Praxis

gebruikers-handleiding

Henk Penning en Lex Wolters

RUU-CS-82-14
December 1982

Rijksuniversiteit Utrecht

Vakgroep informatica

Princetonplein 5
Postbus 80.002
3508 TA Utrecht
Telefoon 030-53 1454
The Netherlands

vakgroep informatica R.U. Utrecht

Praxis

gebruikers-handleiding

Henk Penning en Lex Wolters

Technical Report RUU-CS-82-14
December 1982

Vakgroep Informatica
Rijksuniversiteit Utrecht

PRAXIS is een systeem waarmee beginners in het computer-gebruik programma´s kunnen ontwikkelen op een CYBER 175 met NOS/BE. PRAXIS-gebruikers werken in de zelfde omgeving als "gewone" CYBER-gebruikers nl. INTERCOM. Ze hebben dus alle mogelijkheden van dien, maar hebben voor het gebruik van PRAXIS geen kennis over NOS/BE of INTERCOM nodig.

PRAXIS is een klein, beperkt toepasbaar systeem dat zeer simpel in het gebruik is. Het stelt een gebruiker precies in staat om een aantal, op zichzelf staande, computer-programma´s te maken. De "extra´s" zijn op het gebied van de back-up faciliteiten. De gebruiker is beschermd tegen zijn eigen fouten. Er is een goede bescherming tegen systeem-crashes.

Dit is de tweede versie van PRAXIS. Het belangrijkste verschil met de eerste versie is dat naast PASCAL nu ook ALGOL68 beschikbaar is geworden als programmeertaal.

Elke vorm van kritiek op PRAXIS is welkom. Het is de enige manier om het produkt beter te krijgen. We willen hier de deelnemers van de cursus "computers en informatie-verwerking 82" bedanken voor hun reakties op het werken met de eerste versie van PRAXIS.

We zoeken antwoorden op de volgende vragen :
- Wat moet er uit, wat moet er in, wat moet anders ?
- Zijn er zaken die moeilijk aan te leren of te onthouden zijn ?
- Zijn er dingen die je altijd fout doet ?
- Zijn HELP en DOC zodanig dat je de handleiding niet meer nodig hebt ?
- Hoe is de overstap van PRAXIS op NOS/BE ?
- Zitten er fouten in PRAXIS en de PRAXIS-dokumentatie ?

Antwoorden op bovenstaande vragen gaarne naar :

Henk Penning of Lex Wolters.
Vakgroep Informatica.
Princetonplein 5
Postbus 80.002
3508 TA UTRECHT/UITHOF

Telefoon 030-534408 of 030-531454.

INHOUD :::

1. uitgangs-situatie

We gaan er van uit dat je een of meer computer-programma's wilt maken, geheel of gedeeltelijk, via een terminal. Dat kan in praktikumverband zijn of op individuele basis.
We nemen aan dat je PASCAL of ALGOL68 kent en wilt gebruiken, maar dat je verder geen ervaring hebt met computers of terminals.

2. computer-gebruik

Als je werk gedaan wilt krijgen van een computer, dan zal je de computer opdrachten moeten geven. Dat geven van boodschappen aan een computer kan op twee manieren :

- je tikt de opdrachten in op ponskaarten die je de computer laat lezen door middel van een kaartlezer, die verbonden is met de computer. Je krijgt dan het verslag van je opdrachten op papier terug. Dit heet Batch-verwerking.
- je tikt de opdrachten in op een terminal, die verbonden is met de computer, en krijgt meteen een verslag. Dit heet interactief gebruik.

PRAXIS is een systeem om interactief gebruik gemakkelijk te maken.

3. terminal

Een terminal is een apparaat waarmee je met de computer praat. Het is een vrij dom ding dat slechts bestaat uit een toetsenbord en een scherm. De gebruiker typt boodschappen (commando's) in op het toetsenbord en de computer schrijft die boodschappen op het scherm om je te laten zien wat je intikt. Bovendien schrijft de computer het resultaat van de commando's op het scherm. Dat wat jij intikt op het toetsenbord verschijnt op het scherm in kleine letters; dat wat de computer terugmeldt verschijnt in HOOFD-LETTERS. (Verderop zullen we commando's die jij intikt in hoofdletters schrijven om ze te onderscheiden van de rest van de tekst.)
Met een cursor laat de computer zien waar het volgende character geschreven zal worden op het scherm. De cursor-positie wordt aangegeven met een knipperend lichtvlekje op het scherm.
Als je een typfout maakt, kan je de cursor terug laten lopen over de tekst, die je al hebt ingetypt, door middel van :

- een "backspace"-toets
- heeft je terminal geen "backspace"-toets, gebruik dan de <control>h (spreek uit "control ha"). Die krijg je door TEGELIJK de "control"-toets en de "h"-toets in te drukken. De "control"-toets vind je links op het toetsenbord in de buurt van de "case-shift"-toets.

Het tekstdeel waarover je terugloopt, "vergeet" de computer.

4. commando's

Wanneer de computer klaar is met het uitvoeren van een commando en staat te wachten op een volgend commando schrijft ie op het scherm :

```
        COMMAND-
of      ..
```

Zoiets heet een prompt.
Een commando is gewoon een regeltje tekst. Je geeft de computer opdracht het commando uit te voeren door het indrukken van de "return"-toets.
Nu gaat de computer aan het werk en voert het commando uit of meldt dat ie het niet kent of niet begrepen heeft. Het commando kan tot gevolg hebben dat er van alles op het scherm wordt geschreven. Als de computer klaar is en weer wat van je verwacht, schrijft ie weer :

```
        COMMAND-
of      ..
```

5. terminal-sessie

Een tijdje achter een terminal zitten, commando's geven en naar het resultaat kijken heet een terminal-sessie. Een terminal-sessie begint met een aantal speciale commando's die je toegang tot de computer verschaffen : de LOGIN-procedure. Is die afgewerkt, dan verschijnt COMMAND en kan je commando's gaan geven. Als je geen commando's meer wilt, geven beeindig je je sessie met het laatste commando : LOGOUT. De computer meldt dan wat de terminal-sessie heeft gekost en sluit de terminal af voor verdere commando's. Alleen met een LOGIN-procedure kan je de terminal weer aan de praat krijgen.

6. LOGIN-procedure

Als je de terminal-sessie wilt beginnen, ga je achter een terminal zitten en typt een letter gevolgd door <return> totdat de computer meldt:

```
PLEASE LOGIN
```

Nu moet je te weten komen wat de <user-name> en <pass-word> zijn die bij de terminal horen. Dit staat meestal ergens opgeplakt op de terminal. Meestal zijn het 2-letter combinaties: AH, H1, DU.
Nadat de computer meldt: PLEASE LOGIN typ jij

```
LOGIN,<user-name>,<pass-word>
```

Bijvoorbeeld: LOGIN,D4,EQ of: LOGIN,EP,DK
De computer antwoordt met:

```
ENTER ACCOUNT-STATEMENT
```

Nu moet je melden wie de rekening gaat betalen. Van de praktikumleiding of van je vakgroep-beheerder heb je een account-nummer gekregen en evt. een account-subnummer. Dit meld je aan de computer met

ACCOUNT,<account-nummer>,SN=<account-subnummer>

Bijvoorbeeld: ACCOUNT,UWIPRAK,SN=ABABA
of: ACCOUNT,UWIXYZ,SN=TOPSECRET
Als je geen typfouten maakt ben je nu "ingelogd" en meldt de computer : COMMAND-
Lukt het je niet om in te loggen, vraag dan iemand aan een andere terminal je even te helpen. Het is hen wel gelukt!

7. werken met PRAXIS

Als je ingelogd bent en de computer meldt COMMAND, kan je elk commando geven dat de computer kent. Dat zijn er een heleboel en je hebt er heel veel nodig om een programma draaiende te krijgen. Het zijn vaak rare commando's, die onbegrijpelijke boodschappen opleveren als je iets fout doet, en bijna even onbegrijpelijke boodschappen als je iets goed doet. Als je eerst tegen de computer zegt dat je PRAXIS wilt gebruiken, kent de computer een aantal speciale PRAXIS-commando's, die te maken hebben met het via een terminal maken van een programma.
Als je individueel werkt, kan je de computer melden dat je PRAXIS wilt gebruiken met de volgende twee commando's:

```
ATTACH,PRAXIS,ID=SOLO
LIBRARY,PRAXIS
```

Als je in praktikumverband met PRAXIS werkt moet je een speciale PRAXIS gebruiken die bij dat praktikum hoort.
Je moet dan de computer melden dat je PRAXIS wilt gebruiken met de commando's :

```
ATTACH,PRAXIS,ID=...
LIBRARY,PRAXIS
```

Bij ID=... vul je de code van je praktikum in.

```
ID=CI  voor computers en informatie-verwerking.
ID=CP  voor computers en programmeren.
ID=FA  voor fundamentele algoritmen.
ID=DS  voor data-structuren.
```

Begin hier dus altijd je terminal-sessie mee, totdat je genoeg van de computer weet om met gewone computer-commando's voor elkaar te krijgen wat je wilt. PRAXIS is een piepklein systeem met een beperkt toepassings-gebied. Wil je meer, dan moet je meer commando's kennen. PRAXIS is een systeem om de introductie met interactief-gebruik gemakkelijker te maken.

8. wat doet het PRAXIS-systeem?

Voor alle PRAXIS-gebruikers bewaart PRAXIS in de computer ruimte voor 6 teksten, de oplossingen van Som1 t/m Som5 en programma-input. Dit heet het PRAXIS-bestand van een gebruiker.
Er zijn commando's waarmee je

- nieuwe programma-tekst kunt invoeren
- oude programma-tekst kunt veranderen
- testdata kunt invoeren
- een oudere versie van een programma vervangt door een nieuwe
- een programma kunt laten controleren op taalfouten
- een programma kunt "runnen" met testdata
- programma-tekst en resultaten van een testrun kunt laten afdrukken of bekijken
- een copie van je programma kunt krijgen in de vorm van een pak ponskaarten
- programma-tekst in de vorm van ponskaarten kunt invoeren in het PRAXIS-bestand

9. PRAXIS-bestanden

Ieder PRAXIS-bestand heeft een eigen identificatie (ID). Dat is een naam van hooguit 7 letters of cijfers, beginnend met een letter. De ID's dienen om de verschillende PRAXIS-bestanden uit elkaar te kunnen houden.
Om te voorkomen dat iedereen die de ID van je PRAXIS-bestand kent er in kan snuffelen, zijn de PRAXIS-bestanden beschermd met een password (PW). Een PW bestaat ook weer uit hooguit 7 letters of cijfers, beginnend met een letter. Zonder ID en PW is toegang tot het PRAXIS-bestand onmogelijk.

Hoe je aan een PRAXIS-bestand komt, hangt er van af of je in praktikum-verband werkt of individueel.
Als je in praktikum-verband werkt, spreek je een ID en PW af met je praktikum-leider, die dan een PRAXIS-bestand voor je klaar zet met die ID en PW.
Als je individueel werkt, moet je zelf een PRAXIS-bestand klaarzetten. Dat kan met het PRAXIS-commando : MAKEMF. Nadat je de computer met de commando's :

```
ATTACH,PRAXIS,ID=SOLO
LIBRARY,PRAXIS
```

gemeld hebt dat je met PRAXIS wilt werken, moet je een PRAXIS-bestand maken met het commando :

```
MAKEMF,ID=...,PW=...,LAN=...
```

Bij ID en PW moet je de gewenste ID en PW invullen.
Bij LAN=... moet je opgeven of je in eerste instantie met LAN=ALGOL68 of met LAN=PASCAL wilt werken. Dit is later willekeurig vaak te veranderen.

10. START

Als je wilt werken met je PRAXIS-bestand, dan moet je ALTIJD eerst aan de computer gemeld hebben dat je het PRAXIS-systeem wilt gebruiken. Dit doe je door de reeds eerder genoemde commando's te geven:

```
ATTACH,PRAXIS,ID=....
LIBRARY,PRAXIS
```

Hiermee krijg je toegang tot het PRAXIS-systeem.

Daarna moet je met het START-commando aan PRAXIS de opdracht geven om je PRAXIS-bestand op te halen.
Dat gaat zo : START,ID=...,PW=...

bijvoorbeeld : START,ID=HENKP,PW=OLDSHOE
START,ID=LEXW,PW=WXEL

Na het START-commando staat je PRAXIS-bestand tot je beschikking en kan je er van alles mee doen.

Na het START-commando worden eventueel berichten van je praktikum-leiding of PRAXIS-beheerder op je scherm geschreven. Je kunt dit onderdrukken met met de SUP-parameter in het START-commando.
voorbeeld : START,ID=RLS,PW=TRESIS,SUP

Als je (voorlopig) klaar bent en je wilt je terminal-sessie beeindigen, moet je PRAXIS opdracht geven je bestand weer op te bergen. Dit doe je met het STOP-commando.
voorbeeld: STOP,KEEP

Je kunt nu met LOGOUT je terminal-sessie beeindigen.

11. HELP

Je kan na het START-commando PRAXIS allerlei opdrachten geven, maar in het begin is het niet iedereen duidelijk welke opdrachten wanneer zinvol zijn. PRAXIS kan je een beetje op weg helpen.
Als je het commando HELP geeft, krijg je een aantal commando-suggesties die je op dat moment zou kunnen geven.

12. DOC

Uitvoerige informatie over alle PRAXIS-items kan je op vragen met het DOC-commando.

bijvoorbeeld : DOC
of : DOC,START

In het eerste geval krijg je een overzicht van alle PRAXIS-items. In het tweede geval specifieke informatie over een PRAXIS-item, in dit voorbeeld het START-commando.

13. een programma maken

Het maken van een programma laat zich in een aantal fases onderscheiden :

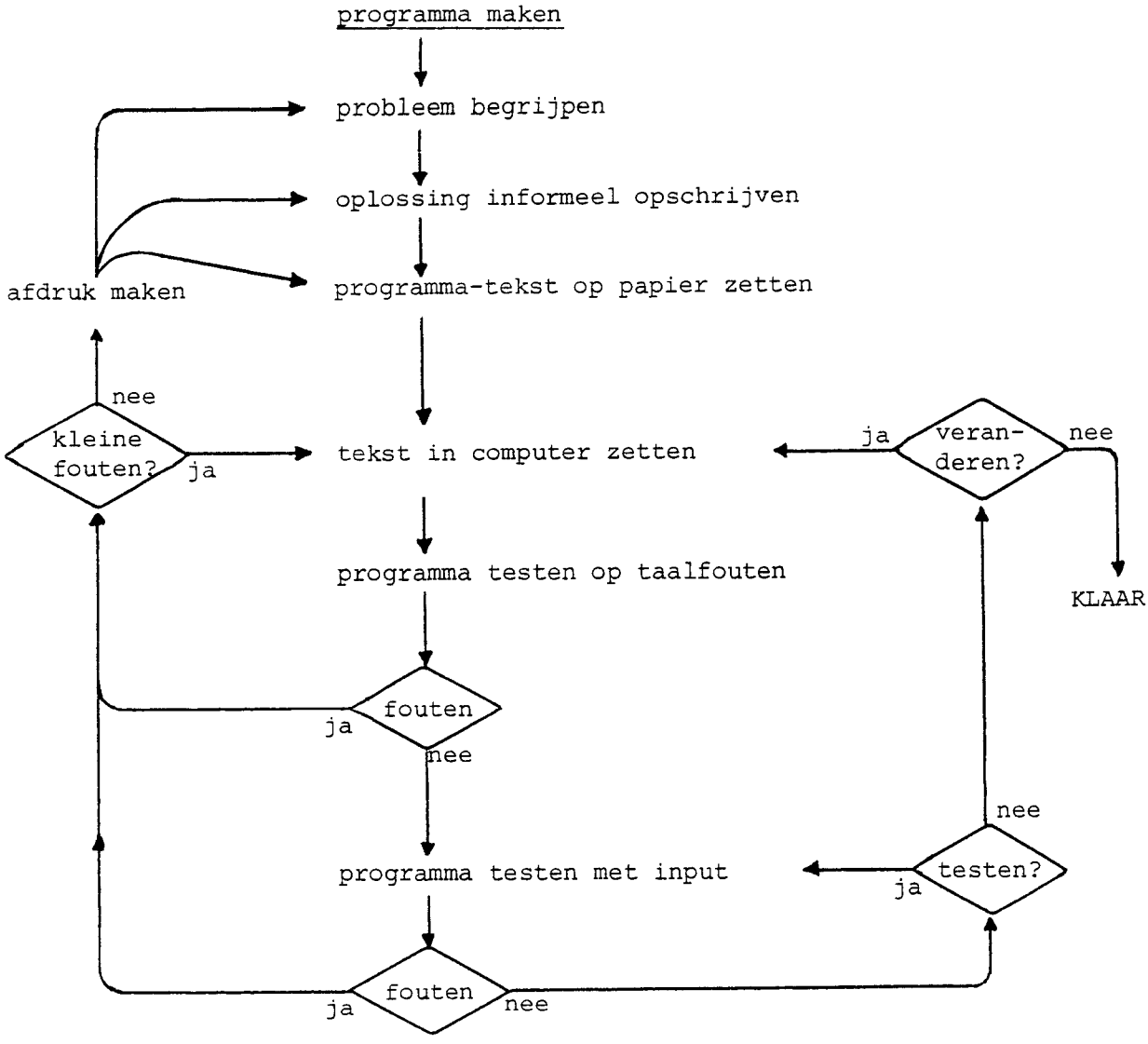
1. het probleem begrijpen
2. het algoritme informeel beschrijven
3. het programma maken op papier
4. het programma in de computer brengen (of veranderen)
5. het testen van het programma op syntax-fouten
6. het syntactisch correcte programma testen met input

De fases 1,2,3 zijn de typische ontwerp-fases en moeten NOOIT achter een terminal gedaan worden. Het kan niet voldoende onderstreept worden dat je niet achter een terminal moet gaan zitten, als je geen nette uitgeschreven versie van je programma op papier hebt staan. Het is bijna onmogelijk om na te denken achter een terminal, en het verzinnen van een programma of programma-details kost achter een terminal altijd meer tijd dan achter een buro met pen en papier erbij.

Fase 4 heet de edit-fase. In het begin zul je de meeste tijd bezig zijn met het invoeren en veranderen van programma-tekst in de computer. Dit manipuleren van tekst is een apart probleem dat niet zoveel met programmeren als wel met het omgaan met computers te maken heeft. Je vindt het terug bij het invoeren en veranderen van data, de tekst van een rapport, een administratie, etc. We gaan in de volgende paragraaf nader op het "editten" in.

De fases 5 en 6 zijn de test-fases. Als je in de edit-fase een programma hebt ingebracht, dan kan je daarna met PRAXIS je programma controleren op syntax-fouten. Als je programma syntactisch niet correct is, moet je, afhankelijk van de ernst van de fout, terug naar 1,2,3 of 4.
Denk je dat de fout gemakkelijk te verhelpen is dan kan je terug naar de edit-fase om je programma een beetje te veranderen.
Het kan echter ook voorkomen dat je niet meteen ziet wat er fout is. Ga dan nooit in het wilde weg zitten gokken, want de kans is groot dat je dan je programma helemaal in de soep draait en veel werk verprutst.
Het beste is om je spullen door PRAXIS op papier af te laten drukken en je PRAXIS-sessie te beeindigen. Je kunt dan thuis nog eens rustig over je probleem nadenken of advies vragen.

Hetzelfde geldt ook voor fouten die je vindt in fase 6.



fases en keuzemomenten tijdens programma ontwikkeling.

14. programma-versies

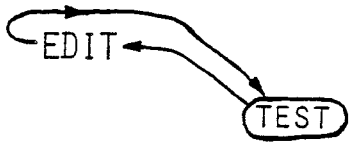
In het START-commando zet PRAXIS het een en ander klaar van de som waar je aan bezig bent. In het begin is dat som 1 en dat blijft zo totdat de je met een commando (het SETSOM-commando) te kennen geeft aan een andere som te willen werken.

Tijdens een sessie heeft PRAXIS een aantal versies van je programma voor je beschikbaar. De eerste keer dat je aan een programma werkt, zijn die versies natuurlijk "leeg", want je hebt er nog niets ingestopt. Later is dat anders, dan blijft er wat bewaard van de vorige keer dat je aan je programma werkte.

Tijdens een PRAXIS-sessie kan je beschikken over de programma-versies ´test´ en ´keep´.
´test´ is de "klad-versie" van je programma waar je tijdens een sessie veranderingen en uitbreidingen op uitprobeert. Het is nuttig om een kladversie te hebben omdat niet elke verandering er een ten goede zal blijken te zijn.
´keep´ is de "net-versie" van je programma. Zonder tussenkomst blijft die de hele sessie ongewijzigd. Er zijn PRAXIS-commando´s om je "net-versie" weer "terug te halen" in je "klad-versie", als je iets aan je "klad-versie" hebt veranderd waar je niet tevreden over bent.

Aan het eind van een sessie, bij het STOP-commando dus, moet je expliciet opgeven wat je wil bewaren tot de volgende keer : de "klad-versie" ´test´ of de "net-versie" ´keep´. De andere wordt dan weggegooid. Aan het begin van de volgende sessie, bij het START-commando, krijg je dan weer een klad- en een net-versie die beide gelijk zijn aan de versie die je de laatste keer bewaard hebt.

15. EDIT

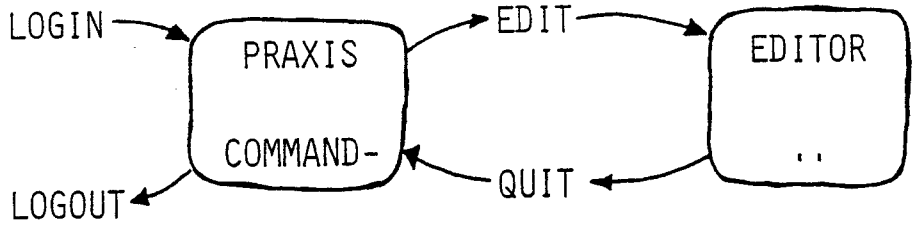


Na het START-commando bevat ´test´ de programma-tekst van de som waar je aan werkt. Je zult ´test´ nogal eens willen veranderen. In deze paragraaf zullen we zien hoe je in het algemeen omgaat met tekst in een computer.

Het manipuleren van tekst gebeurt met een aparte afdeling van de computer die eigenlijk los staat van PRAXIS. Ook als je niet met PRAXIS zou werken zou je die afdeling van de computer gebruiken om met tekst om te gaan. De afdeling heet : de EDITOR. Je kunt EDITOR commando´s geven die te maken hebben met het inspecteren, veranderen en uitbreiden van een tekst die is opgeslagen in de computer.

De EDITOR en PRAXIS zijn gescheiden werelden : je "bent" in de EDITOR of je "bent" in PRAXIS.
Als je in de EDITOR bent, dan kun je EDITOR-commando's geven.
Als je in PRAXIS bent, dan kun je PRAXIS-commando's geven. Je kunt vanuit de PRAXIS-wereld in de EDITOR-wereld springen met het PRAXIS-commando : EDIT. Je kunt vanuit de EDITOR-wereld in de PRAXIS-wereld springen met het EDITOR-commando : QUIT (of Q).

Als je in PRAXIS bent, meldt de computer zich met : COMMAND-
Als je in de EDITOR bent, meldt de computer zich met : ..
Zo kan je dus altijd zien in welke "wereld" je je bevindt en of je EDITOR-commando's of PRAXIS-commando's kan geven.



Na het commando : EDIT beland je in de EDITOR. EDITOR beheert 'test' voor je als een tekst.
Voor de EDITOR bestaat een tekst uit regels, en elke regel uit characters.
Elke regel heeft een nummer. Opeenvolgende regels hebben oplopende (maar niet noodzakelijk opeenvolgende) nummers. Van een tekst van 10 regels kunnen de regel-nummers zijn : 10, 20, 30, 31, 32, 35, 42, 50, 60, 99 .
Elke regel heeft hooguit 72 characters. Elk charachter heeft een positie in die regel. Posities beginnen bij 1 en zijn opeenvolgend. Character-posities liggen dus tussen 1 en 72.

We zullen nu een aantal EDITOR-commando's behandelen. Je doet er goed aan om ze allemaal eens te proberen op een willekeurig tekstje. Hoe handiger je bent met de EDITOR, hoe minder tijd je achter een terminal hoeft te zitten. Er worden hier niet alle EDITOR-commando's behandeld. Een volledige opsomming vind je in het ACCU-bulletin over SUEDI5, verkrijgbaar bij de receptie van het ACCU.

In het onderstaande stellen
<r1>,<r2>,<r3>,.... willekeurige regelnummers
<p1>,<p2>,.... willekeurige positie-nummers
<t1>,<t2>,.... willekeurige character-rijtjes voor.
Met LAST wordt het nummer van de laatste regel aangegeven.
Met CUR wordt het nummer van de regel aangegeven die het laatst op het scherm is geschreven.

ALLE EDITOR-COMMANDO'S, ALSMEDE last EN cur KUNNEN WORDEN AFGEKORT MET HUN EERST LETTER.

1. RESEQUENCE

commando: RES
of RES,<rl> VB RES,200
of RES,<rl>,<inc> VB RES,150,20

Het RESequence-commando voorziet de tekst van een nieuwe regelnummering. Je kunt daarbij opgeven wat het nummer <rl> van de eerste regel moet zijn en met hoeveel <inc> de nummers op moeten lopen. Als je <rl> of <inc> weglaat wordt daar 10 voor genomen.

2. CREATE

Commando : CREATE

Met het CREATE-commando kun je een geheel nieuwe tekst invoeren. De oude tekst wordt weggegooid.
Na het CREATE-commando meldt EDITOR : "ENTER LINES" en kun je regels invoeren. Een regel sluit je af met het drukken op de "return"-toets.

JE BEEINDIGT HET TOEVOEGEN VAN TEKST MET EEN is-gelijk-teken ("=") OP EEN NIEUWE, VERDER LEGE REGEL.

De nieuwe tekst krijgt regelnummers 10,20,30,....

3. ADD

Commando : ADD
ADD, <rl> VB ADD,200
ADD, <rl>,<inc> VB ADD,31,1

Met het ADD-commando kan tekst op een willekeurige plaats toegevoegd worden.
Na het ADD-commando meldt de EDITOR : "ENTER LINES" en kan je regels invoeren. Je beeindigt het invoeren van tekst met een is-gelijk-teken op een verder lege regel. De nieuwe tekst krijgt regelnummers beginnend met <rl>, oplopend met <inc>. Het is wel zo dat hierbij geen bestaande regels overschreven of gepasseerd mogen worden. Als <inc> weggelaten wordt, wordt hiervoor 10 genomen. Als <rl> weggelaten wordt, voegt de EDITOR de nieuwe tekst achter de bestaande tekst toe.

4. <rl> = ...

Men kan regels toevoegen of veranderen door een regel in te typen vooraf gegaan door een nummer en een "=".
VB 187=wat nu?

De EDITOR antwoordt niet met .. Je kunt gewoon doortypen.

5. LIST

Commando : LIST,[range-optie],[string-optie]

Met het LIST-commando kan opdracht gegeven worden bepaalde regels op het scherm te schrijven. Het is nogal een uitvoerig commando, omdat er een aantal manieren zijn om op te geven welke regels getoond moeten worden.

1. [range-optie]
   Met de range-optie wordt opgegeven welke regelnummers in principe gebruikt zullen worden.

   ALL : alles.
   LIST,ALL betekent toon de hele tekst.

   <r1> : een regel.
   LIST,30 betekent toon regel 30.
   LIST,LAST betekent toon laatste regel

   <r1>,<r2> : een gebied.
   LIST,200,LAST betekent toon regel 200 t/m de laatste.

2. [string-optie]
   Met de string-optie kan opgegeven worden dat alleen de regels gelist hoeven te worden waar een bepaalde string in voorkomt. (Een "string" is een character-rijtje.)

   voorbeeld :
   LIST,ALL,/pip/ : list alle regels waar de string "pip" in voorkomt.
   LIST,200,LAST,/pip/ : list alle regels, tussen 200 en de laatste, waar"pip" in voorkomt.

6. DELETE

Commando : DELETE [range-optie],[string-optie],[veto-optie]

Met het DELETE-commando kan men regels uit de tekst verwijderen. Range- en string-optie werken net zoals bij het LIST-commando, behalve dat bij het DELETE-commando de regels verwijderd worden uit de tekst.

[veto-optie] : ,VETO VB : d,10,30,VETO

Als de veto-optie gegeven wordt, worden de regels die voor deletion in aanmerking komen eerst op het scherm getoond en moet met "y" of "n" aangegeven worden of de getoonde regel daadwerkelijk verwijderd moet worden.

7. SUBSTITUTIE

Commando : /<t1>/=/<t2>/,[range-optie],[veto-optie]

In alle regels, opgegeven met de range-optie, wordt de string <t1> vervangen door de string <t2>. Als je de veto-optie opgeeft, worden regels die in aanmerking komen voor verandering eerst op het scherm getoond en moet je met "y" of "n" aangeven of de getoonde verandering aangebracht moet worden.
VB /aap/=/noot/,ALL,VETO

8. MODIFY

Commando : MODIFY,<r1> VB MODIFY,120

Met het MODIFY-commando kan een regel veranderd worden. Na het MODIFY-commando wordt de te veranderen regel getoond op het scherm. Daarna moet je in de regel eronder aangeven hoe de regel veranderd moet worden :
: laat het character erboven weg
blank : neemt het character erboven over
& : verander bovenstaand character in een blank
^ of ~: voeg tekst in, voor het character boven ^ of ~. Tekst wordt afgesloten met een # of <return>

```
VB    : ..MODIFY,100
          100=aappp nootxmies
             ?   ##     &
        ..1,100
          100=aap noot mies

VB    : ..MODIFY,100
          100=aap noot mies
             ?   ^ wim#
        ..1,100
          100=aap wim noot mies
```

We willen hier nogmaals in herinnering brengen dat de EDITOR-commando's afgekort mogen worden.

LIST,ALL	mag zijn	L,A
MODIFY,120	mag zijn	M,120
QUIT	mag zijn	Q
DELETE,CUR,VETO	mag zijn	D,C,V

= = = = = = =

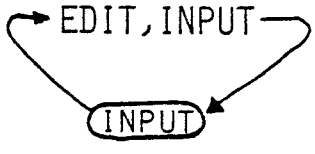
16. COMPILE en ´comlist´

Als je een complete programma-tekst in ´test´ gereed hebt, moet deze tekst op taal-fouten getest worden. Dit gebeurt met het commando: COMPILE

COMPILE pakt ´test´ en speurt naar fouten, d.w.z. incorrect taal-gebruik zoals ongedeclareerde variabelen, fouten in haakjes paren, punt-komma vergeten etc. Vindt het zo´n fout dan wordt een foutmelding op het scherm geschreven. Die moet je even onthouden om later met EDIT je programma-tekst te kunnen verbeteren. Als er veel fouten in ´test´ zitten, begin dan de fouten te verbeteren die het eerst gemeld worden. Als er zoveel fouten zijn dat de meldingen "van het scherm lopen", onderbreek dan de meldingen door op de spatie-balk te drukken (door een aanslag op de return-toets "lopen" de meldingen weer verder). Maak een notitie van de eerste fouten en verbeter die in de EDITOR. De andere fouten verbeter je dan maar in een tweede slag.

COMPILE maakt ook een "compiler-listing" van ´test´. Een "compiler-listing" is een programma-tekst, die door COMPILE voorzien is van een commentaar, foutmeldingen etc.
Deze "compiler-listing" heet: ´comlist´
´comlist´ kun je eventueel uit laten printen met het PRINT-commando.
´comlist´ bevat altijd de listing van ´test´. Als je met EDIT een nieuwe versie van ´test´ maakt, wordt ´comlist´ weggegooid. ´comlist´ is dan niet meer beschikbaar, totdat je weer een keer COMPILE hebt gegeven.

17. ´input´



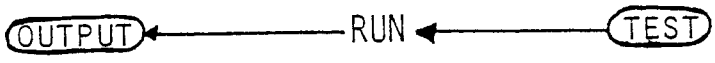
De meeste programma´s verwerken data die als input gelezen wordt in het programma. PRAXIS kan een bestand met datasets bijhouden voor je, die je kunt gebruiken bij het uittesten van je programma´s. Dat bestand heet : ´input´.
Met het commando EDIT,INPUT beland je in de EDITOR met als werk-tekst: ´input´. Je kunt dan het bestand ´input´ veranderen en uitbreiden. Voor een overzicht van de EDIT-commando´s zie EDIT.

Je kunt in ´input´ een aantal afzonderlijke datasets bewaren door ze te scheiden met een regel die alleen bevat: "*eor"

```
VB:  10=aap noot mies          }      1e dataset
     20=wim zus jet            }
     30=*eor
     40=boter melk kaas        }
     50=3.14159265358979       }      2e dataset
     60=3238462643389          }
     70=*eor
     80=abcdefghijk            }      3e dataset
```

Je kunt 'input' af laten drukken met het PRINT-commando.

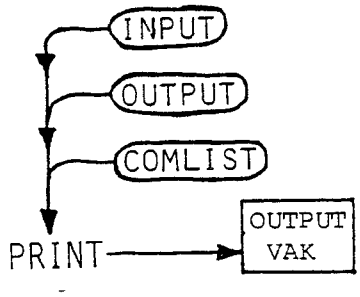
18. RUN en 'output'



Als 'test' geen taal-fouten bevat, kun je het programma 'test' laten uitvoeren met het commando: RUN. De 'output', die het programma met write-statements produceert, wordt op het scherm geschreven en op een file : 'output'. 'output' kan je laten uitprinten met het PRINT-commando. Als je het programma met read-statements input laat lezen, wordt gelezen van de file 'input', en wel de eerste data-set. Als je het programma wilt laten werken met een andere data-set, moet je dat in het RUN-commando opgeven door het nummer van de data-set te specificeren.
Als je bijvoorbeeld de derde data-set wilt gebruiken, geef dan het commando : RUN,3 .

'test' kan alleen gerund worden als het een keer gecompileerd is, want behalve dat COMPILE op syntax-fouten let, maakt COMPILE een voor de computer begrijpelijke, executeerbare versie aan van 'test'. Deze versie wordt gebruikt door RUN. Als je RUN aanroept en 'test' is nog niet gecompileerd, dan kan RUN dus niks doen. Daarom roept RUN zelf COMPILE aan als 'test' nog niet is gecompileerd. Als 'test' foutloos is wordt het uitgevoerd.

19. PRINT



Met het PRINT-commando kan je 'input' en/of 'output' en/of 'comlist' af laten drukken. Bij het PRINT-commando moet je een naam opgeven, die aangeeft in welk output-vak de listing terecht moet komen.

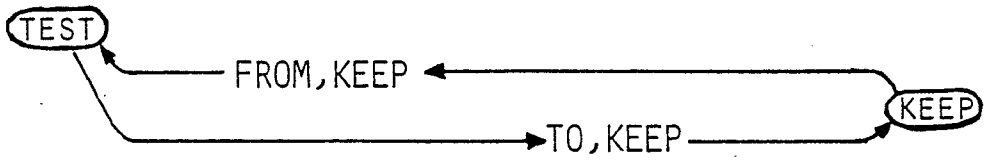
Het commando is : PRINT,NAME=...,XXXXX
Een NAME bestaat uit hoogsten 5 letters of cijfers en moet beginnen met een letter.
XXXXX is een selectie uit (input,output,comlist).

V.B. PRINT,NAME=MARIE,OUTPUT,COMLIST
PRINT,NAME=MARIE,INPUT,OUTPUT

PRINT roept niet zelf COMPILE of RUN aan om de gevraagde listing te maken, als 'comlist' of 'output' er niet is.
De volgorde van de parameters in het PRINT-commando is onbelangrijk.

In het PRINT-commando kan een SITE-parameter opgegeven worden als niet op het ACCU geprint moet worden.
B.v. PRINT,COMLIST,NAME=PPP,SITE=... Voor ... moet je de 'FID' van de printer invullen. Die kan je te weten komen bij de beheerder van de printer. Als je achter een 'hardcopy-terminal' zit, kan met 'SITE=HERE' opgegeven worden dat de gewenste output op de terminal zelf moet worden afgedrukt. In dat geval kan 'NAME=...' achterwege gelaten worden.
Voorbeeld : PRINT,COMLIST,INPUT,SITE=HERE

20. versies : 'test' en 'keep'



Tijdens een PRAXIS-sessie, dus tussen START en STOP, kan je beschikken over een aantal versie van je som. Naast 'test' die we al hebben leren kennen, bekijken we hier ook 'keep'.

'test' : 'test' is de werk-versie van je som die gebruikt wordt bij akties zoals EDIT, COMPILE, RUN, PRINT. Het is een klad-versie waar je veranderingen op aanbrengt en uitprobeert.

'keep' : 'keep' is de net-versie. Als je een 'test' hebt waar je tevreden over bent, moet je die versie bewaren voor je nieuwe veranderingen gaat aanbrengen in 'test'. Deze versie zet je in 'keep' met het TO-commando.

TO,KEEP Het betekent : 'keep' := 'test'

De vorige versie van 'keep' wordt met 'test' overschreven. Als je daarna verbeteringen in 'test' hebt aangebracht waar je tevreden over bent, kan je weer zeggen : TO,KEEP. Als je niet tevreden bent met de veranderingen, kan je je nette versie weer terug krijgen met het FROM-commando :

FROM,KEEP Het betekent : 'test' := 'keep'

Aan het eind van een PRAXIS-sessie, bij STOP dus, wordt de kladversie 'test' weggegooid en wordt de netversie 'keep' bewaard tot de volgende keer. Bij START wordt 'test' dan weer geinitialiseerd op 'keep'.

21. STOP

Met het STOP-commando beeindig je een PRAXIS-sessie. Bij het STOP-commando moet expliciet opgegeven worden of ´test´ in ´keep´ gezet moet worden of niet.

Het commando is : STOP,KEEP
of : STOP,NOKEEP

In het eerste geval wordt eerst het commando TO,KEEP uitgevoerd, ofwel ´keep´:=´test´. De huidige versie van ´test´ blijft dan dus bewaard tot de volgende keer.
In het tweede geval blijft de huidige versie van ´keep´ bewaard tot de volgende keer. Die is gelijk aan de versie die je had toen je de het START-commando gaf, behalve als je tijdens je sessie een keer het commando TO,KEEP gegeven hebt. Dan blijft de versie bewaard die je toen had.

22. SETSOM

Met het SETSOM-commando kan je het nummer van de som waar je aan werkt opvragen of veranderen.
Na het commando : SETSOM meldt PRAXIS welke som er gepakt is.
Om op een andere som over te stappen moet je het commando geven :

SETSOM,<somnummer>,KEEP
of : SETSOM,<somnummer>,NOKEEP

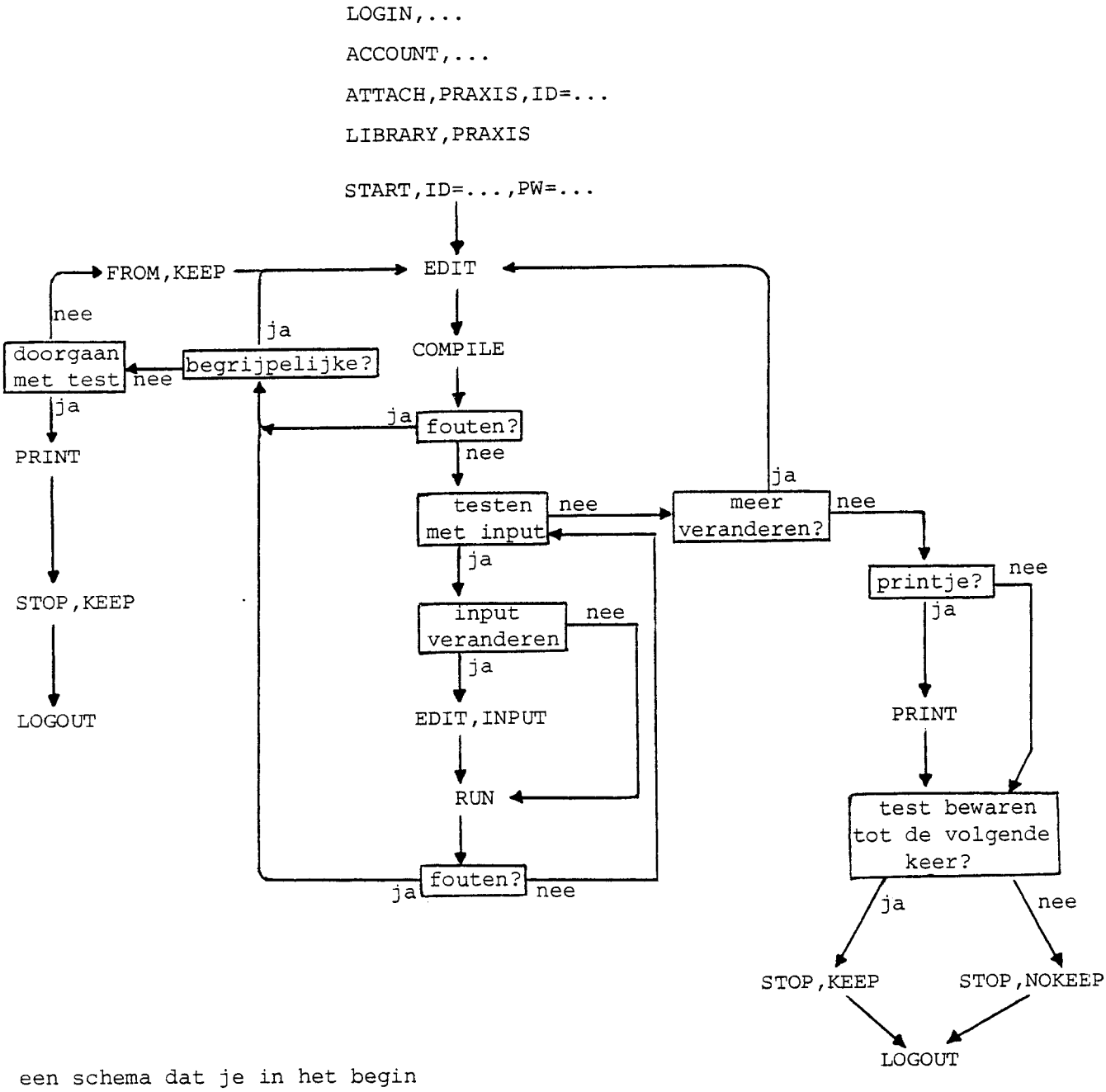
<somnummmer> moet een getal tussen 1 en 5 zijn. Er moet expliciet opgegeven worden of de huidige kladversie ´test´ al dan niet bewaard moet blijven. In het eerste geval wordt eerst TO,KEEP gedaan. SETSOM gooit eventueel ´comlist´ of ´output´ weg.

23. SETLAN

Met het SETLAN-commando kan je de taal waar PRAXIS mee werkt opvragen of veranderen.
Na het commando : SETLAN meldt PRAXIS welke taal het gebruikt.
Om op een andere taal over te stappen moet je het commando geven :

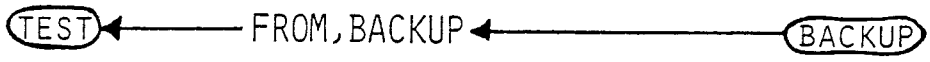
SETLAN,ALGOL68
of : SETLAN,PASCAL

SETLAN gooit eventueel ´comlist´ en/of ´output´ weg.



een schema dat je in het begin
zou kunnen hanteren in een
praxis-sessie.

24. versie : ´backup´



Behalve over ´test´ en ´keep´ kun je tijdens een sessie beschikken over de programma versie ´backup´. ´backup´ is de programma versie die je had aan het begin van je sessie. Je kunt ´test´ overschrijven met ´backup´ dmv. het FROM-commando:

FROM,BACKUP betekent : ´test´ := ´backup´

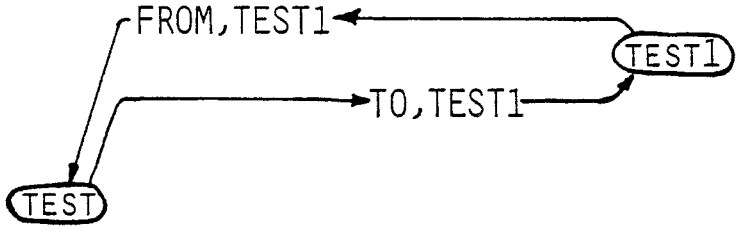
Dit kan alleen nuttig zijn als zowel ´test´ als ´keep´ verprutst zijn. Overnieuw beginnen met ´backup´ in ´test´ betekent zoveel als je sessie overnieuw beginnen. Het is een soort "safety-catch": je kunt nooit meer dan het werk van een sessie verprutsen door een fout.

Als je dus alles (ook ´keep´) verprutst hebt en je wilt je terminal sessie staken, typ je :

```
FROM,BACKUP
STOP,KEEP
LOGOUT
```

Vergeet dit niet anders ben je de volgende keer alles kwijt!

25. versie : ´test1´



In ´test1´ kun je tijdens een sessie, een programma-versie even bewaren. Je kunt er ´test´ in schrijven met het TO-commando en ´test´ overschrijven met de versie uit ´test1´ met het FROM-commando.

```
TO,TEST1     betekent : ´test1´ := ´test´
FROM,TEST1   betekent : ´test´  := ´test1´
```

Bij het START-commando wordt er niks in TEST1 gezet en bij het STOP-commando wordt dat wat er in ´test1´ zit, weggegooid.

´test1´ kun je gebruiken om ´test´ even in op te bergen en een routine van ´test´ even apart onder de loep te nemen of te testen. ´test1´ zal in het begin van weinig praktisch nut blijken.

26. SHOW

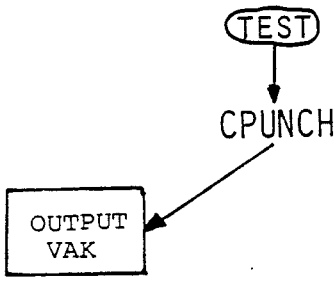
Met het SHOW-commando kunnen alle teksten bekeken (maar niet veranderd) worden. Als je 'x' wilt bekijken, typ je : SHOW,X.
X kan zijn : TEST,COMLIST,INPUT,OUTPUT,KEEP,BACKUP,TEST1.

Na SHOW,X beland je in de EDITOR met een copie van 'x' als werk-tekst. Met EDITOR-commando's (1,a etc.) kun je de tekst bekijken. SHOW werkt met een copie, en veranderingen die je op de tekst aanbrengt worden weggegooid.

Je verlaat het SHOW-commando met : Q (van QUIT)

Als je SHOW,COMLIST of SHOW,OUTPUT geeft en de gevraagde tekst bestaat niet, dan geeft SHOW een foutmelding. Je moet eerst zelf met COMPILE of RUN de gewenste tekst maken.

27. CPUNCH



Soms kan het gebeuren dat je niet meer interactief aan je som wilt werken, maar liever in de Batch met ponskaarten door wilt gaan. Bijvoorbeeld omdat het onmogelijk is om een terminal te krijgen voor langere tijd. Je kunt ook zo werken dat je je eerste versie intypt via de terminal en met PRAXIS er de syntax-fouten uithaalt, maar daarna het testen doet met Batch-jobs.

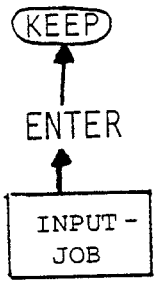
Bij het CPUNCH-commando wordt een copie van 'test' naar de CardPUNCHer gestuurd. In een Output-vak krijg je dan 'test' in de vorm van een pak ponskaarten. Elke ponskaart bevat een regel van 'test'. De ponskaarten bevatten wel 'test' in de vorm van ponsgaatjes, maar nog niet in de vorm van letters. Je moet ze nog interpreteren (zie de handleiding ponsmachines, Accu-Advies-kamer).

In het CPUNCH-commando moet je het OUTPUT-vak opgeven waar je de ponskaarten wilt hebben.

commando : CPUNCH,NAME=....

'NAME' moet bestaan uit hoogstens 5 letters of cijfers en moet beginnen met een letter.

28. ENTER



Als je een programma-tekst op ponskaarten hebt en je wilt die tekst gebruiken als een PRAXIS-som, dan kun je met het ENTER-commando die tekst "opsturen" naar je PRAXIS-bestand.
Stel dat je een tekst op ponskaarten hebt en je wilt die gebruiken voor SOMNUM=3 in PRAXIS-bestand met ID=xxx en PW=yyy.
Je kunt de tekst in het bestand plaatsen met de volgende job:

```
e/o/f
<jobname>.
ACCOUNT,<account-nummer>,SN=<account-subnummer>.
ATTACH,PRAXIS,ID=...
LIBRARY,PRAXIS.
ENTER,ID=xxx,PW=yyy,SOMNUM=3.
e/o/r

ponskaarten met je programma

e/o/f
```

ENTER kan niet gegeven worden als het bestand in gebruik is aan een terminal. ENTER heeft tot gevolg dat de ponskaart-tekst geplaatst wordt in 'keep'

ENTER betekent 'keep' := <card-tekst>

APPENDIX I : Commando´s voor gebruikers in praktikum-verband.

PRAXIS-gebruikers die in praktikum-verband werken hebben twee commando´s extra om met hun praktikum-leiding te communiceren.

1. MAIL

   Na het commando : MAIL beland je in de EDITOR waar je tekst kan intikken die naar de praktikumleiding wordt gestuurd. Vermeld in zo´n brief je naam als je geen anonieme brief wilt sturen.
   Je verlaat de EDITOR met het commando Q (van QUIT).

2. SUBMIT

   Met het SUBMIT-commando kan je ´test´ opsturen naar de praktikum-leiding. Je moet daarbij het nummer van de som vermelden waarvoor ´test´ als oplossing moet dienen.
   Het commando is : SUBMIT,SOMNUM=....

Overigens heeft de praktikum-leiding de mogelijkheid om de hele praktikum-groep of praktikanten individueel een bericht te sturen dat op het scherm wordt geschreven bij het START-commando.

APPENDIX II : Commando's voor individuele gebruikers.

Mensen die met PRAXIS,ID=SOLO werken hebben een aantal PRAXIS-commando's extra tot hun beschikking die te maken hebben met het beheren van hun PRAXIS-bestand.

1. MAKEMF

   Met het MAKEMF-commando kan je een PRAXIS-bestand voor jezelf klaarzetten. Je moet daarbij de ID en PW opgeven die bij het bestand moeten horen. Daarnaast moet je ook de taal opgeven die je in eerste instantie wilt gebruiken. Je kunt daarvoor kiezen uit ALGOL68 en PASCAL.
   Het commando is : MAKEMF,ID=...,PW=...,LAN=...

2. PURGEMF

   Met het PURGEMF-commando ruim je een PRAXIS-bestand op. Je moet daarbij de ID en PW opgeven van het op te ruimen bestand.
   Het commando is : PURGEMF,ID=...,PW=...

3. OKMF

   Door niet te voorziene omstandigheden kan je PRAXIS-bestand "kapot" gaan. Het OKMF-commando probeert zo'n PRAXIS-bestand dan weer te "repareren". Bij het OKMF-commando moeten de ID en PW van het te repareren bestand opgegeven worden. Eveneens moet de voorkeurs-taal opgegeven worden.
   Het commando is : OKMF,ID=...,PW=...,LAN=...

Een PRAXIS-bestand, aangemaakt onder de normale ACCU-regelingen, wordt weggegooid als het 5 werkdagen niet is gebruikt. Onder gebruiken wordt minimaal een START verstaan.

APPENDIX III : Interactieve IO in PASCAL.

Gegevens kan je in je programma inlezen van ´input´ en schrijven naar ´output´. De ´input´ van je programma moet klaar staan voor je programma begint te lopen , en je ´output´ krijg je pas nadat het programma helemaal klaar is.
Er is echter ook een methode om direct met je programma te "converseren" via het toetsenbord en scherm van je terminal, terwijl het programma loopt.

Hieronder staat een programma dat getallen van het toetsenbord leest totdat het getal 0 gelezen wordt. Dan drukt het de som van de ingelezen getallen in.

```
 1.     PROGRAM sum (INPUT,OUTPUT) ;
 2.     VAR keys, screen   :   TEXT     ;
 3.         num,sum        :   INTEGER  ;
 4.
 5.     PROCEDURE connect ( VAR fff : TEXT ) ; EXTERN ;
 6.
 7.     BEGIN  connect(screen) ; connect(keys) ; sum:=0 ;
 8.
 9.            REPEAT WRITELN(screen,´ geef getal ´) ;
10.                   RESET(keys) ; READ(keys,num) ;
11.                   sum:=sum+num
12.            UNTIL  num=0 ;
13.
14.            WRITELN(screen,´ som is ´,sum:1)
15.     END.
```

Verklaring :

regel 2 : lezen van het toetsenbord en schrijven naar het scherm gaat met de VARs keys en screen van het type TEXT. INPUT en OUTPUT zijn ook van het type TEXT.

regel 5 : "connect" is een procedure die een TEXT geschikt maakt voor interactief gebruik. De externe declaratie heeft PASCAL nodig.

regel 7 : ´screen´ en ´keys´ worden interactief gemaakt, dwz. vastgemaakt aan scherm en toetsenbord.

regel 9 : Je kan naar het scherm schrijven met het WRITE-statement. Je moet als eerste parameter in het WRITE-statement dan ´screen´ meegeven.

regel 10 : lezen van ´keys´ gaat idemdito. Je kan van ´keys´ lezen met het READ-statement. Je moet dan wel als eerste parameter ´keys´ meegeven. Speciaal voor interactief lezen is dat je voor elke READ-opdracht een RESET-statement moet zetten. Dat is lastig maar onvermijdelijk.

regel 14 : idem als regel 6.

Het algemene schema voor "geef een prompt, en lees data" is :

```
 a.  writeln(screen,´ prompt prompt prompt ´) ;
 b.  reset(keys) ; read(keys, ... data ... ) ;
```

Let op de writeln in (a) en de reset in (b).

APPENDIX IV : Beyond PRAXIS ....

PRAXIS is een beperkt systeem. Als je meer wilt, zal je meer moeten leren over commando's en data-opslag in de computer. Een vaste methode om dat te doen is er niet. Na PRAXIS begint de jungle van NOS/BE, het operating system van de CYBER-computer. Daarmee communiceert de "gewone" gebruiker met de computer.

PRAXIS is opgebouwd uit NOS/BE. Elk PRAXIS-commando bestaat uit een groepje NOS/BE-commando's. Je zou NOS/BE kunnen leren door te lezen hoe en waaruit PRAXIS-commando's zijn opgebouwd. Documentatie daarvoor ontbreekt echter vooralsnog.
Een andere weg is die van elke andere ACCU-gebruiker. Koop de ACCU-flodders en -bulletins over SUEDI5, INTERCOM, TAPES en FILES, PERMANENT FILES en MASTERFILEs. Door studeren en proberen moet je al die zaken dan maar zien te hanteren.

INDEX :::