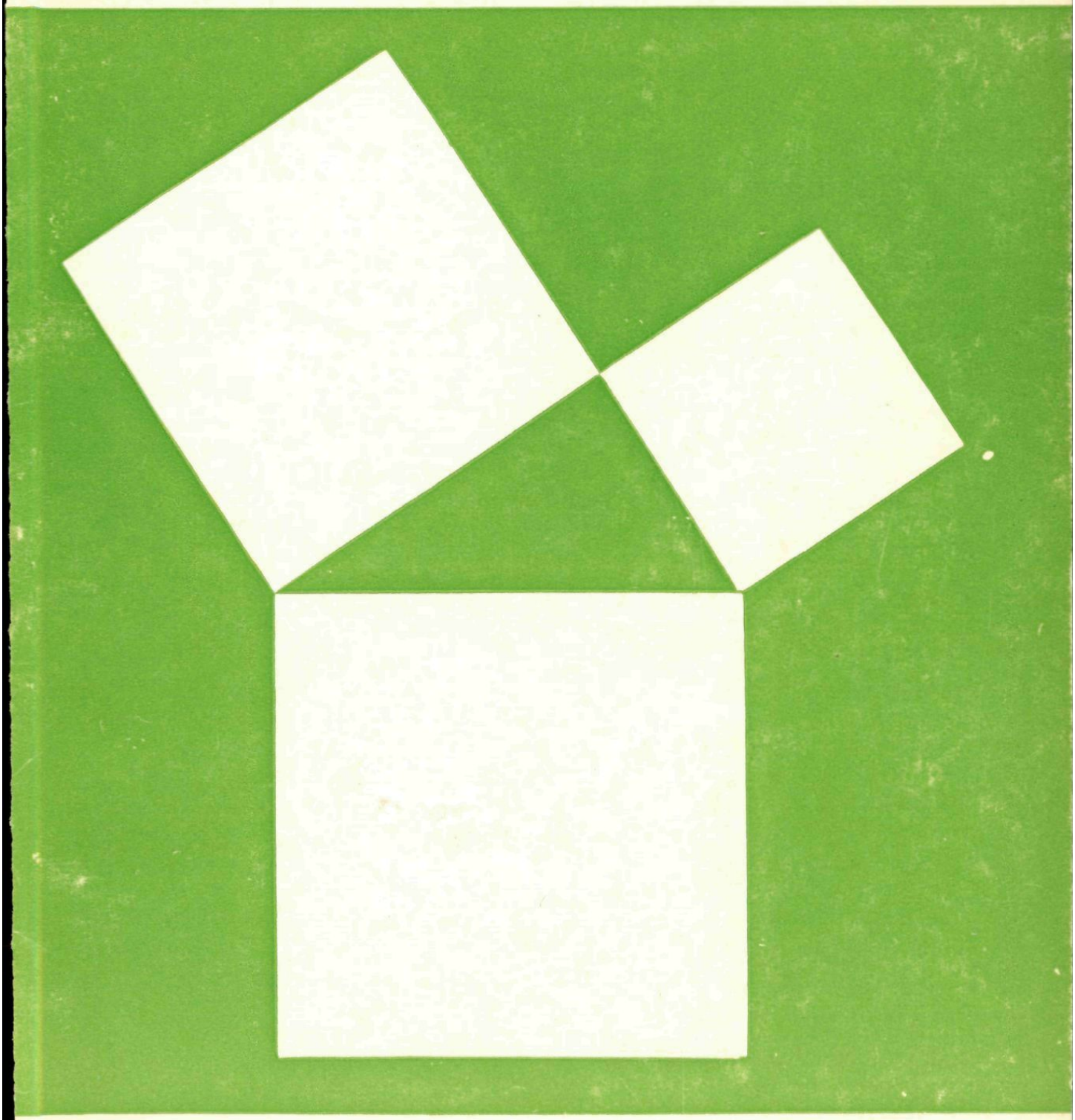
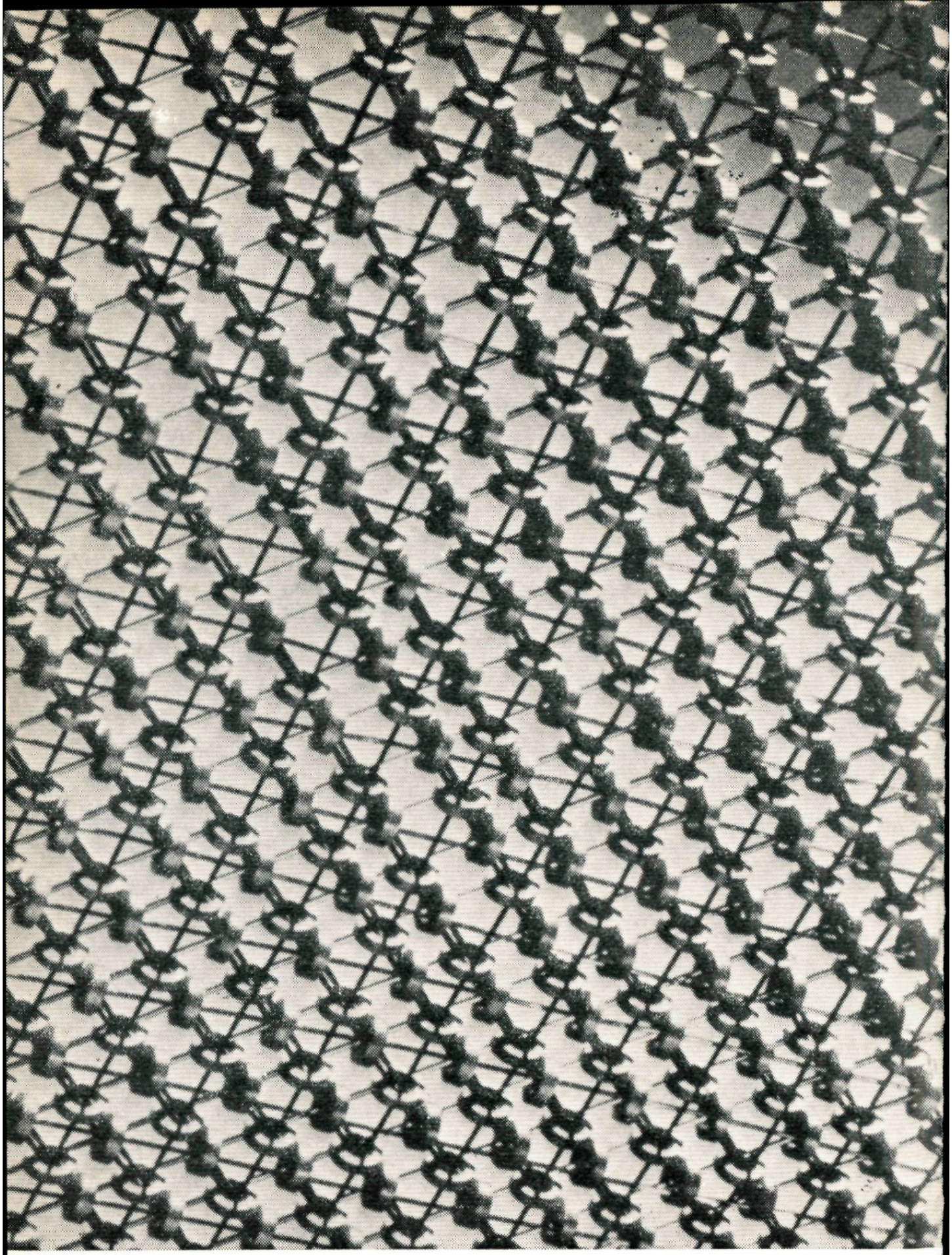
PYTHAGORAS



Wiskundetijdschrift
voor jongeren

5

Achtste jaargang
1968/1969



Deel van kerngeheugen (sterk vergroot)

pythagoras

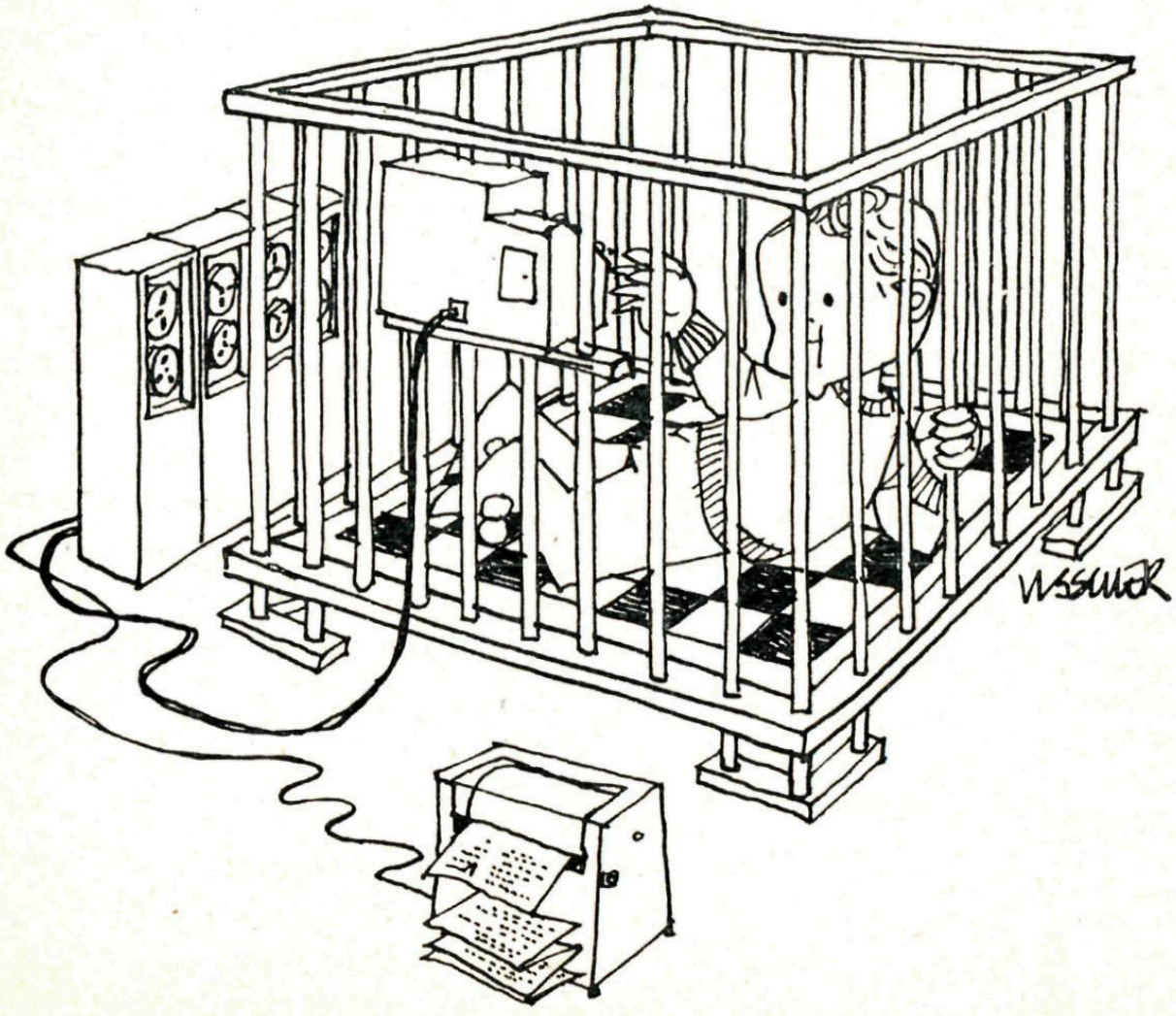
jaargang 8 no 5

1. Het rekentuig en jij

Hoeveel mensen horen geregeld radiomuziek zonder een snars verstand van radiotechniek te hebben? Hoeveel mensen rijden geregeld auto zonder te weten wat er onder de motorkap gebeurt?
Ja, de radio en de auto zijn verworvenheden van onze cultuur geworden. Het zijn gebruiksartikelen die ter beschikking van iedereen staan. En het kost ons moeite ons voor te stellen dat dat vroeger anders was.
Hoeveel mensen maken geregeld gebruik van een computer zonder kennis van computertechniek te bezitten? Deze vraag is heel anders dan de beide voorgaande; het stellen ervan heeft niet het karakter van het intrappen van een open deur. Om de computermensen van vandaag hangt nog de geheimzinnigheid van het pioniertijdperk. Zij zijn zeer gespecialiseerde technici met uitzonderlijke kennis en vaardigheden.
Anderzijds is echter duidelijk te zien in welke richting de toekomstige ontwikkeling zal gaan. En iedereen weet dat over een tiental jaren of hoogstens over een paar tientallen jaren de computer net zo'n verworvenheid van de cultuur zal zijn als de radio en de auto dat nu al zijn. De jongelui van vandaag zullen de volwassenen worden die het rekentuig hanteren zonder precies te weten wat er achter het toetsenbord gebeurt.
Op die toekomst is deze aflevering van Pythagoras gericht. Wij bespreken de computer zonder in te gaan op technische problemen. Wij stellen ons geheel op het standpunt van de gebruiker, die er geen belang bij heeft te weten hoe het allemaal werkt. Voor hem is slechts één vraag belangrijk: wat en hoe moet ik doen om het rekentuig zo te laten functionneren, dat het mijn wensen vervult?

Met enige aarzeling spraken wij zojuist over het vervullen van wensen. Die woorden passen namelijk akelig goed in de romantische sfeer, waarmee de computer tegenwoordig nog veel te vaak benaderd wordt. En die romantiek willen wij nu juist zo graag bestrijden.
De computer is geen toverapparaat dat wensen vervult langs boven-

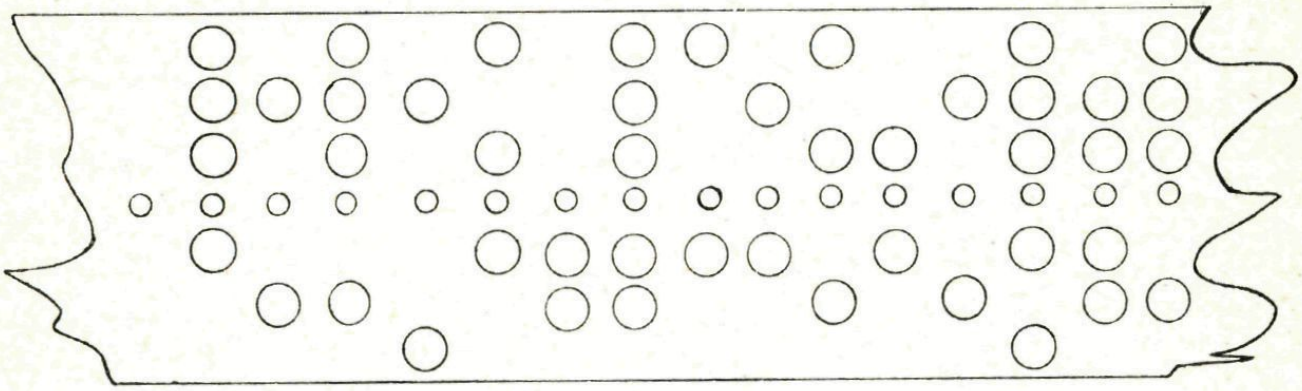
natuurlijke weg. Het is geen „ding" dat zich aan het ontwikkelen is tot een „wezen" met een eigen bestaan, dat mettertijd het mensdom zal gaan beheersen en leiden. Het is geen superbrein met een onpeilbare intelligentie, waar wij met diep ontzag tegen op moeten zien.
Nee, een computer is een mechanisme waarvan de vermogens door zijn constructie gegeven en beperkt zijn. En dat dode mechanisme kan binnen de grenzen van die vermogens bepaalde opdrachten uitvoeren, die door de opdrachtgever tot in de finesses gedetailleerd zijn uitgewerkt. Eigen initiatieven worden daarbij volstrekt niet ontwikkeld, net zo min als dat bij een radiotoestel of een auto het geval is.
Een auto, die zijn bestuurder naar Parijs brengt, hoort in een sprookje thuis. Die chauffeur komt alleen dan in Parijs terecht als hij het zelf wil en als hij bovendien de weg nog kent ook. Hij moet er bij wijze van spreken eerst naar toe kunnen lopen voor hij er naar toe kan rijden. En zo is het met de computer net zo. Zijn baas kan hem alleen die berekeningen uit laten voeren, die hij zelf met een potloodje en een papiertje ook maken kan. De computerslaaf doet het alleen vlugger dan de baas, dat is alles.



2. Toch wat techniek vooraf

Het is voor ons doel voldoende wanneer we aan een computer de volgende delen onderscheiden:

a. het invoerorgaan
b. het geheugen annex besturingsorgaan
c. het rekenorgaan
d. het uitvoerorgaan



Via het invoerorgaan (in onze tijd van Engelse ziekte meestal „de input" genaamd) wordt aan de computer het programma meegedeeld. Dit bevat gedetailleerde opdrachten voor alle bewerkingen, die de computer moet uitvoeren, en ook strenge regels voor de volgorde waarin die bewerkingen aan de beurt moeten komen. Ook komen via de input de getalgegevens de computer binnen, waarop de in het programma genoemde bewerkingen moeten worden uitgevoerd. In principe gebeurt dit alles op dezelfde manier, waarop ook de door een bandrecorder te produceren muziek wordt meegedeeld aan de apparatuur. Alleen worden de opdrachten van het programma en de getallen niet in een magnetische code op een band aangebracht, maar in een gaatjescode.

Er is echter een essentieel verschil met de genoemde bandrecorder. Bij de laatste is het aantal te produceren tonen gelijk aan het aantal verstrekte opdrachten op de band. En bovendien kunnen die tonen in het gezapige tempo van bijvoorbeeld een driekwartsmaat geproduceerd worden. Daarom kan bij de bandrecorder de productie van geluid gelijk op lopen met het verstrekken van opdrachten. Bij de computer is het tempo waarin de opdrachten door het rekenorgaan worden uitgevoerd veel en veel hoger dan het tempo waarin de opdrachten door het invoerorgaan kunnen worden opgenomen. Het rekenen zelf gebeurt

electronisch, het invoeren echter gebeurt slechts mechanisch. Om het optreden van wachttijden te voorkomen wordt het programma daarom tijdelijk opgeborgen in het geheugen van de computer. En dat geheugen is in staat om onder invloed van het besturingsorgaan het rekenorgaan de gegevens te verschaffen in het tempo waarin deze verwerkt kunnen worden. De getalgegevens worden soms wel en soms niet in het geheugen opgeslagen. Tenslotte dient het geheugen ook nog voor het tijdelijk bewaren van de tussenresultaten van de berekeningen. Overigens is het grondprincipe van het geheugen wèl hetzelfde als dat van de bandrecorder: wat er bewaard moet worden, wordt in een magnetische code omgezet en opgeslagen.

Het uitvoerorgaan is evenals het invoerorgaan weer een mechanisch onderdeel van de computer. Stel het je maar voor als een ultrasnelle typemachine, die de eindresultaten van het werk ter kennis van de opdrachtgever brengt. En dat is dan alles wat we van de computertechniek zullen bespreken.

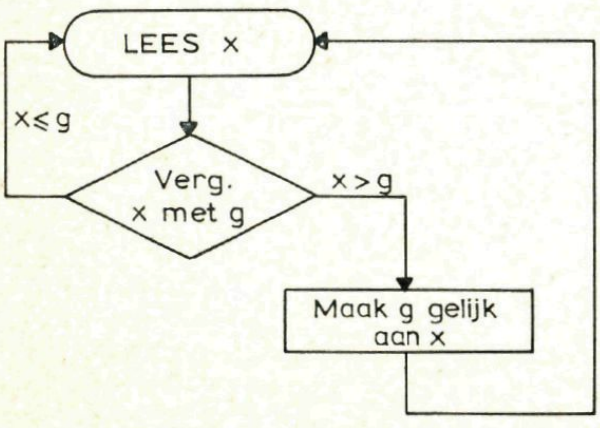
3. Een voorbeeld

Na het droogzwemmen van de vorige paragrafen komt er nu een uitgewerkt voorbeeld aan de beurt, dat de gepresenteerde gedachten toelicht.

Neem aan dat we een getallenband bezitten, waarop een groot aantal positieve getallen staat. We weten niet precies hoeveel getallen er op die band staan; het zijn er zeker meer dan 500 en minder dan 1000. Om de een of andere reden willen we precies weten hoeveel het er zijn en bovendien willen we het grootste en het kleinste getal van die band kennen. Hoe laten we onze computer te werk gaan?

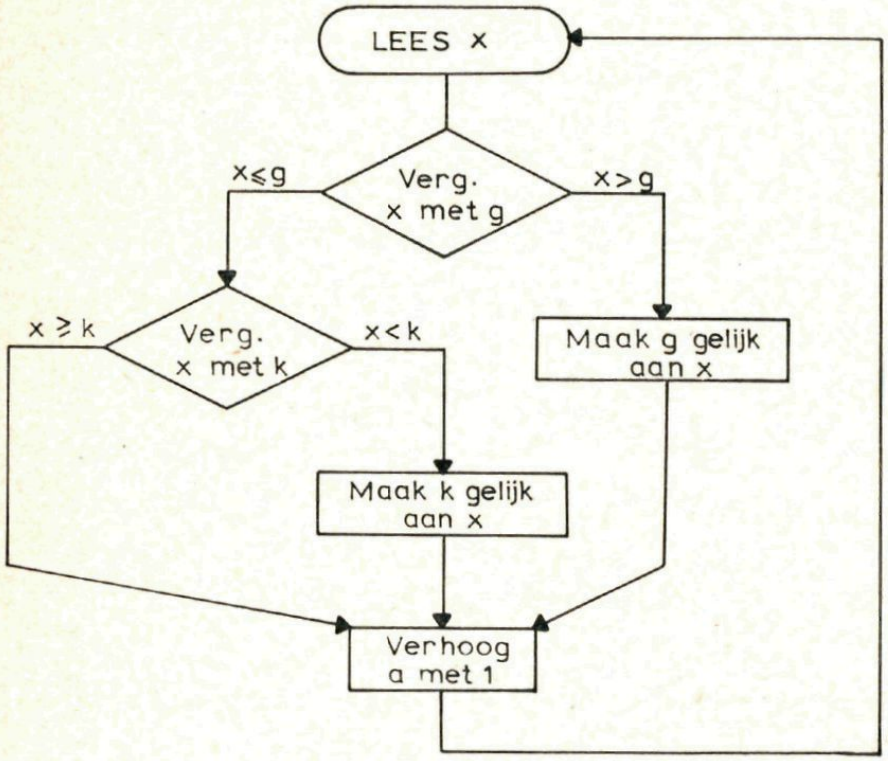
Om die vraag te kunnen beantwoorden moeten we eerst precies weten hoe we dat karwei zelf op zouden knappen. Wel, natuurlijk zouden we de band doorlezen. En al lezende zouden we drie dingen moeten doen: tellen, bijhouden wat het grootste tot nu toe gelezen getal is, bijhouden wat het kleinste tot nu toe gelezen getal is.

Dit plan is echter nog veel te vaag voor onze domme slaaf, de computer. Als we het hem zo commanderen dan kan hij het niet uitvoeren. Wat voor betekenis moet er bijvoorbeeld gehecht worden aan het werkwoord „bijhouden"? Goed, we moeten dus heel precies gaan detailleren. Stel je voor dat we bij ons werk een blaadje papier gebruiken, waar het grootste tot nu toe gelezen getal op staat geschreven. Lezen we een volgend getal van de band, dan vergelijken we dat met het getal op dat blaadje papier. Blijkt het gelezen getal groter te zijn dan het getal op het blaadje, dan strepen we het getal op het blaadje door en schrijven er het gelezen getal voor in de plaats, waarna we weer een volgend getal van de band onder handen nemen. Blijkt het gelezen getal kleiner te zijn dan het getal op het blaadje of gelijk daaraan, dan nemen we meteen een volgend getal van de band onder handen.



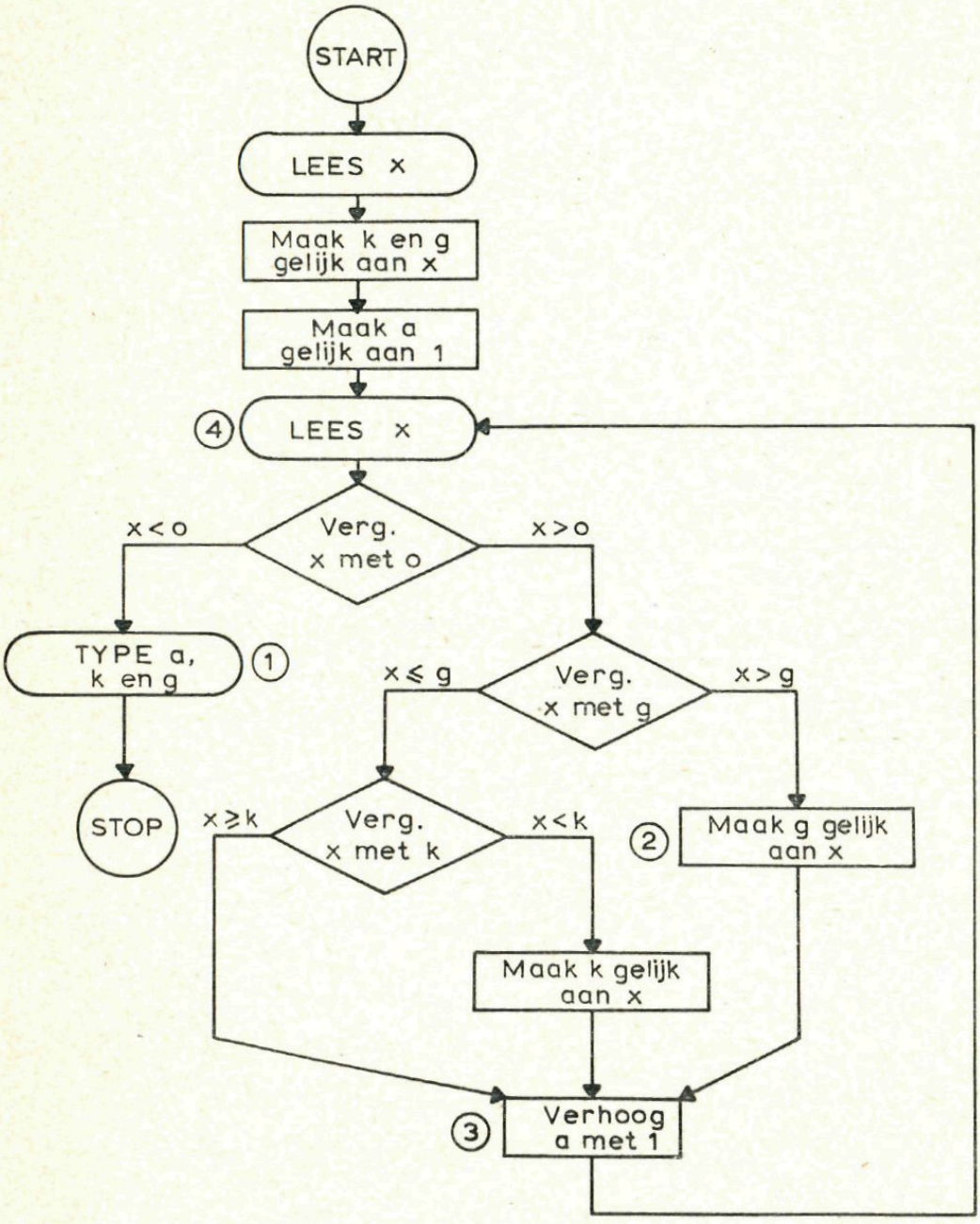
Beter dan uit deze vele woorden blijkt de te volgen handelwijze uit het

diagram op pag. 101, waarin x betekent het volgende getal van de band en g betekent het grootste tot nu toe gelezen getal.
Ziezo, dat is verfijnd genoeg. Maar . . . er moet meer gedaan worden dan alleen maar het grootste gelezen getal bijhouden. We moeten ook het kleinste tot nu toe gelezen getal (voortaan k te noemen) bijhouden en het aantal tot nu toe gelezen getallen (voortaan a te noemen) bijhouden. Dat leidt dan tot een wat uitgebreider diagram:



Tussen twee haakjes, zo'n diagram heet een blokschema. En dit blokschema is al bijna volmaakt. Het enige wat er nog aan mankeert is dat het geen kop en geen staart heeft. Het maakt wel duidelijk hoe we moeten doen als we eenmaal bezig zijn, maar vertelt niet hoe we moeten starten en stoppen.
De start is niet zo moeilijk. Zodra we het eerste getal van de band gelezen hebben, geven we zowel aan g als aan k de waarde van het zojuist gelezen getal en de waarde van a maken we gelijk aan 1.
Het stoppen is echter een neteliger zaak. Je zou kunnen denken: als er niks meer te lezen valt op de band, dan zijn we klaar en houden we op. Goed, dat geldt dan voor jou en voor ons. Maar onze slaaf de computer is een superezel. En hij blijft alsmaar doorgaan met het leesmechanisme te verstellen ook als er niets meer te lezen valt. En zelfs stoppen doet hij niet uit zichzelf.
We verzinnen dus een list om hem te laten ophouden met zinloze lees-

pogingen. Die list bestaat uit het aanbrengen van nog één extra getal op de band en wel een negatief getal (alle andere getallen op de band zijn positief, dat weten we van te voren). En kijk nu het definitieve blokschema maar eens aan om te zien hoe het verhaal afloopt:

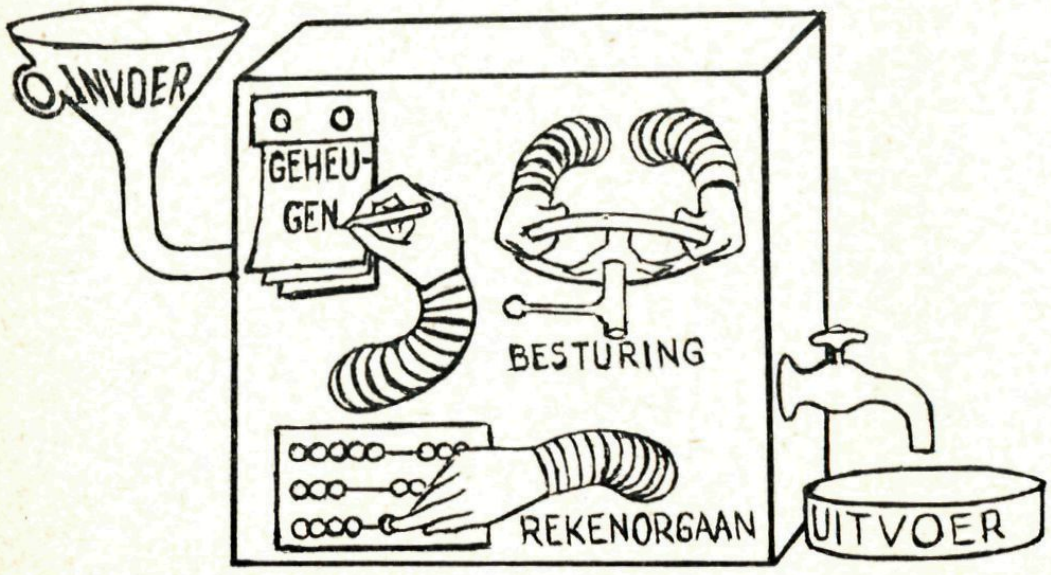


Het programma kan nu met behulp van dit blokschema gemakkelijk geschreven worden. Het zou er bijvoorbeeld zo uit kunnen zien:

```
START
LEES x
k = x
g = x
a = 1
```

```
4     LEES x
      ALS x<0 GADANNAAR 1
      ALS x>g GADANNAAR 2
      ALS x≥k GADANNAAR 3
      k = x
3     a = a + 1
      GANAAR 4
2     g = x
      GANAAR 3
1     TYPE a
      TYPE k
      TYPE g
      STOP
```

Dit programma wordt nu op een band aangebracht met behulp van daarvoor aanwezige apparatuur en door de computer gelezen. Dit betekent dat de programmaband aan het invoerorgaan wordt gehecht en dat de operateur op de knop LEES PROGRAMMA drukt en wacht tot de computer stopt (dit doet hij automatisch indien de LEES PROGRAMMA-knop is ingedrukt en er STOP wordt gelezen).
Daarna wordt de lange getallenband aan de input gehecht en drukt de operateur op de START-knop. Meteen begint de getallenband met grote snelheid door de computer heen te lopen. Zodra hij er door is begint meteen ook de typemachine van het uitvoerorgaan te ratelen en er verschijnen drie getallen op het papier: de waarden van a, k en g. Hoe lang het hele karwei geduurd heeft, hangt natuurlijk af van de lengte van de getallenband en van de soort computer. Maar het is zeker een werk van seconden.



Het is eigenlijk allemaal zo eenvoudig . . .

De ultrasnelle typemachine

4. Het programma

We zullen het programma uit de vorige paragraaf nog eens even scherp moeten bekijken. Het eerste opvallende feit is natuurlijk dat de computer Nederlands blijkt te verstaan. Nou is dat natuurlijk maar larie. In feite moeten de opdrachten aan de computer de vorm hebben van, laten we maar zeggen, bepaalde patronen van stroomstootjes. Deze machinetaal moest vroeger door de gebruikers van de computer braaf geleerd worden. Tegenwoordig kan een programma geschreven worden in een van de speciaal voor het gemak van de gebruiker ontwikkelde programmeertalen. En de fabrikant levert bij zijn rekentuig een vertaalprogramma, dat het in de programmeertaal geschreven programma omzet in een in machinetaal geschreven programma. Maar dit alles is een zeer technische zaak en zal ons dus verder niet bezig houden.

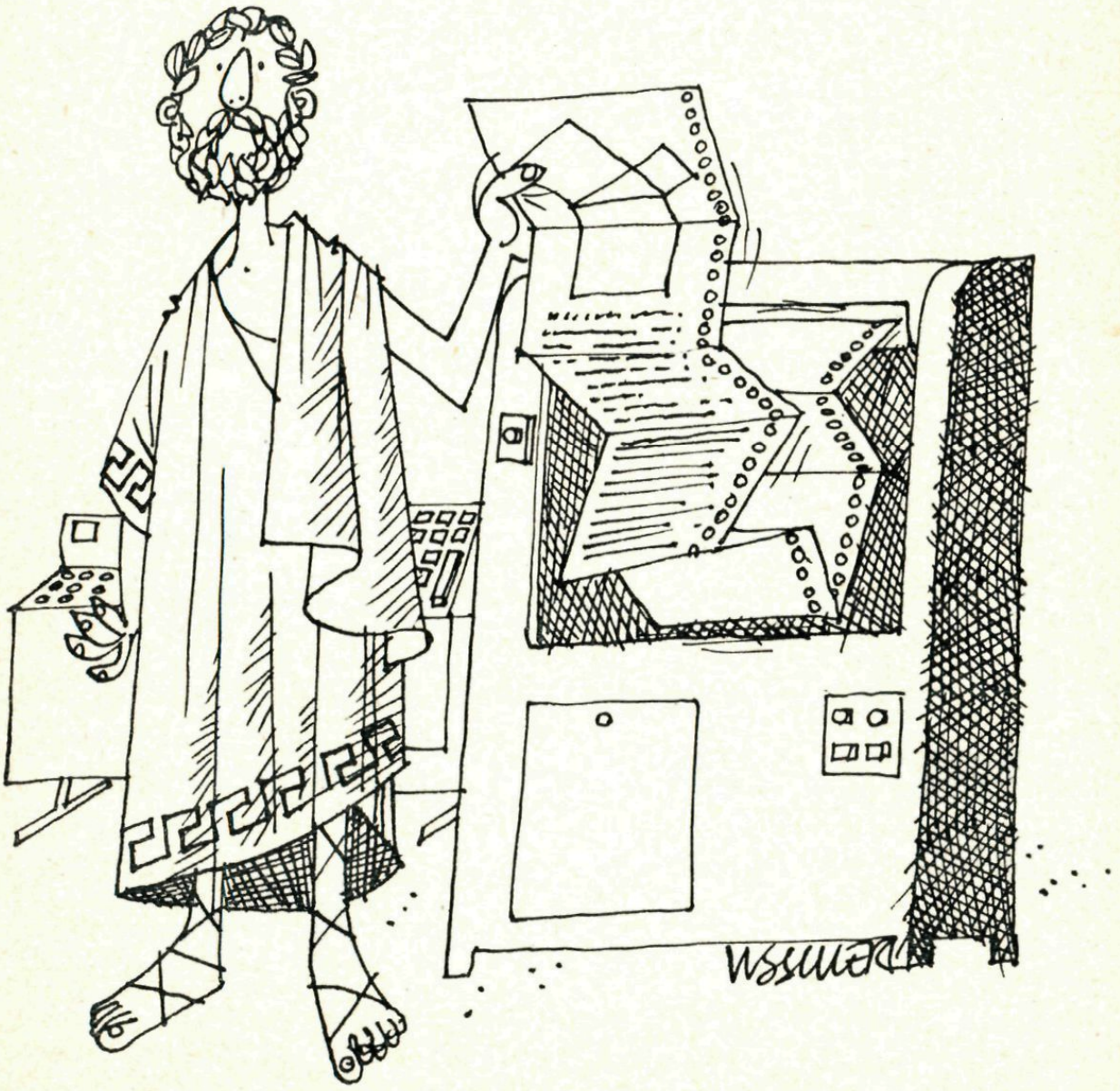
Iemand, die een programma schrijft in een van de gebruikelijke programmeertalen, moet zich strikt houden aan de vaste uitdrukkingen van die programmeertaal. In de denkbeeldige mini-taal die wij hier gebruiken zijn dat de met hoofdletters geschreven uitdrukkingen START, LEES, ALS . . . GADANNAAR . . ., GANAAR . . ., TYPE en STOP. De nummers 1, 2, 3, 4 worden gebruikt als uithangborden, als herkenningstekens voor die delen van een programma waarheen het besturingsorgaan moet overspringen. In het blokschema zijn die uithangborden ook geschreven. Normaal wordt het programma uitgevoerd in de volgorde, waarin de opdrachten geschreven zijn. Zo zal na de opdracht k = x de opdracht a = a + 1 uitgevoerd worden (maak de nieuwe waarde van a gelijk aan 1 meer dan de oude waarde van a). Maar daarna stuit het besturingsorgaan op de sprongopdracht GANAAR 4, dat de computer een nieuwe x doet lezen. Een voorwaardelijke sprongopdracht als bijvoorbeeld ALS x<0 GADANNAAR 1 heeft tot gevolg dat er in het rekenorgaan een vergelijking gemaakt wordt tussen de heersende waarde van x en het getal 0. Valt het resultaat van die vergelijking zo uit dat x kleiner dan 0 blijkt te zijn, dan laat het besturingsorgaan volgen de opdracht TYPE a. En als x niet kleiner dan 0 blijkt te zijn, dan wordt er geen sprong gemaakt. Dan komt dus normaal de opdracht aan de beurt die als volgende geschreven is. En dat is ALS x>g GADANNAAR 2.

Ziezo, nu moet je aan de hand van het blokschema het programma na enige studie kunnen begrijpen en nabouwen.

Als oefenmateriaal bieden we je de volgende opdrachten aan. Ze kunnen en moeten ook gemaakt worden met dezelfde vaste uitdrukkingen als in het voorbeeldprogramma. Maak eerst een blokschema voor je aan het schrijven van een programma begint. Uitwerkingen van deze opdrachten vind je achterin dit nummer.

Opdracht 1. Op een getallenband staan 365 getallen; het zijn verkeerstellingen op de achtereenvolgende dagen van een jaar. Men wil de pieken kennen in die telling; daar onder verstaat men die getallen van de band die zowel voorafgegaan als gevolgd worden door een kleiner getal. De computer moet zowel die pieken zelf als hun rangnummers typen.

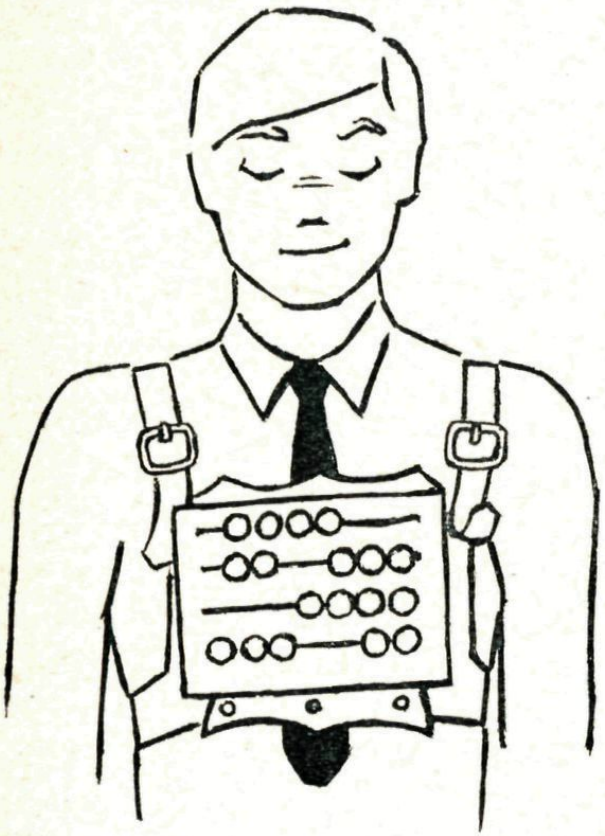
Opdracht 2. Op een getallenband staan 500 getallen in niet-dalende volgorde (dat wil zeggen dat elk volgend getal of gelijk aan zijn voorganger of groter dan zijn voorganger is). De getallen 138, 467 en 253 hadden ook op die band moeten staan, maar zijn per abuis niet opgenomen. Schrijf een programma dat de computer een niet-dalende rij van 503 getallen laat schrijven, bestaande uit de 500 getallen van de band en de drie vergeten getallen.



Pythagoras gaat met zijn tijd mee.

5. Genummerde variabelen

In het in paragraaf 3 uitgewerkte voorbeeld hadden we van het geheugen slechts vier verschillende plaatsjes nodig: een vast plaatsje waar de telkens weer veranderende waarde van x wordt opgeborgen, een vast plaatsje voor de variabele k, een plaatsje voor g en een plaatsje voor a. In vaktermen spreken we niet over „plaatsjes", maar over registers.

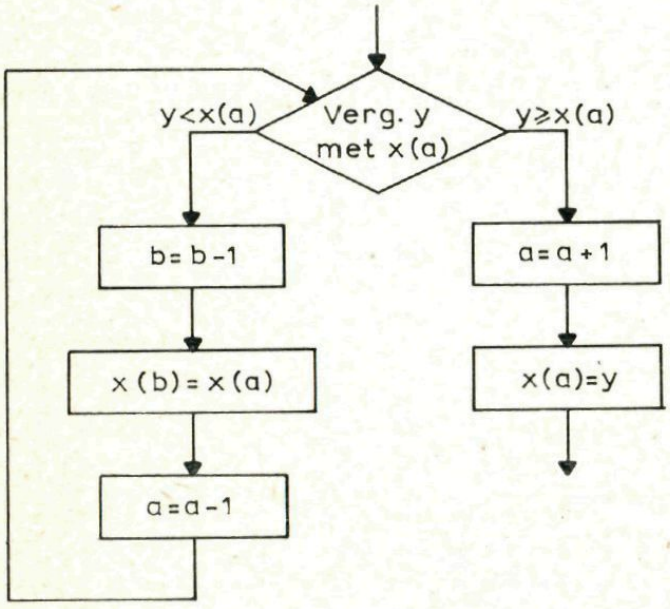


Jij en je rekentuigje

Als we eerst de gehele getallenband gelezen zouden hebben, dan zouden we evenveel geheugenregisters nodig gehad hebben als er getallen op de band stonden. En in het programma zouden we bovendien elk van die getallen een andere naam hebben moeten geven. Het eerste feit, het zo grondig in beslag nemen van geheugenruimte, kan een bezwaar zijn. Het tweede feit, al die vele namen, kan geen bezwaar zijn: in dat geval werken we met genummerde variabelen.
Van deze uitbreiding van ons mini-programmeertaaltje geven we een uitgewerkt voorbeeld.

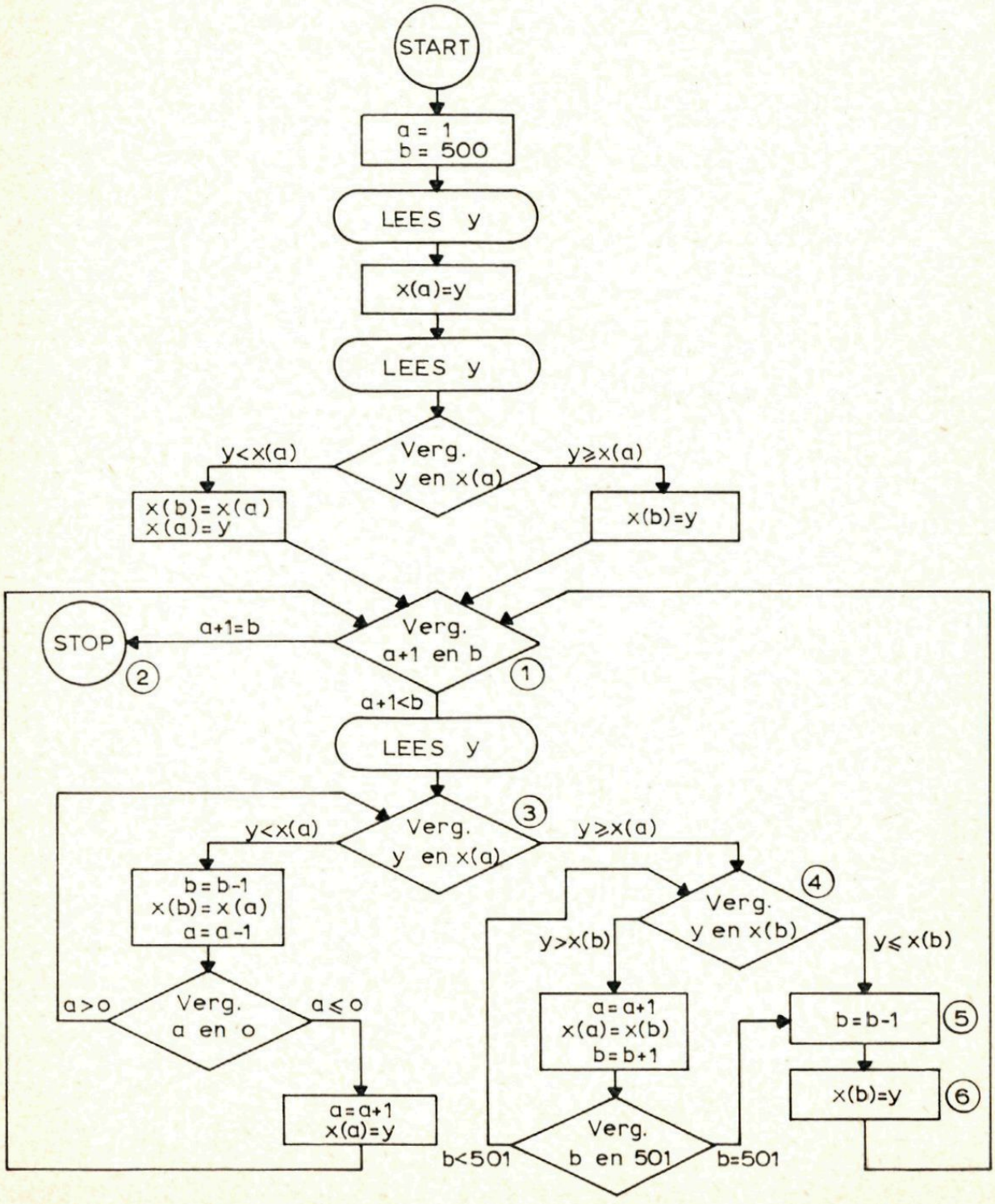
Op een getallenband staan 500 getallen, in de een of andere willekeurige volgorde. We willen de computer die getallen laten sorteren en ze in een niet-dalende rij laten uittypen.

Dit probleem kan zonder genummerde variabelen opgelost worden, maar dan moeten we de getallenband vijfhonderd keren achtereen door de computer jagen: eerst om het kleinste getal van de band te zoeken, dan voor het zoeken van het kleinste getal op één na, enzovoorts. Aanlokkelijk is dat beslist niet.
Daarom reserveren we in het geheugen een vijfhonderdtal registers, genummerd van 1 tot en met 500 en aangeduid als x(1) , x(2) , x(3) , . . . , x(500). Voorlopig zijn die registers leeg. Maar we stellen ons voor dat we in een tussenfase van het sorteerproces de registers met de nummers 1 tot en met a gevuld hebben met een niet-dalende rij en de registers met de nummers b tot en met 500 gevuld hebben met het vervolg van die rij. In totaal hebben we dan een niet-dalende rij van 501 + a — b getallen, die in twee brokstukken opgeborgen is.
Nu wordt er van de band een nieuw getal gelezen en dat wordt eerst met x(a) vergeleken (of beter met de inhoud van x(a) vergeleken). Laten we eens aannemen dat het zojuist gelezen getal y kleiner blijkt te zijn dan x(a), zodat y tussengevoegd zal moeten worden in het linker brokstuk. Uit het blokschema blijkt hoe we de juiste plaats van y dan opzoeken.



Bij oppervlakkige beschouwing lijkt dit een sluitend verhaal te zijn. Maar helaas is het dat niet. Er zitten zelfs wel twee lekken in: ten eerste komen we in moeilijkheden als bij de eerste vergelijking van y met x(a) al blijkt dat y niet kleiner is dan x(a); ten tweede hebben we ons niet gewapend tegen de mogelijkheid dat we wel eens alle getallen van links naar rechts zouden kunnen moeten schuiven voor y ge-

plaatst kan worden, in dat geval wordt a zelfs gelijk aan 0 volgens het blokschema en dat is kennelijk onzin. En hiermee is dan een aardig voorbeeld gedemonstreerd van de soort strubbelingen, die een beginner in dit vak kan ontmoeten.
Toch is de grondgedachte van het plan beslist niet ongezond. En de reparatie is dan ook niet moeilijk te vinden.
Ziehier een wel goed functionnerend en volledig blokschema en het bijbehorende programma.



```
      START
      a = 1
      b = 500
      LEES y
      x(a) = y
      LEES y
      ALS y≥x(a) GADANNAAR 6
      x(b) = x(a)
      x(a) = y
1     ALS a + 1 = b GADANNAAR 2
      LEES y
3     ALS y≥x(a) GADANNAAR 4
      b = b — 1
      x(b) = x(a)
      a = a — 1
      ALS a>0 GADANNAAR 3
      a = a + 1
      x(a) = y
      GANAAR 1
4     ALS y≥x(b) GADANNAAR 5
      a = a + 1
      x(a) = x(b)
      b = b + 1
      ALS b<501 GADANNAAR 4
5     b = b — 1
6     x(b) = y
      GANAAR 1
2     STOP
```

Uit de voorgaande voorbeelden zou je de indruk gekregen kunnen hebben dat onze computer alleen maar kan sorteren. In het volgende voorbeeld zetten we hem daarom maar eens aan het rekenen, je kunt er wellicht ook zelf je voordeel mee doen.

Stel je voor dat je het getal moet berekenen, waarin de vijfterm $x^4 - 2x^3 + 8x^2 - 17x + 7$ *overgaat bij substitutie van 1,7 voor x. Zou je dan de tweede, derde en vierde macht van 1,7 eerst gaan becijferen?*

Wij niet!

Natuurlijk moet elk van de vier factoren x van de term x^4 door 1,7 vervangen worden. Maar het is niet noodzakelijk om die vervanging

bij elk van die vier x'en tegelijk te doen. Begin maar eens met een enkele x te vervangen door 1,7. De term x^4 gaat dan over in $1{,}7\ x^3$. Nu kan hij samengenomen worden met de tweede term, met $-2x^3$ dus, en dat leidt dan tot $-0{,}3\ x^3$. Substitueer voor een van de drie hierin voorkomende x'en door 1,7 en dan komt er $-0{,}51\ x^2$. Samen met $8\ x^2$ levert dat $7{,}49\ x^2$. Door weer opnieuw een enkele x op te ruimen ontstaat hieruit 12,733 x en samen met —17 x wordt dat —4,267 x. Nu verdwijnt de laatste x en daarbij gaat —4,267 x over in —7,2539, dat samen met de slotterm +7 het gevraagde getal —0,2539 oplevert. Op deze manier gaat de berekening, zelfs met potlood en papier, veel sneller dan op de „ouderwetse" manier. Ook bij gebruik van een computer is dat een voordeel, tenminste wanneer je hem dat karweitje vaak wilt laten uitvoeren. Je zult dat begrijpen wanneer je weet dat een computer in het gebruik enkele tientallen guldens per minuut kost. Daar komt dan nog bij dat het programma voor dit domme rekenwerk zo eenvoudig geschreven kan worden bij het hanteren van de aangeduide bekorting, mits . . . de coëfficienten van de veelterm als genummerde variabelen behandeld werden. Op school noem je de coëfficienten in het bouwschema van kwadratische drietermen gewoonlijk a, b en c. Je spreekt herhaaldelijk over $ax^2 + bx + c$. Je zou die coëfficienten echter ook wel c(1), c(2) en c(3) kunnen noemen en dus spreken over $c(1)x^2 + c(2)x + c(3)$. En zo gaat het bouwschema van een vijfterm van de vierde graad er dan als volgt uitzien:

$$c(1)x^4 + c(2)x^3 + c(3)x^2 + c(4)x + c(5)$$

Opdracht 3. Je hebt de coëfficienten c(1) tot en met c(13) van een veelterm van de twaalfde graad in het geheugen van de computer gebracht. Nu wil je de computer een tabel laten produceren van functiewaarden van die veelterm. De veranderlijke x in de veelterm moet eerst door 1 vervangen worden, dan door 1,01, dan door 1,02 enzovoorts. De laatste functiewaarde in de tabel moet zijn die, welke door substitutie van 2 voor x ontstaat (er komen dus 101 functiewaarden in de tabel). Schrijf het programma.

6. Worteltrekken

De meeste leerlingen moeten op school zogenaamde „staartwortel-trekkingen" leren maken en vinden dat maar een matig genoegen. Tot hun troost kunnen we zeggen dat een programmeur er evenmin schik in zou hebben als hij zijn computer datzelfde kunstje moest laten uitvoeren. Onmogelijk is dat beslist niet, maar een pittige opgave zou je het zeker kunnen noemen.
Er bestaat namelijk een gemakkelijker uit te voeren en gemakkelijker te begrijpen methode, die zowel voor computers als voor leerlingen eenvoudig in het gebruik is. Bij die methode ga je uit van een benaderde waarde van de gewenste wortel. Uit die benadering leid je een betere benadering af, daaruit een nog betere benadering en die wordt dan gebruikt voor het maken van een nog veel betere benadering, en zo voorts en zo verder.

Het positieve getal, waar de wortel uit getrokken moet worden, noemen we a. Het eveneens positieve getal, dat we als benadering voor $\sqrt{a}$ gebruiken en dat het uitgangspunt van onze beschouwingen is, noemen we x. Om de gedachten te bepalen nemen we aan dat x iets groter is dan $\sqrt{a}$. We kunnen dus stellen dat

$$\sqrt{a} = x - \delta$$

waarin δ een (klein) positief getal is.
Hieruit volgt na kwadratering $a = x^2 - 2x\delta + \delta^2$. Is nu x inderdaad een aardige benadering voor $\sqrt{a}$, is dus δ inderdaad klein, dan is δ^2 kleiner dan klein. De bewering

$$a = x^2 - 2x\delta$$

is zeker fout, maar kan niet zo erg fout zijn. Met andere woorden:

$$\delta = \frac{x^2 - a}{2x}$$

levert een goede schatting van δ op. We verwachten dus dat een betere benadering van $\sqrt{a}$ verkregen wordt door x te vervangen door

$$x - \frac{x^2 - a}{2x} = \frac{x^2 + a}{2x}$$

Als bijvoorbeeld $a = 15$, dan is $x = 4$ een bruikbare benadering voor $\sqrt{a}$. Volgens het voorgaande zou dan een (nog) betere benadering van $\sqrt{15}$ moeten zijn het getal

$$\frac{16 + 15}{8} = 3{,}875.$$

Dit is inderdaad het geval. Uitgaande hiervan vinden we weer een betere benadering door te nemen:

$$\frac{(3{,}875)^2 + 15}{2 \times 3{,}875} = 3{,}8729838.$$

Dit is al heel erg nauwkeurig, want in zeven decimalen afgerond is $\sqrt{15}$ gelijk aan 3,8729833.
De achtereenvolgende benaderingen van $\sqrt{a}$ noemt men
$x_1, x_2, x_3 \ldots\ldots x_n, x_{n+1} \ldots$
waarbij

$$x_{n+1} = \frac{(x_n)^2 + a}{2x_n} = \left[x_n + \frac{a}{x_n} \right] : 2 \text{ voor elke n.}$$

Dit soort benaderingen kunnen we ook door de computer laten uitvoeren.
Voor computertoepassing mogelijk is moeten er echter nog twee kleine probleempjes worden opgelost: Hoe brengen we het proces op gang en hoe stoppen we het? We kunnen niet naar een geschikte begin-benadering laten raden, zoals we zelf gedaan hebben zojuist. We nemen daarom als x_1 het gemiddelde van a en 1, dus

$$x_1 = \frac{a + 1}{2}$$

Het stoppen gaat zo: Stel we willen een benadering waarvan 10 decimalen nauwkeurig zijn, we laten dan ophouden zodra bij twee opvolgende stappen geen verschil meer optreedt in de eerste 11 decimalen.
Dit wordt het programma:

```
      START
      x = (a + 1) : 2
2     y = a : x
      ALS x — y<10^-11 GADANNAAR 1
      x = (x + y) : 2
      GANAAR 2
1     TYPE x
      TYPE y
      STOP
```

Voor $\sqrt{15}$ vind je op deze manier:
$x_1 = 8$
$x_2 = 4{,}9375$
$x_3 = 3{,}9877$

$x_4 = 3{,}874708$
$x_5 = 3{,}87298388$
Hoewel de eerste benadering erg ver van de waarheid afligt, kom je dus toch snel tot een goede benadering, wat niet wegneemt dat je voor eigen gebruik beter wat dichterbij kunt beginnen.

°°Voor de liefhebbers gaan we nog wat dieper op de zaak in.
Een redenering die ook tot de gebruikte benadering leidt is de volgende. Als x groter dan $\sqrt{a}$ is, is $\frac{a}{x}$ kleiner dan $\sqrt{a}$. We kunnen dus stellen dat $\sqrt{a}$ tussen x en $\frac{a}{x}$ ligt. En, hoewel de waarheid niet precies in het midden behoeft te liggen, er is toch reden om aan te nemen dat het gemiddelde van x en $\frac{a}{x}$ een betere benadering voor $\sqrt{a}$ is dan x was. Dat gemiddelde nu is gelijk aan de zojuist gevonden waarde, want $\left(x + \frac{a}{x}\right) : 2 = (x^2 + a)/2x$.

Dit alles verschaft ons echter nog altijd niet de *zekerheid* dat onze $\left(x + \frac{a}{x}\right) : 2$ onder alle omstandigheden, dus voor elke waarde van a, een betere benadering is dan x al was. Bewijzen hebben we nodig, geen vermoedens. Ten eerste tonen we aan dat de nieuwe benadering, evenals x zelf, groter dan $\sqrt{a}$ is. Dit blijkt uit het positief zijn van

$$\frac{x^2 + a}{2x} - \sqrt{a} = \frac{x^2 - 2x\sqrt{a} + a}{2x} = \frac{(x - \sqrt{a})^2}{2x}$$

Ten tweede tonen we aan dat de nieuwe benadering kleiner is dan x. Dit blijkt uit het feit dat hij het gemiddelde is van x (die groter is dan $\sqrt{a}$) en $\frac{a}{x}$ (die kleiner is dan $\sqrt{a}$ en dus kleiner is dan x).
Ziezo, die twee zaken samen leveren het gewenste bewijs dat we er altijd beter op worden bij het vervangen van x door $\left(x + \frac{a}{x}\right) : 2$.
Ook houdt dit in, dat we er telkens weer beter op worden wanneer we het proces herhalen. Bij de constructie van een rij getallen

$$x_1 , x_2 , x_3 , x_4 , x_5 \ldots$$

volgens het voorschrift

$$x_{n+1} = \left[x_n + \frac{a}{x_n}\right] : 2 \text{ voor elke n}$$

zal dus gelden

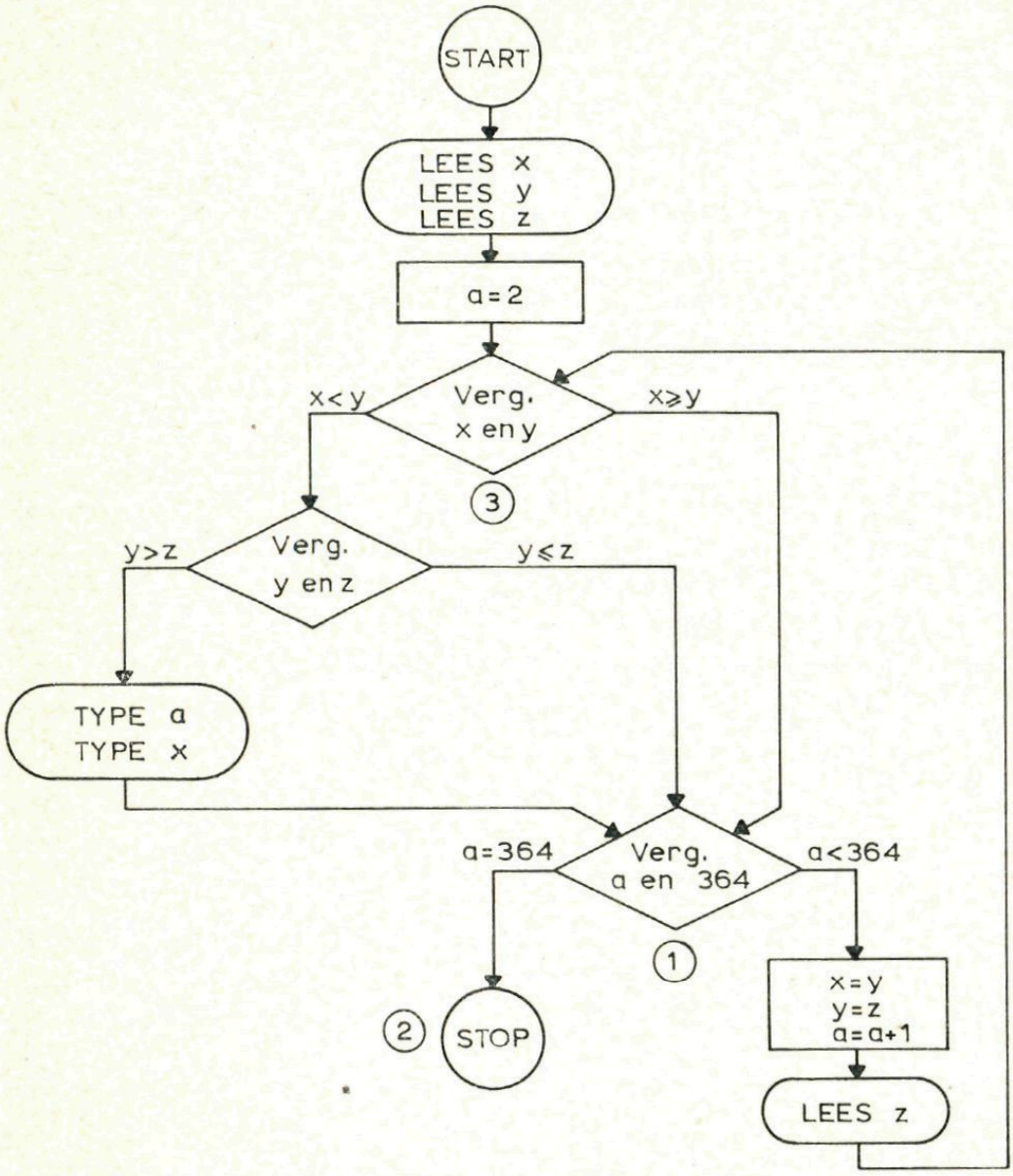
$$\sqrt{a} \leq x_{n+1} \leq x_n \text{ voor elke n,}$$

indien tenminste geldt dat x(1) groter is dan $\sqrt{a}$.

Wij drijven nu de zaken op de spits met de vinnige bewering dat dit niet de garantie geeft dat x(n) bij onbeperkt stijgende n ook onbeperkt dicht bij $\sqrt{a}$ komt. Die zekerheid is wel te vinden in het boekje INDUCTIE EN ITERATIE, door Prof. dr. H. J. A. Duparc, dat als deel 2 van de TORUS-REEKS bij de uitgever van Pythagoras verschenen is.

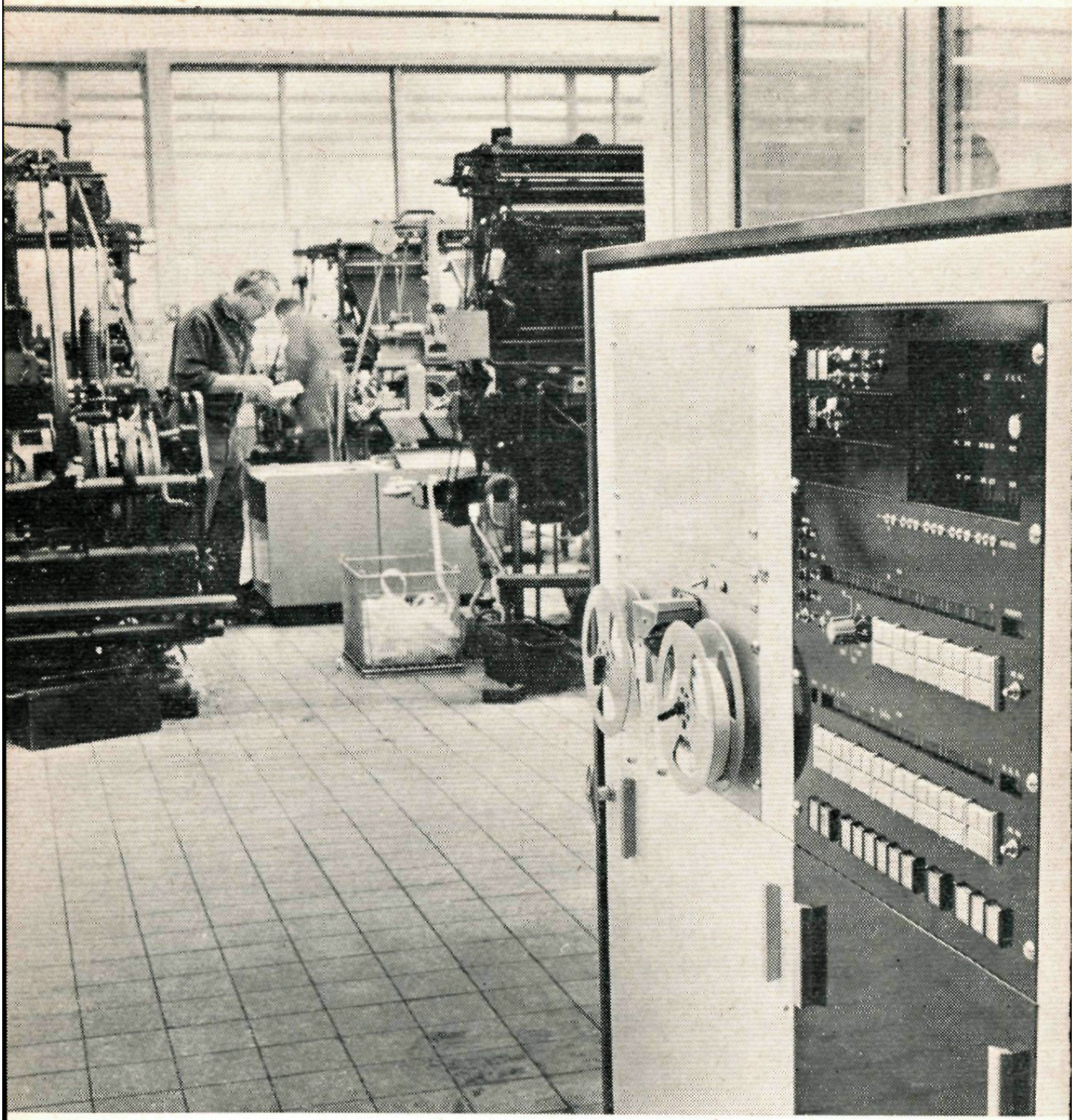
7. Uitwerking opdrachten

Het blokschema van opdracht 1 zou er als volgt uit kunnen zien:



Het bijbehorende programma is dan:

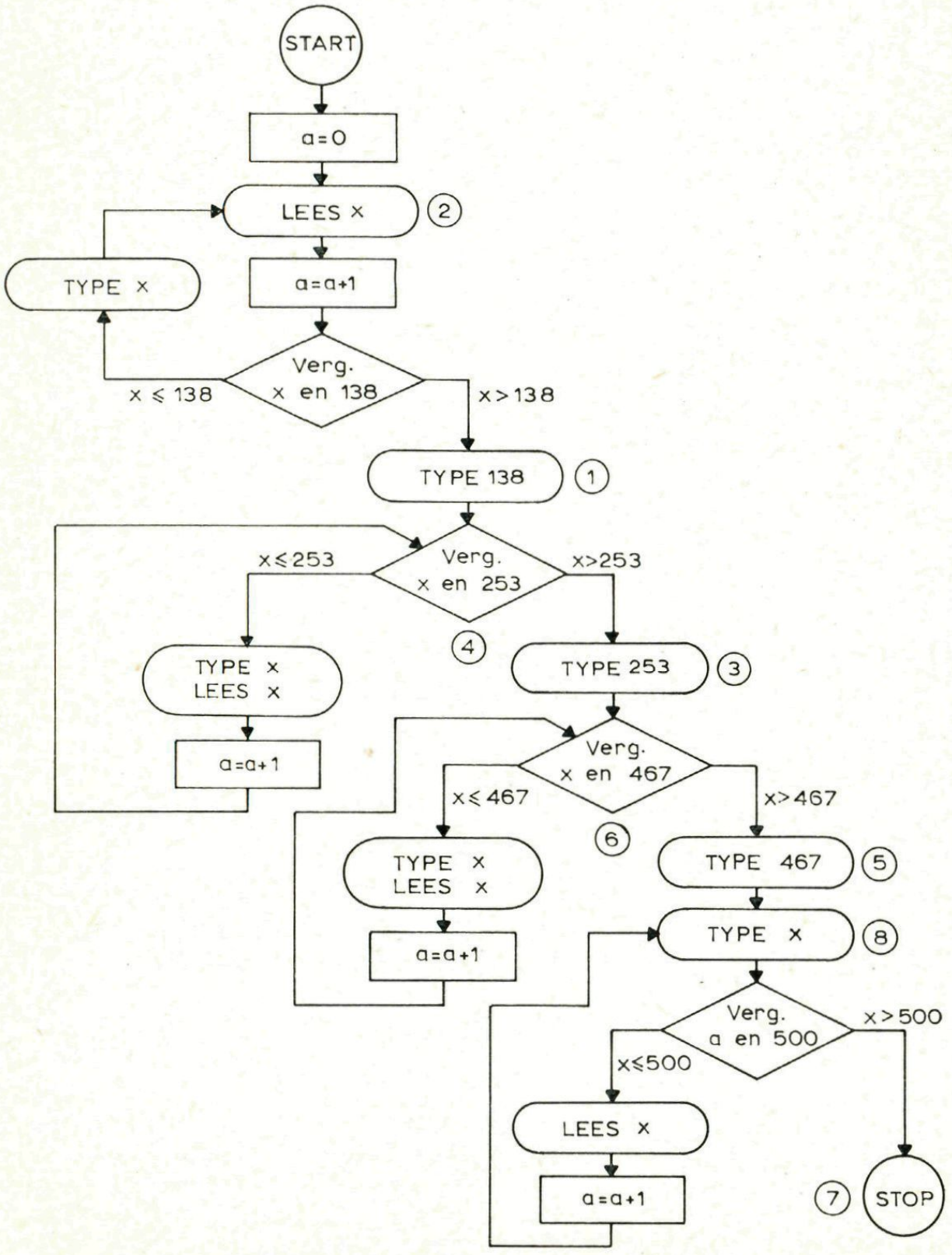
```
      START
      LEES x
      LEES y
      LEESz
      a = 2
3     ALS x ≥y GADANNAAR 1
      ALS y≤ z GADANNAAR 1
      TYPE a
      TYPE y
```



Rekenmachine in (het) bedrijf

```
1     ALS a = 364 GADANNAAR 2
      x = y
      y = z
      a = a + 1
      LEES z
      GANAAR 3
2     STOP
```

Ziehier het blokschema van opdracht 2:



Het programma van opdracht 2:

```
      START
      a = 0
2     LEES x
      a = a + 1
      ALS x>138 GADANNAAR 1
      TYPE x
      GANAAR 2
1     TYPE 138
4     ALS x>253 GADANNAAR 3
      TYPE x
      LEES x
      a = a + 1
      GANAAR 4
3     TYPE 253
6     ALS x>467 GADANNAAR 5
      TYPE x
      LEES x
      a = a + 1
      GANAAR 6
5     TYPE 467
8     TYPE x
      ALS a =500 GADANNAAR 7
      LEES x
      a = a + 1
      GANAAR 8
7     STOP
```

Het programma van opdracht 3:

```
      START
      x = 1
1     y = 0
      n = 1
2     y = xy + c(n)
      n = n + 1
      ALS n≤ 13 GADANNAAR 2
      TYPE x
      TYPE y
      x = x + 0,01
      ALS x≤ 2 GADANNAAR 1
      STOP
```

De werking van het geheugen

Zakelijke mededelingen

Dit tijdschrift wordt uitgegeven onder auspiciën van de Nederlandse Onderwijs Commissie van het Wiskundig Genootschap.

REDACTIE

H. J. ENGELS, Muggenbroekerlaan 41, Roermond.
BRUNO ERNST, Daniël Marotstraat 184, Breda.
Drs. A. B. OOSTEN, Turftorenstraat 11A, Groningen.
A. F. VAN TOOREN, Nachtegaalplein 10, Den Haag.
G. A. VONK, Fahrenheitstraat 688, Den Haag.

Artikelen en problemen kunnen naar één van de redacteuren worden gestuurd.
Oplossingen van *Denkertjes* kunnen naar het eerste adres worden gezonden, oplossingen van andere problemen naar het vierde adres.
Vermeld bij alle inzendingen duidelijk naam, adres, school en leerjaar.

ABONNEMENTEN

Pythagoras verschijnt 6 maal per schooljaar.
Voor leerlingen van scholen, besteld via een der docenten, *f* 2,50 per jaargang. Voor anderen *f* 4,—.
Abonnementen kan men opgeven bij Wolters-Noordhoff N.V., Postbus 58, Groningen.
Het abonnementsgeld dient te worden gestort op girorekening 807707 van Wolters-Noordhoff.