\sum_{k=l}^{u} a_{ik} \times b_{kj}$

tammat (l,u,i,j,a,b): $\sum_{k=l}^{u} a_{ki} \times b_{kj}$

mattam (l,u,i,j,a,b): $\sum_{k=l}^{u} a_{ik} \times b_{jk}$

De procedures zijn equivalent met de volgende ALGOL-declaraties:

```
comment mca 2000;
real procedure vecvec (l, u, shift, a, b); value l, u, shift;
integer l, u, shift; array a, b;
begin     integer k;
          real s;
          s:= 0;
          for k:= l step 1 until u do s:= a[k] × b[shift + k] + s;
          vecvec:= s
end vecvec;

comment mca 2001;
real procedure matvec (l, u, i, a, b); value l, u, i;
integer l, u, i; array a, b;
begin     integer k;
          real s;
          s:= 0;
          for k:= l step 1 until u do s:= a[i,k] × b[k] + s;
          matvec:= s
end matvec;

comment mca 2002;
real procedure tamvec(l, u, i, a, b); value l, u, i;
integer l, u, i; array a, b;
begin     integer k;
          real s;
          s:= 0;
          for k:= l step 1 until u do s:= a[k,i] × b[k] + s;
          tamvec:= s
end tamvec;

comment mca 2003;
real procedure matmat (l, u, i, j, a, b); value l, u, i, j;
integer l, u, i, j; array a, b;
```

```
begin       integer k;
            real s;
            s:= 0;
            for k:= 1 step 1 until u do s:= a[i,k] × b[k,j] + s;
            matmat:= s
end matmat;

comment mca 2004;
real procedure tammat (l, u, i, j, a, b); value l, u, i, j;
integer l, u, i, j; array a, b;
begin       integer k;
            real s;
            s:= 0;
            for k:= 1 step 1 until u do s:= a[k,i] × b[k,j] + s;
            tammat:= s
end tammat;

comment mca 2005;
real procedure mattam (l, u, i, j, a, b); value l, u, i, j;
integer l, u, i, j; array a, b;
begin       integer k;
            real s;
            s:= 0;
            for k:= 1 step 1 until u do s:= a[i,k] × b[j,k] + s;
            mattam:= s
end mattam;
```

Toepassingen van genoemde procedures bij het oplossen van lineaire stelsels en het bepalen van eigenwaarden zullen in een binnenkort te verschijnen rapport gegeven worden.

6.2 `integer procedure` STRINGSYMBOL (k, text); `value` k;
`integer` k; `string` text;
Deze levert de RESYM-waarde van het k-de symbool uit de string text, waarbij k begint te tellen bij 0. Voor k < 0 of k > aantal symbolen uit de string wordt 255 als waarde van STRINGSYMBOL gegenereerd.

7. Nieuwe foutmeldingen zijn:

522	bij een aanroep van een van de procedures vecvec t/m mattam is een van de laatste twee actuele parameters geen array identifier van real of integer type
523	bij een aanroep van een van de procedures vecvec t/m mattam is de dimensie van een der array identifiers incorrect
611	bij een aanroep van PRINTTEXT, PUTEXT of STRINGSYMBOL is (een van) de actuele parameter(s) geen string

Aanvullingen II op MR 81

8 Met ingang van 1 maart 1970 is het MC-ALGOL 60-systeem voorzien van een gemakkelijk uit te breiden bibliotheek van ALGOL 60 procedures, welke zonder declaratie in een ALGOL 60 programma aangeroepen kunnen worden.
In deze bibliotheek zijn dan, behalve de procedures van sectie 7 en de procedures van de eerste aanvullingen, opgenomen:

1 De procedures beschreven in:
T.J. Dekker, ALGOL 60 procedures in numerical algebra part 1, MC Tracts 22
Mathematisch Centrum Amsterdam 1968.
T.J. Dekker, W. Hoffmann,
ALGOL 60 procedures in numerical algebra part 2, MC Tracts 23
Mathematisch Centrum Amsterdam 1968.

2 De procedure available, waarvan de heading luidt:
integer procedure available;
Deze levert het aantal nog beschikbare geheugenwoorden op het moment van aanroep.
Men zij zeer voorzichtig in het gebruik van available in verband met:
a) Een ALGOL 60 programma heeft voor zijn uitvoering "leefruimte" nodig; dit is meer naarmate de expressies ingewikkelder zijn, dit kan zeer groot zijn wanneer procedures recursief worden aangeroepen. In vele gevallen is de reservering van 100 geheugen woorden voor die "leefruimte" voldoende zodat men de rest van het geheugen kan gebruiken voor het declareren van arrays.
b) Het huidige bedrijfssysteem is niet definitief. Het resultaat van het volgende programma:
begin print(available) end
behoeft daarom niet elke dag hetzelfde te zijn. Er wordt echter naar gestreefd om op zijn minst ca 35 000 woorden ter beschikking te stellen voor het objectprogramma en zijn werkruimte. Men zie voorts sectie 6.1 waar het getal 18 000 vervangen moet worden door 35 000. De mededeling betreffende "zonder protectie" in sectie 6.2 is vervallen.

Treedt tijdens executie binnen een bibliotheek procedure een fout op dan wordt in de foutmelding (zie 4.2.3 en 4.2.4) het eventuele regelnummer vermeld van de "laatst aangevangen statement" uit de programma tekst.

Nieuwe foutmeldingen zijn:

493 het programma is na toevoeging van bibliotheek procedures te lang voor het beschikbare geheugen. In plaats van het laatst ingevoerde getal wordt de lengte van het totale objectprogramma afgedrukt

495 het programma is te lang