Villy Lymann
~~Rud. Wulffsgade 24 - 2.tv.~~
~~DK-8000 Århus C~~

DATALOGISK INSTITUTAFD:

HJÆLP3 SYSTEMET VED GIER

redigeret
af
FRED MOSEKJÆR MADSEN

Villy Lymann

1971

INDHOLDSFORTEGNELSE

HELP3 SYSTEMET VED DATALOGISK INSTITUTAFD:

For at kunne kommunikere med datamaten Gier må man kende lidt til help3. En beskrivelse findes i A Manual of Help3 ed. Søren lauesen, udgivet af regnecentralen, København 1967. Nedenfor er givet et nødtørftigt resume af indholdet i denne manual, samt en beskrivelse af de udvidelser der eksisterer her på stedet. Det anbefales at benytte manualen som reference.

Help3 er et system af hjælpeprogrammer, som hjælper brugeren af datamaten med at rette, oversætte og køre programmer.

EKSEMPEL 1.

Man vil have indlæst og oversat et algolprogram, som er hullet på papirstrimmel. På strimlen står:

```
algol<
begin
   .
   .
program
   .
   .
end
```

Hulstrimlen lægges i læseren, og man skriver følgende hjælp3 information på skrivemaskinen:

```
r<
```

Virkningen af dette vil være, at strimmellæseren aktiveres (r=reader) og indlæsningen startes. Det første der indlæses er algol<. Indlæsningen standses, og Gier algol4 compileren hentes frem fra baggrundslageret. Derpå fortsættes indlæsningen af det egentlige algolprogram som oversættes til maskinkode.

Hjælp3 bestyrer, hvorfra der skal indlæses, og hvortil der skal udskrives. I ovenstående eksempel, kunne man også have tænkt sig at programmet var lagret på Giers baggrundslager, under navnet abcde, og vi ønsker derfor at algol oversætteren skal tage input derfra. Hjælp3 informationen til skrivemaskinen bliver da:

```
abcde<
```

———— o ————

HJÆLP3 KATALOGET

For at kunne administrere input og outputmedierne, som altså omfatter både ydre medier (strimmellæser, skrivemaskine, perforator, linieskriver mm.) og indre medier (navngivne, reserverede områderpå baggrundslagrene), samt hjælpeprogrammerne, har hjælp3 et katalog. Dette katalog indeholder en række navne, nemlig:

1) navne på ydre input-output enheder
2) navne på hjælpeprogrammer
3) navne på reserverede områder på baggrundslageret
4) navne på konstanter

Foruden navne indeholder kataloget også oplysninger om, hvor man kan finde et navngivet område etc.

EKSEMPEL PÅ KATALOG

navn:	kommentar:
free	det frie areal i baggrundslageret. hertil indlæses der når man vil have reserveret og navngivet et areal.
work	et arbejdsareal. Det er her det oversatte algolprogram bliver lagret.
date	en konstant, som indeholder oplysning om dato og nr hjælp-kald den dag.
image	et areal, hvor indholdet af ferritlageret gemmes af vejen ved et kald af hjælp3.
r	strimmellæser (r=reader)
t	skrivemaskine som input (t=typer)
p	perforator
l	linieskriver
w	skrivemaskine som output (w=writer)
ga 4	gier algol 4 - oversætteren
algol	et hjælpeprogram til at starte processen, at oversætte et algolprogram.
run	et hjælpeprogram til kørsel af allerede oversatte programmer.
edit	et hjælpeprogram til rettelse af texter (herunder programmer)
res	et hjælpeprogram som reserverer den del af free der er booked, under det navn, der er givet som parameter.

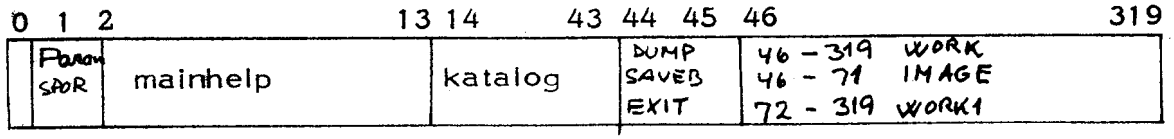
BAGGRUNDSLAGRE I GIER

Foruden ferritlageret, som er på 1024 celler (ell. ord) a 40 bit, og bufferlageret, der er på 4096 ord, har Gier tilknyttet følgende baggrundslagre:

1 tromle, 1 disk, 2 båndstationer

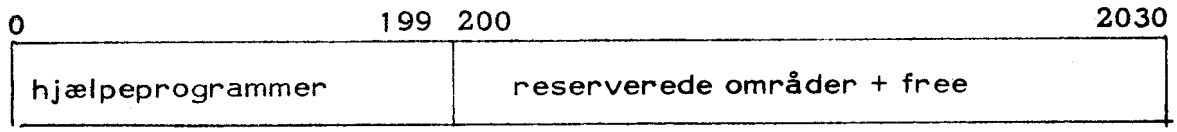
TROMLEN

Består af 320 blokke (kanaler) a 40 ord, og er af hjælp3 delt op således:



DISKEN

der består af2030 blokke (spor) a 600 ord, er af hjælp3 delt op således:



BÅNDSTATIONERNE

anvendes ikke fast af hjælp3, og er altså frit tilgængelige for brugerne.

HJÆLP3 INFORMATION

består i det væsentlige af en liste af navne. Desuden kan forekomme understregede bogstaver, samt heltal. Listens elementer adskilles af komma, og listen afsluttes med < (udtales hak).

EKSEMPEL r, edit, o_free<

Hjælp3 informationen indlæses af hjælps løbende inputmedium, som i almindelighed vil være skrivemaskinen. Hvis dette ikke er tilfældet (den grønne lampe lyser ikke) kan man trykke på en knap ved Giers kontrolbord, HP-knappen (=hjælpknappen). Herved hentes hjælp3 maskinkodeprogrammet ind i ferritlageret og startes, noget af det første der sker er at skrivemaskinen vælges som løbende input, og hjælp3 information kan nu indlæses.

Efter indlæsningen, som slutter når der mødes et <, vil hjælp3 udfra katataloget identificere listens navne et for et. Hvis et navn ikke findes i kataloget, vil der komme en udskrift på skrivemaskinen (med rødt): undef. Ellers sker der følgende:

Hvis navnet er

1) en outputenhed, vælges denne som hjælps outputmedium, og hjælp3 fortsætter i listen.

2) en inputenhed eller et reserveret område i baggrundslageret, vælges dette som hjælps løbende inputmedium, og hjælp fortsætter i listen.

3) et hjælpeprogram, vil resten af listen blive flyttet til en parameterkanal, og hjælp3 vil kalde det pågældende hjælpeprogram ind i ferritlageret, og overgive kontrollen til dette, Hvad der vil ske med resten af listen, afhænger af det pågældende hjælpeprogram, idet resten af listen opfattes som parametre til hjælpeprogrammet. Når hjælpeprogram met er færdigt, venter Gier på hjælp3 information fra hjælps løbende inputmedium.

Møder hjælp3 et < i listen uden at have overgivet kontrollen til et hjælpeprogram, vil hjælps løbende inputmedium blive valgt som inputmedium, og ny hjælp3 information vil blive læst herfra.

EKSEMPEL 3

det samme som eksempel 1.

der skrives r< på skrivemaskinen. navnet r tydes af hjælp3, som et legitimt navn på et inputmedium, nemlig strimmellæseren. Denne vælges som hjælps løbende inputmedium, og da der står et hak efter r'et, starter hjælp3 nu indlæsning fra strimmellæseren. I dette tilfælde må papirstrimlen indledes med hjælp3 information. Forrest på strimlen står algol<, som medfører et kald af hjælpeprogrammet algol, der igen kalder hjælpeprogrammet ga4, der indlæser og oversætter programmet. Hvis algolprogrammet blev oversat ville der på skrivemaskinen komme udskriften "ok", og hjælpeprogrammet er færdigt. Hjælp venter nu på mere information fra løbende input (i dette tilfælde strimmellæseren) og det havde derfor været hensigtmæssigt, hvis papirstrimlen havde haft følgende udseende:

algol< begin .....end run<

thi da ville programmet straks efter oversættelsen blive kørt. kræver programmet data fra strimmel er det mere hensigtsmæssigt at lade programstrimlen se således ud:

algol< begin.....end t<

efter endt oversættelse og ok vil Gier læse t< og derfor vælge skrivemaskinen (typewriter) som løbende inputmedium. Man lægger nu sin datastrimmel i læseren, hvor datastrimlen har følgende udseende:

run< data1, data2.....

man taster nu r< på skrivemaskinen, hvorefter programmet bliver kørt.

VIGTIGT

Man bør altid tilrettelægge sin kørsel på Gier således at man kun skal taste r< på skrivemaskinen, derved sparer man megen tid for sig selv, og mange irritationsmomenter for de øvrige brugere, der venter på at komme til

HJÆLPEPROGRAMMER.

I det følgende vil vi beskrive nogle meget brugte programmer, som også findes beskrevet i amanual of help3, desuden vil de udvidelser af hjælp3, som ikke er beskrevet i manualen, blive beskrevet her.

algol

kalder en algol4 oversætter med hjælps løbende inputmedium, som programkilde. Man kan kalde algol med forskellige parametre, f. eks.

algol, n< algol, 10<

n betyder at algolprogrammet skal køres uden indexcheckpå de indicerede variable (benyttes kun ved indkørte programmer, man sparer ca0. 3 msek, pr. indiceret variabel).
10 bevirker, at der kommer udskrift af hver 10. linie i algolprogrammet (en hjælp ved fejlsøgning - andre positive heltal kan benyttes).

Normalt vil den kaldte oversætter være standardudgaven ga4, men man kan vælge en anden udgave ved at give dennes navn, som første parameter til kaldet af algol. Der findes i øjeblikket følgende muligheder:

algol<	henter standardudgaven ga4
algol, pl<	henter oversætter med plotteradministration
algol, samba<	henter oversætter med båndadm.
algol, sambaplot<	henter oversætter med bånd og plotteradm.
algol, buffer<	henter oversætter, der kan udføre visse mindre algolprogrammer hurtigere end ga4.

eksempel:

hvis man har et program, der anvender plotteren, stående på området pax og man ønsker det oversat med udskrift på printeren af hver linie, og uden indexcheck, er den nødvendige hjælp3 information altså

l, pax, algol, pl, n, 1<

Lidt om bufferoversætteren.

Denne oversætter er i princippet en almindelig algol4-oversætter uden plotteradministration. Den udmærker sig ved, at det færdigoversatte program, når det startes, begynder med at flytte sig selv over i bufferen fra celle 0 og fremefter. Det betyder, at den operation at hente et kanal segment ind i ferritlageret i stedet for at tage 20 millisek. nu tager 0. 5 millisek., så hvis programmet tidligere anvendte megen tid på at vente på tromletransporter, kan man spare en del køretid. At programmet venter på tromletransporter, kan ses ved, at hele den nederste række lamper på det store kontrolbord lyser uden at YE - lampen lyser.
Da bufferen jo også anvendes til lagring af indicerede variable, kan det let blive et problem at få plads til det hele, Størrelsen af det oversatte program kan man finde ved at liste work lige efter at programmet er oversat, og antallet af indicerede variable kan man finde ud fra en programudskrift. Viser det sig at programmet ikke mangler meget i at kunne være i bufferen, vil det måske være ulejligheden værd at få det presset lidt mere sammen. I bedste (værste) fald kan programmet komme til at køre på en trediedel tid.

KONTROLUDSKRIFTER FRA ET KØRENDE ALGOL 4-PROGRAM

Som et experiment er der lavet en Algol 4-oversætter med navnet tr (med tilhørende plotter-variant trp), der i et opgivet linieinterval kan give udskrift af næsten alle assignments, der bliver udført under kørslen af et program.

Det ønskede linieinterval opgives som parametre kaldet af oversætteren ligesom ved kald af hjælpeprogrammet line, ønsker man udskrift af alle assignments i linierne 13 til 35 og 75 til 87 i programmet cha gøres det ved hjælp3-informationen:

```
cha, a, tr, 13..35, 75..87<
```

Metoden er denne: Under indlæsning af programmet vil oversætteren hver gang, den møder et begin eller et semikolon i det opgivne linieinterval begynde at opsamle de indlæste tegn i en lille intern buffer. Viser den påbegyndte sætning sig at være en simpel aritmetisk sætning, bliver hele den tekst der udgjorde venstresiden, gemt og benyttet i et kald af proceduren writetr, der automatisk bliver kopieret ind i starten af programmet. De to sætninger

```
i:=1;
A[i+5]:=1.0;
```

vil komme ud af pass 1 således:

```
i:=1; writetr (boolean i, ‡< i ‡);
A[i+5]:=1.0; writetr (boolean A[i+5], ‡< A[i+5]‡);
```

og under kørslen vil der komme følgende udskrift på printeren:

```
          1 ⇒ i
1.000000 ⇒ A[i+5]
```

Da hele mekanismen skulle lægges ind i pass 1, hvor der er meget lidt plads har det været nødvendigt at fjerne en del sjældent benyttede mekanismer fra denne passage:

1. Alle fejludskrifter er fjernet
2. Punch off og Punch on er blinde
3. Endcode, Clearcode og Sumcode er blinde
4. message behandles som comment
5. Fede kommaer er ikke tilladt.

6. Ved input fra skrivemaskinen kan man ikke rette en linie ved at taste fire case-skift.

Derudover medfører den anvendte mekanisme følgende restriktioner:

1. Sætninger umiddelbart efter then, else eller do vil ikke blive "fanget".
2. Hvis der på venstre side af := findes et procedurekald, vil denne procedure blive kaldt en extra gang. Der må derfor ikke være sætninger af typen A[readinteger]:=.....; i det specificerede linieinterval.
3. Der må ikke ske et assignment til procedurenavnet i en typeprocedure.
4. Hvis mekanismen anvendes i et case statement, hvor der findes simple statements, vil de indsatte kald af writetr få maskinen til at tælle galt.

På trods af de nævnte begrænsninger vil den beskrevne metode i en del tilfælde være den hurtigste vej til at finde f. eks. en indexfejl eller et spill. På den anden side må man ikke udelade sine egne kontroludskrifter, da erfaringen viser, at tracing af et program giver så store outputmængder, at man let drukner i overflødig information.

run
u

Disse to programmer er helt identiske. De undersøger om der findes et oversat program på work, er dette tilfældet, startes programmet, ellers kommer fejludskriften 'not present'. udover at starte programmet kommer der også en overskrift på printeren, der angiver dato og sted samt en eventuel meddelelse til brugerne.

y

er helt identisk med run og u, bortset fra overskriften, som ikke kommer.

Alle tre programmer udfører det samme som beskrevet under run i help3-manualen + følgende:
hvis der er en parameter til run ell. u ell. y, og denne parameter beskriver et buffermedium, vil området (med move) blive flyttet til work (work vil starte på spor 65 lige efter image), hvorefter kontrol overgives til det kaldte program. Work er naturligvis derved blevet ændret.

alarmudskrifter.

kind	hverken tromle eller bufferareal er kaldt.
undef	det kaldte område er ikke beskrevet i kataloget
not present	det kaldte område var ikke et oversat algolprogram

eks.
et oversat program, der er reserveret under navnet paxo på disken ønskes kørt. Der ønskes ikke overskrift på printeren, hjælp3 informationen er da:

y, paxo<

VIGTIGT: man må ikke trykke på hp medens run, u eller y arbejder.

edit

er et retteprogram. Det læser rettelser ind fra hjælps løbende inputmedium retter texten (som specielt kan være et algolprogram) i et nærmere angivet inputareal og lagrer den rettede text i et nærmere angivet outputareal, hvis intet angives som inputareal vælger edit strimmellæseren, og hvis intet angives som outputareal vælger edit perforatoren.
eksempel.

r, edit<

I dette eksempel læses rettelserne fra hjælps løbende inputmedium (r=strimmellæser). Den text der skal rettes læses også fra strimmellæseren, og den rettede text kommer ud på perforatoren.

free, edit, opax<

I dette eksempel læses rettelserne fra det frie areal på disken (free), den text der skal rettes læses fra strimmellæser, og den rettede text kommer til at ligge på et område kaldet 'pax'.

edit, ipax, ofree<

rettelserne læses fra hjælps løbende inputmedium, som når intet specificeres er skrivemaskinen. Den text der skal rettes befinder sig på området pax og den rettede text lægges ud på free.

RETTELSESLISTEN

Rettelseslisten har form af en liste af enkeltrettelser, som skal komme i den rækkefølge, hvori de forekommer i programmet.

En enkeltrettelse består af textstrenge adskildt af .

<kopier til og med> . < indsæt> . < slet til og med> .

listen af rettelser afsluttes med en stop code eller et å. Bemærk at cr og space er blinde symboler i strengene < kopier til og med > og < slet til og med >. Det tilrådes af hensyn til edit at afslutte sit program med en stop code.

eksempel:

rettelsesstrimmel:
```
edit,o free<
(¢ . <Jens.Jens.
STOP CODE
```

programstrimmel:
```
algol<
begin integer i,j;
...
select(8);
writetext(¢ Jens Olsens program ÞÞ);
i:= readinteger;
...
end t<      STOP CODE
```

øvelse. find hvorfra rettelserne læses.

Udover det her beskrevne anvendelsesområde, har edit flere andre nyttige funktioner. En beskrivelse af disse kan findes i hjælp3 manualen.

e1 er identisk med edit (e) bortset fra at mens edit vender tilbage til skrivemaskinen, vender e1 tilbage til løbende unput.

e2, a2, ok, nonok

hjælp3 giver brugerne mulighed for at tilrettelægge mere indviklede kørsler så at operatøren kun behøver at taste r< på skrivemaskinen. Derved har man opnået væsentlig sikkerhed mod operatørfejl, men i visse situationer har virkningen været den stik modsatte. et eksempel:
Man har lavet en rettelsesstrimmel, der skal foretage en helt enfoldig ret telse i et af ens områder på disken og lader derfor edit aflevere det rettede program på samme område som det tog input fra. Imidlertid har man lavet en lille fejl i sin rettelsesstrimmel, og rettelsen går galt, så at man har mistet hele sit program. Hvad man kunne ønske sig er en mulighed for først at rette programmet, derpå, hvis rettelserne gik godt oversætte det nye program, og derefter hvis oversættelsen gik godt, da flytte den nye version tilbage på det oprindelige område, dette kan opnås ved anvendelsen af e2, a2, ok, nonok.

e2 afviger fra edit på to punkter. For det første vender det ikke tilbage til skrivemaskinen efter at have udført rettelserne, men vender tilbage til løbende inputmedium (derved ligner det edit-varianten e1), for det andet slutter e2 med at sætte en intern boolean til true, hvis alle rettelser blev fundet og det rettede program blev afleveret i hel tilstand (dog ikke overlab) ellers sættes den interne boolean til false.

a2 er identisk med hjælpeprogrammet algol (a) udover, at a2 sætter omtalte boolean til sand, hvis oversættelsen gik godt ellers false.

ok og nonok anvendes herefter til at spørge på den interne boolean. ethvert hjælp3 kald kan indledes med enten ok eller nonok, og resten af hjælp3-kaldet vil da kun blive udført, hvis den interne boolean er sand henholdsvis falsk.
eksempel: se eksemplet under move og moveb.

outmt

formål: programmer eller andre textområder på internt medium der ønskes udskrevet på perforator lægges over på magnetbånd og udskrives over konverteren for at spare tid på gier.

kald: outmt,<input>,<numberlist><

<input> er navnet på det område der ønskes udskrevet.
<numberlist> ::= <empty> |<helpnumber>|
<helpnumber> , <helpnumber>

virkning af kaldet: texten på inputområdet lægges over på magnetbånd således:

<numberlist>	station no.	file no.
<empty>	2	0
i	2	i
i, j	i	j

eksempel.

outmt, free< skriver free på station no. 2 file no. 0
outmt, atm, 3< skriver atm på station no. 2 file no. 3
outmt, pax, 4, 1< skriver pax på station no. 4 file no. 1

udskrivning. ved udskrivning benyttes SB - PT katalogstrimlen.

fejludskrifter. udover de sædvanlige hjælp3 udskrifter kan følgende forekomme:

file protect — skrivning ikke tilladt på det monterede bånd.
illegal tape — der forsøges skrevet på et bånd med << cattap>> label, når et nyt bånd er monteret vil et SPACE på skrivemaskinen få programmet til at starte forfra.

programmet spoler selv båndet tilbage til loadpoint før skrivning, der sættes filemark efter hvert område der er lagt op på båndet.

NBNBNBNB man opfordres kraftigt til at benytte outmt, når man skal have textområder ud på puncher, istedet for edit, ofree<å eller lignende.

out

formål: at udskrive textområder på linieskriver. out vender tilbage til løbende inputmedium.

kald: out,< input><
<input>::=<drum> | <buffer medium>

eksempel.

out, free< teksten på free udskrives på linieskriver.
jum3, out, jum3< texten i jum3 udskrives på linieskriver, og derefter oversættes jum3 (jum3 har naturligvis de nødvendige help3 kald).

virkemåde. linieskriveren kan normalt modtage karaktererne lige så hurtigt som gier kan aflevere dem. En undtagelse er karakteren lige efter et linieskift. Ved at opbygge en hel linie ad gangen i gier, medens linieskriveren skriver den foregående linie, og dernæst hurtigt afleverer linien, er der sparet megen tid (dobbel buffer princippet).
ved skrivning af linier der har færre end 10 karakterer er edit og out lige hurtige, i alle andre tilfælde er out hurtigere.

line

formål. at udskrive et algolprogram med linienumre i udvalgte intervaller. input fra alle medier, output på ydre medier. Vender tilbage til løbende inputmedium, som ethvert andet hjælpeprogram. Udskrift i overensstemmelse med algols linienummerering.

kald. line,<input>,<linelist><
<input>::=<external>|<drum>|<buffermedium>|<empty>
<linelist>::=<helpnumber>|<helpnumber>,<linelist>|<empty>
<helpnumber> er af formen first line .. last line
hvis <input>=<empty>, vælges strimmellæser som input.
hvis <linelist>=<empty> afbrydes programmet og hjælp fortsætter på løbende input. hvis first line = last line, kan man nøjes med at skrive last line, når det ikke er første helpnumber.

eksempel.

line, free, 15..18<	skriver line 15 til line18 ud på løbende outputmedium af algolprogrammet på free.
w, line, jum3, 1000<	skriver de første 1000linier ud på skrivemaskinen af algolprogrammet på jum3.
l, jum3, line, jum3, 13..15, 20, 30, 100..125, 200<	dette kald skriver linierne 13-15, 20, 30, 100-125, 200 ud på linieskriveren, derefter kaldes jum3 for at oversætte (idet jum3 naturligvis begynder med a,n< eller lignende).

alarmudskrifter.

param	<linelist> er forkert
undef	<input> eksisterer ikke
text end	teksten er ikke så lang som texten fordrer.

bemærkning. hvis man benytter copy< i sit algolprogram, passer linienumrene ikke.

pæn

formål at udskrive algolprogrammer med understregninger og gennemstregninger sat som ved flexowriterudskrifter. pæn virker fuldstændig som out, men tager dobbelt så lang tid om en udskrift. Det henstilles at man inskrænker brugen af pæn til det absolut minimale.

alged
ae

tidligere kaldet dr.
Formålet med dette program er at kunne få en printerudskrift af et algol-program, hvoraf begin - end strukturen tydeligt fremgår. Da det er ønskeligt at udskriften leveres hurtigst muligt, antager hjælpeprogrammet at algolprogrammet opfylder følgende:

1. der forekommer ikke begin, end i strings, comments eller messages.

2. ingen casesymboler forekommer i beg (begyndelsen af begin) eller i end, dette medfører at man ved internt output fra edit, ikke må foretage rettelser - heller ikke tomme rettelser midt i disse strenge (f. eks. be... eller en...) men gerne f. eks. beg... eller gin...

3. ingen compound-fejl i begin og end

alle indrykninger der måtte forekomme i programmet i forvejen fjernes. dvs. hvis strings indeholder cr (vognretur) ødelægges opstillingen i programudskriften. Trods disse mangler og restriktioner har programmet vist sig særdeles anvendeligt, når man ønsker overblik over strukturen i et algolprogram, og ikke selv laver denne ved hulning af programmer og evt påfølgende rettelser

For at kunne udskriften til opsøgning af evt. syntaktiske fejl i programmet er udskriften forsynet med linienummerering, som dog kan udelades, hvis det ønskes.

Programmet har skrivebuffer i ferritlageret ligesom out, og er kun 5% i langsommere end dette.

kald:

ae,<input>,<form><
<input>::=<empty> | <area name>
<empty> : strimmellæser vælges som input
<area name> : skal beskrive et textområde eller et ly-medium.

<form>::=<empty> | - (understreget minus)
<empty> : linienumre printes.
- : ingen linienumre.

alged,r<
alged< input fra strimmellæseren, output med linienumre.

ae,-<
ae,r,-< input fra strimmellæseren, output uden linienumre.

alged,atm< input fra atm, output med linienumre.

alged,atm,-< input fra atm, output uden linienumre.

Som det ses af ovenstående, kan der ikke vælges outputmedium, hvilket naturligvis skyldes at man kun har brug indrykning på printeroutput, det ville jo være vanvittigt at have hundredevis af overflødige spacer på strimmel eller interne områder.

res

benyttes til at reservere et område af free. Det navn man vil tildele området, skal gives som parameter, og res vil begynde med at sikre sig, at navnet ikke allerede står i kataloget. (i så tilfælde får man fejludskriften name). Det anbefales at man bruger sine forbogstaver som de første karakterer i navnet og en kort identifikation som de sidste karakterer, Jørgen Mylius kunne således passende kalde sit første program for jmprg1. Hvis man følger denne regel kan man få megen nytte af et andet hjælpeprogram kaldet lis (omtales senere).
Har man indlæst sit program til free ved hjælp af edit, kan man få det reserveret under navnet jmprg2 ved kaldet:

res,jmprg2<

den del af free som programmet fylder, vil så blive reserveret under navnet jmprg2, under forudsætning af der ikke var et andet program med dette navn. Man kan nu regne med at området findes til næste morgen. Der findes dog en mulighed for at angive at man ønsker at bevare området til en bestemt dato. Hvis man f. eks. mener at skulle bruge sit område indtil juleaften 1971 ser kaldet således ud:

res,d24. 12. 71,jmprg2<

I forbindelse med denne datomærkning må man være opmærksom på 2 ting. For det første vil der altid være risiko for, at området alligevel bliver ødelagt, enten ved operatørfejl eller - mere sandsynligt - af rent tekniske årsager, og man må derfor altid sørge for, at man er i stand til at reetablere hvad man har stående på disken.
For det andet er pladsen ikke ubegrænset, så det vil ikke være god tone, hvis man blot ukritisk reserverer det ene område efter det andet i adskillige måneder.
Yderligere information kan findes i manualen.

redate

kald:
redate,< vilkårlig permutation af elementerne <form><værdi> og <navneliste>><
<form> se manualen under list
<værdi>::= <empty> | d<helpnumber>
<navneliste> må ikke være tom.

for hvert navn i parameterlisten sker følgende:

1. Hvis navnet refererer til et reserveret område, fjernes eventuelle forekomster af p, i og s bit i arealordet. Hvis der findes et sekundært ord indsættes seneste forekommende <værdi> heri, eller 0, hvis der ingen har været. Katalogindgangen skrives i overensstemmelse med seneste form, hvis der har været nogen.

2. hvis navnet er work, ændres <work-as-output> så det begynder med den kanal, hvis nr står i de sidste 9 bit af <helpnumber> i den seneste <værdi> hvis kanalnr. ligger udenfor området 39-319 indsættes dog 1. kanal for image. Udskrift som ovenfor

3. i andre tilfælde kommer udskrift som ovenfor efterfulgt af udskriften no change.

eksempler
hvis Jørgen Mylius finder ud af (se eksemplet under res) at han ikke skal bruge sit område længere end til 1/12-71 ser kaldet således ud:

redate,d1.12.71,jmprg2<

området vil da blive fjernet den 2/12-71 om morgenen.
kaldet
redate,d5.10.71,jmprg1,prg2,d23.12.71,jmprg3,jmprg4,jmprg5<
vil bevirke at jmprg1 og jmprg2 vil blive stående til 5/10-71 og områderne jmprg3, jmprg4 og jmprg5 til 23/12-71.

list
se manualen

lis
kald
lis,<parameterliste> <
hvor <parameterliste> er den samme som for hjælpeprogrammet list, med den begrænsning at intet navn må være længere end 6 bogstaver.

virkningen af et kald af lis er følgende:
For hvert navn i <parameterliste> udskrives alle de katalogindgange, hvori forekommer et navn, hvis første bogstaver er identiske med navnet i listen. udskriften sker i overensstemmelse med <form>.

eksempel. Hvis Jørgen Mylius (se res) havde fulgt råd det om at lade alle sine områder begynde med de samme bogstaver, ville han kunne have fået en fuldstændig listning af alle katalogindgange af alle sine områder ved kaldet:

lis,ajm<

move
kald:
move,<base>,<base>,<tracks><
<base>::=<area name> | b<helpnumber>
<tracks>::=<empty> | <number>

programmet flytter (kopierer) informationen i område1 til område2.
eksempel:

move,pax2,free<

kopierer informationen på pax2 til free og sætter samtidig booked i free.

move,free,pax2,15<
kopierer første spor af free ind i pax2, idet 15 betyder 15 spor á 40 ord

move,free,pax2<

vil almindeligvis give fejludskrift, men da det er et kald man ofte kan have brug for kan udføre dette ved hjælp af moveb.

moveb

kald:
moveb,<area name><
virkning. Programmet flytter den del af free der er booked til <area name>. på følgende måde: først udregnes hvor mange diskspor der er booked, og derefter kaldes move med de rigtige parametre.
eksempel.
antag at man med move eller edit har lagt information ud på free, og at denne information fylder 5 diskspor, booked vil da være sat til 5. denne information ønsker man nu at flytte ind på et område kaldet pax3, som også fylder 5 diskspor, moveb genererer nu kaldet:

move,free,pax3,75<

EKSEMPEL

vi er nu i stand til at give et eksempel på en veltilrettelagt kørsel.
Antag at en bruger har et algolprogram på et område kaldet alprg. Han ønsker nu at rette i dette program, oversætte den rettede version, og hvis alt er gået godt, at gemme det oversatte program under navnet alprgo. For at genere de øvrige brugere mindst mulig er kørslen tilrette således at der kun skal tastes r< på skrivemaskinen. Al anden information findes på strimmel:

move,alprg,free<	alprg kopieres til evt. senere brug.
e2,i,alprg,oalprg<	der rettes med kildetextindput fra alprg, og den
<rettelser>	rettede version lægges tilbage på alprg.
<STOPCODE>	rettelsesliste slut.
nonok,moveb,alprg<	hvis rettelsen ikke gik godt, kopieres free tilbage
nonok,t<	til alprg, og skrivemaskinen vælges som inputmedium.
l,algprog,a2<	det rettede program oversættes.
nonok,l,line,alprg,1000<	hvis oversætt. ikke gik godt, udskrives program-
nonok,t<	met med linienr. og skrivemaskinen vælges som hjælps løbende inputmedium.
move,work,free<	det oversatte program kopieres til free.
res,d1.10.71,alprgo<	det oversatte program reserveres til den 1.okt. 71.
t<	skrivemaskinen vælges som løbende input.

INPUT - OUTPUT MEDIER

t beskriver input fra skrivemaskinen ved gier.

w beskriver output på skrivemaskinen ved gier.

p beskriver output på perforator ved gier.

l beskriver output på printer (linieskriver).

lp beskriver output på både printer og perforator.

r beskriver input fra læseren ved gier. Stopper ikke på paritetsfejl.

r1 beskriver input fra læseren ved gier. Stopper på paritetsfejl.

r2 beskriver input fra læseren ved converteren, under forudsætning af at den er koblet til gier på omkoblingskassen. Stopper ikke på paritetsfejl.

f

hjælpe programmet laver så mange formularskift på printeren (via 6 på krydsfeltet), som parametren angiver.

f,2<	to formularskift
f<	et formularskift

dump

ved hjælp af dump, kan man afbryde et kørende program, for senere at gøre kørslen færdig. Visse betingelser skal dog være opfyldt. Disse er:

1. programmet skal være et algol4 program eller et slipprogram.
2. celle 1-9 må ikke røres(kun relevant for slip-programmer).
3. kanal 46-71 må ikke røres (dvs. et oversat program må højest fylde 248 tromlekanaler).
4. programmer, der anvender put og get skal behandles med omhu. Hvis et program anvender put og get direkte på free, er dumping ikke mulig, og da operatøren kan blive nødt til at dumpe en brugers program er det i brugerens egen interesse altid at reservere sig et område til brug for put og get. Da imidlertid put og get anvender absolutte adresser på disken, må man ikke flytte de benyttede områder. Et dumpet program, der benytter disken, kan således ikke i almindelig fortsættes, hvis der i mellemtiden har været benyttet compress eller clean. der er dog en mulighed for at klare dette problem, snak med datalogisk instituts medarbejdere om dette.

Hvis et program opfylder de nævnte 4 betingelser, kan det dumpes ved, at man trykker på hp (aldrig reset), og derefter taster f. eks.

dump,d3.11.71,alprgd<

Virkningen af dette kald af dump er følgende:
først gemmes bufferens indhold på bufim. dernæst undersøges, om det opgivne navn allerede findes i kataloget, er dette tilfældet, flyttes bufim, image og work til det angivne område, og der indsættes i kataloget en oplysning om works placering, hvis området ikke findse i forvejen, reserverer dump et passende stort område og fortsætter som ovenfor.

Hvis programmet ikke har læst færdigt fra strimmellæseren, når det bliver afbrudt, er man nødt til at kopiere de resterende data over på en ny strimmel ved hjælp af copy. Man kan nemlig ikke vide, hvor man skal starte den oprindelige datastrimmel, når programmet fortsættes.

Hvis et program benytter magnetbånd, skal man selv sørge for at båndet ikke røres før programmet fortsættes.

Et dumpet fortsættes ved hjælp af programmet cont.

cont

benyttes til at starte et dumpet program med.
kaldet er som følger:

cont,<navnet på det dumpede program><

følgende fejludskrifter kan komme i forbindelse med dump og cont:

no work spec.	man forsøger at dumpe over i et område, hvortil der ikke i kataloget er afsat plads til en beskrivelse af work. Man kan altså kun dumpe oven i et område der oprindelig er lavet af dump.
improper work spec.	man har sagt cont til et område der ikke indeholder et dumpet program.
overlap	man forsøger at dumpe oven i et område der er blevet for lille.

NB: Store algol programmer bør laves, så de kan dumpe sig selv, snak med datalogisk instituts medarbejdere .

saveb
exitb

de to programmer vil henholdsvis gemme bufferens indhold på diskområdet bufim og hente bufim frem i bufferen. De har ingen parametre.

Hovedformålet med programmerne er at muliggøre anvendelsen af hjælpe-programmer i forbindelse med et kørende algol4-program. Vil man under kørslen af et program se indholdet af bufferens celle 3000-4020 på integer-form kan dette gøres således:

1. tryk hp
2. tast saveb<
3. tast l,print,bufim,w 5,i,5.0.6.420<
4. tast exitb<
5. tast exit<

Et kørende algol 4-program kan benytte move til at flytte disk-området data over på båndområdet tape således:

```
code i;
2,44
hs 1
hv re1
t saveb;
0c
e1: hs 1
hv re2
t move;
t data;
t tape;
0c
e2: hs 1
hv re3
t exitb;
0c
e3: e;
```

MAGNETBÅNDADMINISTRATION.

I visse tilfælde ønsker en bruger at gemme meget store mængder af information eller at gemme information over et par måneder. Man kunne f. eks. tænke sig, at en bruger har et sæt standardprogrammer, der skal køres hyppigt i uændret form, eller at han ved hver kørsel skal have adgang til nogle meget store tabeller. I sådanne tilfælde vil et magnetbånd være meget mere pålideligt en disken.

udlevering af magnetbånd
Ved henvendelse til en af instituttets operatører kan en bruger få udleveret et bånd til eget brug. Ved samme lejlighed kan han få en instruktion i den manuelle betjening af båndstationerne.

beskrivelse af båndområder
Ligesom de reserverede områder skal båndområderne være beskrevet i kataloget. Dette sker ved hjælp af hjælpeprogrammet set, det kaldes således:

set, 3, 1, st. nr., file nr., 0, navn<

Her st. nr. nummeret på den båndstation man ønsker at benytte - normalt vil 2 være at foretrække. file nr. angiver, hvor mange områder ligger før det nyoprettede område, file nr. skal altså gennemløbe værdierne 0, 1, 2, 3 -----.

som man vil se er det en ret kompliceret sag at beskrive et båndområde, imidlertid er flere af parametrene faste, og da move og edit altid ajourfører længden af et båndområde, hvortil de har leveret output, og da femte parameter, der angiver hvor mange blokke der skal ovespringes, normalt skal være 0, og da file nr. normalt skal være 1 større end højeste tidligere benyttede filenr., findes der et hjælpeprogram med en betydelig lettere administration, der kan erstatte set. Dette program hedder stp.

stp

benyttes i stedet for set (se manualen) hvis ovenstående betingelser er opfyldt. Hvis der i kataloget er beskrevet områder til og med file nr. 4 på båndstation nr 3 , vil

stp, 3, peter<

være ækvivalent med

set, 3, 1, 5, 0, peter<

stp kan indsætte alle former for beskrivelser. Således kan en ga4-oversætter på magnetbånd laves ved:

stp, pis 0, tapega4, 11. 199. 199. 0<

idet stp benytter station 2, når andet ikke er angivet.

For at undgå at andres beskrivelser af båndområder får stp til at indsætte for store værdier af filenr. bør man indlede sin kørsel med et kald af outtp (se dette), og så siden benytte excl og incl.

cattap

før man kan benytte områder på et bånd, skal dette være forsynet med en cattap-label. dette gøres med hjælpeprogrammet cattap.

```
cattap,3<       skriver en label på station nr. 3
cattap<         skriver en label på station nr. 2
```

nb: der skal ikke trykkes på hp, som ved cattap fra strimmel.

outtp

outtp fjerner fra kataloget alle beskrivelser vedrørende den ønskede båndstation. man bør altid kalde outtp før en kørsel der involverer stp eller excl. De andre brugeres båndområder kan ellers gribe forstyrrende ind.

```
outtp,1<        fjerner beskrivelser vedrørende station nr. 1
outtp,<         fjerner beskrivelser vedrørende station nr. 2
```

excl

opsamler alle beskrivelser vedrørende den ønskede station, og gemmer dem bagerst (efter sidste beskrevne file) på båndet. Samtidig fjernes beskrivelserne fra kataloget.

```
excl,4<         arbejder med båndstation nr. 4
excl<           arbejder med båndstation nr. 2
```

incl

søger på den ønskede båndstation, til den møder en stump hjælp3-katalog, der er lavet af excl. Derpå gåes alle navne efter, og findes et navn ikke allerede i det egentlige hjælp3-katalog, bliver det indsat med sin beskrivelse, ellers bliver det skrevet på skrivemaskinen med rødt. er KA tændt, vil alle navnene blive skrevet ud med sort.

```
incl,3<         henter katalog på båndstation nr. 3
incl<           henter katalog på båndstation nr. 2
```

o

spoler båndet på den ønskede station tilbage til loadpoint.

```
o,1<            spoler station nr. 1 tilbage
o<              spoler station nr. 2 tilbage
```

eof

skriver et filemark på den ønskede båndstation.

```
eof,3<          skriver et filemark på station nr. 3
eof<            skriver et filemark på station nr. 2
```

Som det ses arbejder alle båndadministrationsprogrammerne på station nr. 2, når intet andet er angivet. For yderligere oplysninger om båndadministration se bind 3 af gier manualen, samt manualen om sambasystemet.

copymt

kald:

copymt,<helpnumber>,<wordlength>,<parity><

<helpnumber> ::=<inputst>.<inputfile>.<outputst>.<outputfile> | <empty>

hvis helpnumber er udeladt (<empty>) kopieres fra øjeblikkelig stilling på station til øjeblikkelig stilling på station nr. 2 hvis file =0 kopieres til eller fra øjeblikkelig stilling på input eller outputstation.

<wordlength>::=<empty> | s

s der kopieres med afkortet ordlængde. (default: normal ordlængde.)

<parity>::=<empty> | e

e derkopieres med lige paritet. (default: ulige paritet.)

formål:

at kopiere magnetbånd. Programmet kan endvidere benyttes når man ønsker at dumpe et program der benytter tapeinput.

udhop:

programmet hopper ud når en af følgende sitiuationer forekommer:

1. filemark på inputbånd.
2. ændret bloklængde på inputbånd.
3. end-of- tape på input eller outputbånd.

eksempler:

copymt <	kopiering fra station 1 til station 2 uden tilbagespoling af båndene. normal ordlængde. ulige paritet.
copymt, 2. 0. 3. 0, e<	kopiering fra station 2 til station 3, ingen tilbage spoling af båndene. normal ordlængde. lige paritet
copymt, 5. 2. 3. 1, s, e<	kopiering fra station 5 file 2 til station 3 file 1. afkortet ordlængde, lige paritet.

NB: hvis man ønsker at starte ved loadpoint må man benytte hjælpeprogrammet o.

EKSEMPLER PÅ ANVENDELSE AF MAGNETBÅND.

Antag at man har 3 algol4-programmer, der skal køres hyppigt uden ændringer. De oversatte programmer gemmes på bånd på station nr. 2 således:

```
outtp<
cattap<
stp, tp2prg1<
stp, tp2prg2<
stp, tp2prg3<
a1, prg1, n<
move, work, tp2prg1<
a1, prg2, n<
move, work, tp2prg2<
a1, prg3, n<
move, work, tp2prg3<
excl<
t<
```

ovenstående er naturligvis skrevet på strimmel, så at man kun behøver at taste r< på skrivemaskinen.
De oversatte programmer ligger nu på båndet, og på den bageste file findes en stump hjælp3-katalog, som beskriver områderne.
Når man senere vil køre f. eks. prg1 gør man følgende:

1. monterer båndet på station nr. 2.
2. taster incl<
3. taster y, tp2prg1<

Som tidligere nævnt, sker der det når man taster incl< at der søges på båndet på station nr. 2 efter en stump hjælp3-katalog, og når den er fundet kopieres den ind i det rigtige hjælp3-katalog. Da beskrivelsen imidlertid altid står begest på båndet (efter den sidste file) kan det tage nogen tid (2-10 min. afhængig hvor mange og hvor lange områderne er), at finde beskrivelsen af båndområderne, dette kan imidlertid undgås hvis følgende er opfyldt:

1. alle programnavne starter med de samme bogstaver.
2. ingen andre programmer i kataloget starter med disse bogstaver.

man kan da benytte hjælpeprogrammet lis:

```
lis, rtp2<
outparam, t<        ;for at undgå at gier hænger på læseren.
```

genererer da en strimmel, der indeholder de til områderne hørende beskrivelser, i form af kald af hjælpeprogrammet set. Denne strimmel gemmer man sammen med båndet, og når man så skal bruge sit program, gøres følgende:

1. monterer båndet på station nr. 2
2. indsætter strimlen i læseren og taster r<.
3. taster y, tp2prg1<

man sparer derved 2-10 minutter.

Advarsel.

Hver gang man flytter information til et båndområde, skal man huske at det område der står lige efter også vil blive ødelagt. Dette hænger sammen med at f. eks. en bloklængde på 200 ord ikke behøver at fylde lige mange centimeter bånd, hvis den lægges op to gange, det afhænger ganske af den hastighed hvormed båndet paserer skrivehovedet. Hvis man således også gemmer uoversatte programmer på bånd, må man være klar over at det eneste område man kan rette i uden risiko er det sidst beskrevne område, idet man så kun får ødelagt kaltloget (på båndet), men det reetableres jo igen når man taster excl<.

newkit

Formålet med dette hjælpeprogram er at sikre, at det hjælp3-katalog, der står på tromlen, virkelig er det der svarer til det disk-kit, der er monteret.

ved kald af newkit sker følgende:

1. kataloget gemmes på spor 116 og 117 på disken.

2. hjælpeprogrammerne på spor 1, 5, 9, 13, ... 181 forskydes cyklisk 4 spor.

3. hp-knappen spærres og følgende udskrift kommer:
monter det nye disk-kit, rør ikke hp-knappen.

4. programmet går ind i en løkke, hvor det dels afspiller pausesignalet fra program 1 (radioen), og dels tester på, om der er blevet skiftet kit. Når dette er sket udføres:

5. hjælpeprogrammerne flyttes på plads.

6. spor 116 og 117 på disken flyttes over på katalogets plads på tromlen.

7. programmet går videre i parameterlisten, og udfører den hjælp3-information der eventuelt måtte forekomme.
eksempel:
en bruger har på sit private kit liggende et dumpet program under navnet paxd, som han ønsker at fortsætte efter at der er skiftet kit, dette sker ved kaldet:

newkit, cont, paxd<

Takket være flytningen af hjælpeprogrammerne vil man, hvis man blot skifter kit uden at kalde newkit, ikke kunne kalde bl. a. edit, move og algoloversætteren. Det er således vanskeligt at komme til at køre, hvis katalog og disk ikke passer sammen.

NB
hvis et eller andet går galt under udskiftningen af diskkit, må man for guds skyld ikke foretage sig andet end at tilkalde en af datalogisk institut's medarbejdere.

<u>parametre til newkit:</u>
som parameter til newkit kan gives et helt hjælp3 kald, som så vil blive udført når der er skifter disk.

newkit, move, work1, free, 10< bevirker at man efter at have skiftet diskkit får udført kaldet af move.

newkit, r< bevirker at strimmellæseren er valgt som løbende input, når der er skiftet disk.

BETJENING AF HULKORTLÆSEREN.

1. tryk på MAIN POWER. MAIN POWER skal da lyse - hvis ikke er en af kontakterne bag den venstre låge faldet ud, og skal tændes igen. Desuden skal ON LINE og AUTO lyse - hvis ikke, så tryk på dem.

2. slå kortlæseren og rc2000 til gier på omkoblingskassen.

3. luk lågen på converterens læser, tryk strimmelstyret ned (det bliver nede) og tryk på reset.

4. tryk på ready på kortlæseren, denne vil nu læse et kort, og maskinen er klar.

5. Går kortlæseren i stå under kørslen, har den forsøgt at læse flere kort samtidig. Træk da kortene fri og tryk på ready. Eventuelt kan en af kontakterne bag venstre låge atter være faldet ud.

hvis en anden bruger i forvejen har indlæst sit katalog, kan han godt samtidig benytte konverteren til arbejde, der ikke kræver strimmellæser. (plotning, udskrivning af bånd på printeren etc.).

Specielle fejludskrifter:

SUM (med rødt og stort).

mainhelp er ødelagt. Indlæs hjælp3 disk system (findes på strimmelbakken eller i nederste skuffe under læseren) med et space. Hvis hp-udskriften ikke kommer nu tilkaldes en systemoperatør.

catalog (med rødt og lille)

kataloget er ødelagt: Indlæs hjælp3 disksystem med ≤
hvis hp-udskriften ikke kommer, tilkaldes en systemoperatør.

sum (med rødt og lille)
man har prøvet at kalde et program med sumcheck, summen er imidlertid forkert (fordi programmet er ødelagt), hvis det kaldte program er enten saveb, dump eller exit, indlæses hjælp3 disk system. I alle andre tilfælde tilkaldes en systemoperatør.

image
hvis udskriften kommer efter kørsel af et algolprogram, betyder det at det oversatte program er så stort at det når ind i image (kanal 46-71 på tromlen). der tastes et space og situationen er normal.
advanced users information: se a manual of help3.

hvis systemet iøvrigt gør alvorlige knuder tilkaldes i arbejdstiden en operatør, ellers gøres følgende.

1. På øverste hylde ved siden af højre båndstation udvælges omhyggeligt det store bånd der har den ældste dato.
2. dette bånd monteres på en magnetbåndstation, der tildeles nummeret 1.
3. den valgte station tilkobles gier på omkoblingskassen.
4. på strimmelbakken eller i nederste skuffe under læseren, findse en strimmel med navnet: hent eller gem. denne strimmel indlæses med ≤
5. når programmet spørger hent eller gem tastes g.
6. når gem er færdig afmonteres båndet, og det afleveres hurtigst muligt til en operatør, sammen med en fyldestgørende beskrivelse af uheldet, samt bilag i form af de sidste sider skrivemaskine-output.
7. på hylden beskrevet under pkt. 1. findes nu det bånd med den nyeste dato.
8. dette bånd monteres på den nys brugte station, og hent eller gem indlæses.
9. når programmet spørger hent eller gem tastes h.
10. når programmet er færdigt kan SUM eller catalogudskriften forekomme
11. hvis systemet ikke er i orden nu,, pakker man sin taske sammen og efterlader en besked til næste bruger om at systemet er nede.