NAS Z80 BUS NYT

UDGIVET AF Z80 BRUGERGRUPPEN 5. ARGANG NR. 5

MAJ 1984

Fra brugerundersøgelsen har der lydt en kraftig opfordring til at beskæftige os mere med hardware og andre maskiner end Nascom og CP/M. Dog er der en meget (endog meget) stor gruppe, der vil høre om CP/M! Ligeledes efterlyses anmeldelser om software, enten erhvervet gennem vores programbibliotek eller fra forskellig softwarehuse. Derfor vil jeg endnu engang opfordre jer til at finde jeres hardwarekonstruktioner frem af skufferne og sende dem til os, så de kan komme i bladet. Anmeldelser modtages også meget gerne.

INDHOLD:

ALMINDELIGE OPLYSNINGER OM FORENINGEN

HENVENDELSE TIL FORENINGEN TIL FORRETNINGSFØREREN:

I. SKAVIN
Broholms alle 3
2920 Charlottenlund
Telefon 01 - 64 03 14

Hertil skal rettes henvendelse om indmeldelse, adresseforandring, salg af foreningens materialer (bånd, blade og programmer). Øvrige henvendelser af generel art til formanden. Stof og annoncer til foreningens blad sendes til Asbjørn Lind.

Indmeldelsesgebyr:	25.00 kr.
Kontingent 1.1.84 - 1.7.84.	60.00 kr.

Annoncering for medlemmer er gratis i Z80 NYT. For andre 250 kr. pr. A4 side.

Bestyrelsesmedlemmer:

Formand:
René Hansen
Bispevangen 6,13,th
2750 Ballerup
Tlf. 02 65 59 76.
Kl. 18.30 - 21.00

Næstformand:
Jesper Skavin
Broholms Alle 3
2920 Charlottenlund
Tlf. 01 64 03 14.

Ansvarsh. red.:
Ole Hasselbalch
Vibeskrænten 9
2750 Ballerup
Tlf. 02 97 70 13.

Frank Damgaard
Kastebjergvej 26A
2750 Ballerup

Per Thomsen
Ulspilager 75
2791 Dragør

Redaktør for Z80 NYT:
Asbjørn Lind
Sidevolden 23
2730 Herlev
Tlf. 02 91 71 82. (20.00 - 21.00)

Sidste frist for indlevering af stoft til næste nummer:
8.6.1984

7. maj 1984

Z80 BRUGERGRUPPEN
BROHOLMS ALLE 3.
2920 CHARLOTTENLUND.

Referat af generalforsamlingen søndag den 29 april 1984,
Svishaveskolen Blok B
Totten
5330 Munkebo

DAGSORDEN:
1. Valg af dirigent.
1a. Valg af referent.
2. Formandens beretning.
3. Fremlæggelse af regnskab for 1983.
4. Indkomne forslag.
5. Fastsættelse af kontigent for det kommende år.
6. Valg af bestyrelse og 2 suppl., revisor og revisorsppleant.
7. Eventuelt.

1) Der var i strålende solskin (vejret var vist for godt) fremmødt 10 deltagere til generalforsamlingen, heraf ca. 10% af medlemmerne på Fyn samt en fra Jylland. Til dirigent valgtes medlem nr 203 F.Mann Jensen.

1a) Frank Damgaard valgtes til referent (medlem nr 13).

2) Formandens beretning blev godkent uden indsigelser.

3) Jesper Skavin oplyste at salg af CP/M-programmer hørte under punktet "salg af progr., bånd, og Z80Nyt". Regnskabet blev godkendt enstemmigt.

4) Ingen indkomne forslag fra medlemerne.

5) Næstformanden (Jesper Skavin) redegjorde for enkelte punkter i regnskabet:
Kontingentet er beregnet ud fra ca. 450 medlemmer og ca. 100 nye medlemmer. Kontingentforhøjelsen skyldes delvist stigende porto og forsendelsesomkostinger, og øgede omkostninger ved trykning af bladet. På grund af overgang fra kassettebånd i kansas-city interface til de bedre FSK-modem, er klubben ved at udvikle et færdigt FSK-print (til max 300kr), heraf fremkommer posterne for køb og salg af FSK-modem i budgettet. Rente og udbytte er et overslag, og kommer fra to konti og en aktie i Handelsbanken (hvorved der opnås højere rente på aktionærkontoen).

Foranlediget af et spørgsmål fra et medlem, der gerne ville vide mere om databasen, svarede bestyrelsen, at der endnu ikke er nogle konkrete planer med hensyn til anlæg, placering m.v. Der vil tidligst til efteråret fremkomme endelige planer, når prisen på modems er mere afklaret. Dette sker i forbindelse med P&T's frigivelse lavhastigheds modem (bla. 300 baud), pr. 1. juli 1984. Det afsatte beløb i budgettet er dog et maksimumsbeløb, som ikke nødvendigvis vil blive brugt fuldt ud. Der har dog været tanker om måske at låne eller leje et anlæg i

en forsøgsperiode. Til en begyndelse var det ideen, at databasen skal benyttes til meddelelser mellem medlemmerne (køb,salg, etc.), og meddelelser til og fra klubben. Senere skulle databasen også kunne bruges i forbindelse med programbiblioteket, f.eks. bestilling af bånd , programmer, og søgning i programbiblioteket.

Fra et af medlemmerne fremkom den ide at medlemmerne rundt om i landet kunne finde sammen i mindre grupper og være fælles om et modem, sådan at udgifterne til dette ikke skulle blive for store, for den enkelte.

Budgettet blev enstemmigt godkendt, og kontingentent fastsat til 150 kr for et år.

6) Næstformanden, Jesper Skavin, og bestyrelsesmedlemmerne, Ole Hasselbalch og Frank Damgaard blev enstemmigt genvalgt.

Til 1. suppleant til bestyrelsen blev Erik Søe, medlem nr 445, valgt.

Til 2. suppleant blev Peer From, medlem nr 12, genvalgt (havde skriftligt givet besked til bestyrelsen at han var villig til genvalg, da han var forhindret i at komme).

Til revisor valgtes Cai Chritiansen, medlem nr 104, der ligeledes skriftligt havde givet til kende (idet han ikke kunne være tilstede), at han var villig til valg.

Til revisorsuppleant genvalgtes Christian Lausten, medlem nr 54, der skriftligt havde givet tilkende, at han var villig til genvalg.

7) Eventuelt.

På bestyrelsens vegne takkede næstformanden vores afgående revisor, Jan Jakobsen, for hans store indsats hvert år, når regnskabet skulle revideres. Jan har været med siden starten i 1980 og som tak herfor, vil Jan modtage seks flasker vin på bopælen, idet han ikke var til stede ved generalforsamlingen.

Ligeledes takkede bestyrelsen Peter Villadsen for hans initiativ og undervisning ved Pascal-studiegruppen i vinterhalvåret. Peter vil modtage en erkentlighed for indsatsen.

Dirigenten syntes, at det var en god ide at afholde mødet på Fyn. Desuden mente han, at der ville være et større fremmøde, hvis generalforsamlingen blev afholdt i forbindelse med et fordrag, fremvisning af maskiner eller lignende.

I den forbindelse fremkom den IDE, at der måske skulle oprettes et mobilt programbibiliotek til medlemsmøder og udstillinger ude i landet, så medlemmerne kunne se programmerne køre, og evt. samtidig også have mulighed for at købe programmerne. Måske kunne biblioteket også udlægges til lokalgrupper, hvis sådanne skulle blive dannet.

Dirigenten afsluttede generalforsamlingen og takkede de fremmødte for god ro og orden.

Frank Damgaard

SIMULERING

Her følger nogle få kommentarer til programmet. Hovedpunkterne er: Ankomst, Kødannelse, Udkørsel p.gr.a. ventetid, Udkørsel, Subrutiner og Udskrift. Bilernes intervaller og betjeningstid beregnes i subrutinen, og formlen for normalfordeling i nr.3 var forkert! Den giver absolut ingen normalfordeling, viser det sig ved nærmere afprøvning, og den er erstattet af Box-Muller-metoden:

```
SQR(-2*LOG(RND(1)))*COS(3.14159*RND(1))
```

hvor COS er i radianer, og LOG er 10-er-logaritmen, som kan erstattes med LN(RND(1))/LN(10). Ved ankomst forsynes bilerne med nr. N. Ved kødannelse er problemet at finde den bil, som har den korteste kø efter sig (mon ikke man ville vælge den?), og udregne den nye udkørselstid for bilen - efter ventetiden. Hvis ventetiden bliver for stor, kører bilisten. Ved udkørsel slettes bilen, og en evt. kø efter bilen tælles ned. Tidsberegningen starter kl.8 om morgenen og angiver klokken. Udskriften undervejs fortæller, når en bil kommer, kødannelse, udkørsel m.v. Udskriften til sidst samler oplysninger i alt og i gennemsnit.

Venligst Henrik Dyhr (006)

```
100 CLS
110 PRINT"     Simulering af kø ved benzintank"
120 REM                  H.Dyhr april 84
130 CLEAR 2000
140 DIM AN(500),UD(500),K(500),B(500)
150 S=4:REM Antal benzinstandere
160 TA=2:REM Antal minutters tålmodighed!
170 ST=72 :REM Minutters simuleringstid
180 REM ***********************************
190 REM Dan 1. tider
200 N=1
210 GOSUB 850
220 Z=AN(N):GOSUB 940
230 PRINT"Ankomst bil nr."N"kl."TI$
240 PL=PL+1
250 REM ***********************************
260 REM Dan næste tid
270 N=N+1
280 GOSUB 850
290 FOR T=1 TO N:IF AN(T)=0 THEN NEXT T
300 IF UD(T)<=AN(N) THEN GOTO 320
310 GOTO 380
320 MIN=ST
330 FOR T1=T TO N:IF AN(T1)=0 THEN NEXT T1
340 IF UD(T1)<MIN THEN T2=T1
350 IF UD(T1)<MIN THEN MIN=UD(T1)
360 NEXT T1
370 GOTO 760:REM UD(T2)=første udkørsel
380 NEXT T
390 REM ***********************************
400 REM Her ankomst:
410 Z=AN(N):GOSUB 940
420 PRINT "Ankomst bil nr."N"kl."TI$
430 PL=PL+1:PRINT "På pladsen er"N-AK"biler."
440 IF N-AK>S THEN GOTO 470
450 GOTO 260
460 REM ***********************************
```

```
470 KØ=KØ+1:PRINT "Kø = "KØ
480 IF MK<KØ THEN MK=KØ
490 REM Find første bil,der bliver færdig,
500 REM og som ikke har kø efter sig.
510 REM Brugt til kø: K(N)=1
520 MIN=ST
530 FOR T1=N-(PL-1) TO N-1
540 IF K(T1)=1 GOTO 570
550 IF UD(T1)<MIN THEN T2=T1
560 IF UD(T1)<MIN THEN MIN=UD(T1)
570 NEXT T1
580 Z=UD(T2):GOSUB 940
590 PRINT"(Kø efter bil"T2"betjenes kl."TI$")"
600 IF UD(T2)-AN(N)>TA GOTO 660
610 SK=SK+(UD(T2)-AN(N)):REM Sum af køtider
620 UD(N)=UD(T2)+B(N):REM Ny udkørselstid
630 K(T2)=1:REM Kø dannet efter denne bil
640 GOTO 260
650 REM *********************************
660 REM Her udkørsel p.gr.a. ventetid
670 UD(N)=AN(N)+TA:SK=SK+TA:AK=AK+1
680 Z=UD(N):GOSUB 940
690 PRINT"(Bil nr."N":"
700 PRINT"Udkørsel p.gr.a. ventetid kl."TI$
710 SV=SV+1:PRINT"Sum kørte p.gr.a. kø:"SV")"
720 KØ=KØ-1
730 AN(N)=0:K(N)=1:REM Bil annulleret(også kø)
740 GOTO 260
750 REM *********************************
760 REM Her udkørsel:
770 Z=UD(T2):GOSUB 940
780 PRINT "Udkørende bil nr."T2"kl."TI$
790 SB=SB+1:REM Sum af betjente biler
800 PL=PL-1:AK=AK+1
810 AN(T2)=0:K(T2)=1:REM Bil annull.(også kø)
820 IF KØ>0 AND K(T2)=1 THEN KØ=KØ-1
830 GOTO 290
840 REM *** Subrutine Intervaller ***
850 I=(INT(-2.5*LOG(RND(1))*100))/100
860 TI=TI+I:AN(N)=TI:IF TI>=ST GOTO 1000
870 B(N)=SQR(-2*LOG(RND(1))/LOG(10))
880 B(N)=B(N)*COS(3.14159*RND(1))
890 B(N)=B(N)*1.3+4.5
900 B(N)=(INT(B(N)*100))/100
910 UD(N)=AN(N)+B(N)
920 RETURN
930 REM *** Subrutine Tidsomregning ***
940 H=8+INT(Z/60):REM Start kl.8 om morgenen
950 M=INT(Z-(INT(Z/60)*60))
960 SE=INT((Z-INT(Z))*60+.5)
970 TI$=STR$(H)+"."+STR$(M)+"."+STR$(SE)
980 RETURN
990 REM *********************************
1000 PRINT "Der er betjent"SB"biler i alt."
1010 PRINT SV"bilister er kørt p.gr.a. ventetiden."
1020 SK=INT(SK*100)/100
1030 PRINT "Bilisterne har holdt i kø i alt"SK"min."
1040 GS=INT(SK/N*100)/100
1050 PRINT "I gennemsnit"GS" min."
1060 PRINT "Kølængde max.:"MK
```

Hvordan begynder man på PASCAL ?

Arne har været så venlig at videregive os sine ideer og tanker fra den første gang, han skulle i gang med Pascal. Arne skriver dette til en god ven, der ligeledes skal i gang. Vi kigger dem over skulderen og ser, hvordan det hele udvikler sig.

Det skulle helst blive til et par sider i hvert af de kommende numre - er det lovet.

Red.

Vi starter med første linie i program-hovedet og kalder det plus, jeg vil senere lave det om til en PROCEDURE, som kan kaldes fra et større program med flere udregningsmuligheder. Men først skriv.

```
PROGRAM PLUS;
```

Vi skal have fundet ud af, hvad variablene A,B og C skal være, da alle variable, der anvendes i et program, skal erklæres i VAR-erklæringsdel. REAL = Tal med komma og INTEGER = Heltal. Vi vælger heltal.

```
VAR A,B,C:INTEGER;
```

Så til selve programmet, vi starter med BEGIN, der angiver, hvor programmets sætninger begynder og til sidst END., hvor programmet slutter.

```
BEGIN
```

Vi skal have et tilfældigt tal, som vi kalder A og B, vi bruger RANDOM, dette samme som RND i Basic. Imellem to order skriver vi ':' i Basic, men i Pascal ';' også ved linieskift (som jo også er en adskildelse mellem to ordre).

```
A:=RANDOM(9); B:=RANDOM(9);
```

I næste linie skal vi have udskrevet A og B. WRITE står for print (uden linie skift) og WRITELN for print med linie skift, " bliver til (for begynd og) for slut. Da vi skal have input på samme linie, tage vi WRITE.

```
WRITE(' Hvad bliver ',A,' + ',B,' = ');
```

Input C skrives som READ(C), og da der skal være linieskift er det ligesom ved printningen.

```
READLN(C);
```

Talvariabel skal defineres, i Basic vil der stå C=B+A, næsten det samme her.

```
C:=B+A;  (C tildeles værdien af summen af A og B)
```

For at se om vi nu kan lægge to tal rigtigt sammen, skal vi have den til at fortælle os om det, så bruger vi IF...THEN...ELSE. Hvis ikke C og D er ens, skrives FORKERT ellers skrives RIGTIGT. Det gør vi ved at skrive.

```
 IF C<>D THEN  WRITELN(' FORKERT ')
         ELSE  WRITE(' RIGTIGT ');
```

Efter THEN kan jeg bruge BEGIN..END. Hvis jeg vil have udført mere en en sætning. Se senere, hvor jeg har tilføjet to funktioner, (SCREEN,WRITE). Til sidst skal vi skrive END.. Husk punktum. Programmet skrevet i Basic.

```
10 A=INT(RND(1)*12):B=INT(RND(1)*12)
20 PRINT" Hvad bliver "A" + "B" = ";
30 C=B+A
40 INPUT D
50 IF C<>D THEN PRINT" FORKERT"
60 IF C=D THEN PRINT" RIGTIGT"
50 END
```

Ved at tilføje GOTO kan vi få den til at stille det samme spørgsmål, indtil det er besvaret rigtigt, det gøres ved at skrive en LABEL-erklæring, med et heltal. Placer heltalet før spørgsmålet og placer GOTO med heltal efter FORKERT. Heltalet kalder vi 10. Det færdig program se nu sådan ud. Jeg har dog tilføje WRITE(CHR(12)), SCREEN(x,y), (se under overskrift), og FOR...TO...DO, i Basic FOR...TO...NEXT.

```
PROGRAM PLUS;
  LABEL 10;
  VAR   A,B,C:INTEGER;
BEGIN
  A:=RANDOM(9);B:=RANDOM(9);
  WRITE(CHR(12));
10:
  SCREEN(22,5);WRITE('                    ');
  SCREEN(7,5);WRITE('Hvad bliver ',A,' + ',B,' = ');
  C:=B+A;
  READ(D);
  IF C<>D THEN
            BEGIN SCREEN(13,10);
            WRITE('*** FORKERT ***');
            GOTO 10;
            END
          ELSE
            BEGIN SCREEN(13,10);
            WRITE('>>> Rigtigt <<<');
            END;
  FOR I:=1 TO 30000 DO
END.
```

Vi lave nu et underprogram med en PROCEDURE, som vi kalder frem fra program-hovedet, vi clearer skærmen med ASCII 12 (FormFeed), og sætter den vejledende tekst til brugen op på skræmen ved hjælp at SCREEN(x,y), som er den samme som i Basic.

```
PROCEDURE OVERSKRIFT;
BEGIN
  WRITE(CHR(12));
  SCREEN(5,2);WRITE('Dette er et regneprogram du har');
  SCREEN(5,3);WRITE('der er 4 muligheder at vælge imellem.');
  SCREEN(10,5);WRITE('1. Plus.');
  SCREEN(10,6);WRITE('2. Minus.');
  SCREEN(10,7);WRITE('3. Gange.');
  SCREEN(10,8);WRITE('4. Dividere.');
  SCREEN(10,9);WRITE('>>ESC<< For at slut.');
  SCREEN(33,0);
  WRITE(CHR(9),'  ');
END; (* for slut på underprogrammet.*)
```

Og nu til selve HOVED-PROGRAMMET, hvor vi begynder med BEGIN, og da vi skal have mulighed for at prøve alle 4 spørgsmål, bruger vi REPEAT...UNTIL. (En løkkestruktur indtil udtrykket er sandt). I denne løkke skal vi have lagt WRITE(CHR(12));.
Da vi skal se, hvad der står i procedure OVERSKRIFT, kalder vi den frem ved at skrive OVERSKRIFT;.
Nu skal der lægges et input ind, så vi kan vægle mellem 1 til 3. Med READ(SVAR), svar er en strengvariabel, der skal erklæres i VAR-delen. READ(SVAR) sætter vi imellem en ny REPEAT...UNTIL. Og går først videre, når der er blevet svaret med 1 til 3 eller >ESC<.
(ORD er det samme som ASC i Basic.)

```
  REPEAT
    READ(SVAR);
  UNTIL ((SVAR<='3') AND (SVAR>='1')) OR (SVAR=CHR(27));
```

For ikke at bruge IF..THEN..ELSE tre gange ved svar. Bruger vi CASE..OF..END.
Casen skal være lig med svar.

```
      OPGAVE:=ORD(SVAR)-ORD('0');
      CASE OPGAVE OF
        1 : PLUS;
        2 : MINUS;
        3 : GANGE;
      END;
```

Nu skrives den sidste del af løkken, UNTIL OPGAVE=27-ORG('0'); for at komme ud af programmet.
Og tilsidst END. Husk punktum.
Mellem den sidste løkke og END. kan du skrive en sluttekst. Som bliver vist, når man stopper programmet.
Nu kan du lave programmet PLUS om til en PROCEDURE PLUS; der nu kan kaldes frem nede i CASE.
Prøv nu at lave PROCEDURE MINUS; og PROCEDURE GANGE; det du skal lave om er C:=A+B; til - og *, let ik'?, dividere har jeg taget med, men da jeg har brugt TRUNC, springer jeg det over, men det kommer på et senere tidspunkt.
RANDOM kan laves om til en procedure eller lægges i hovedprogrammet, og f.eks. WRITE(CHR(12)) til en procedure du kalder CLS.
Du skal bare have hovedprogrammet til aller sidst.
Ved at bruge PLOT x,y,type. Sætter og resetter eller inverterer punktet x,y. Dermed kan vi lave en ramme om overskriften og rigtigt/forkert i en procedure og kalde frem.

```
      (* rigtigt/forkert ramme *)
      PROCEDURE KANT;
        VAR I,J:INTEGER;
      BEGIN
        WRITE(CHR(12));
        FOR I:=21 TO 60 DO BEGIN PLOT(I,28,1);PLOT(I,33,1);END;
        FOR J:=29 TO 32 DO BEGIN PLOT(21,J,1);PLOT(60,J,1);END;
      END;
```

Placer KANT; lige efter BEGIN i procedure PLUS.

```
      (* Ramme om overskrift *)
      PROCEDURE CANT;
      BEGIN
        WRITE(CHR(12));
        FOR I:=5 TO 76 DO BEGIN PLOT(I,4,1);PLOT(I,38,1);END;
        FOR J:=5 TO 38 DO BEGIN PLOT(5,J,1);PLOT(76,J,1);END;
      END;
```

Du skal placere CANT i overskrift i stedet for write(chr(12)).

Arne (234).

Eksemplerne er skrevet i BLS-pascal til Nascom 2.

PIEZODAN aps

NASDICOS - nu også som diskoperativsystem

Nasdicos er en kombination af en kraftig udvidet Nas-Sys monitor og et styresystem til betjening af forskellige typer eksterne lagermedier som f.eks. tape (Phillips mini DCR), floppy-disk eller RAM-disk.

Du kan starte med at købe et system til Phillips mini DCR og senere udvide til disk. Programmer udviklet til DCR vil uden ændring kunne køre på disk-versionen.

Systemet optager ikke plads i RAM-lageret, hvilket er opnået ved at placere det på en særlig systembank, der kun kobles ind i CPU'ens adresseringsområde, når en systemrutine kaldes. Som systembank bruges Nascom'ens otte RAM/EPROM-sokler, der efter en mindre ombygning vil kunne pages ud og ind. I soklerne anbringes enten 4k-EPROM eller 1k/2k statisk RAM (EPROM til tape-versionen og RAM til disk-versionen).

Af andre egenskaber ved systemet kan nævnes:

* Der findes rutiner til håndtering af sekventielle data-filer. Til disk-systemet vil senere komme rutiner til direkte data-filer.

* Der leveres med systemet en udvidelse af BASIC'en, så filsystemet i Nasdicos kan betjenes fra BASIC.

* Til systemet kan købes ændringer af NAP, PASCAL og Wordease så filsystemet også kan betjenes fra disse programmer.

Nasdicos er naturligvis helt Nas-Sys kompatibelt, det er fuldstændig interruptbart og eneste krav til dit system er, at det har har 64k RAM (til disk-versionen med paging).

Priser:	Tapeversionen	950,-
	Tape- og diskversionen	1400,-

Prisen inkluderer matrialer til konstruktion af paging. Til diskversionen må du selv have mindst 8k statisk RAM til at sætte ned i soklerne.

PIEZODAN aps
Bernhard Bangs alle 17A
2000 København F
Tlf: 01 86 12 17
Åbent mandag - fredag 11.00 - 17.30

EUROMICRO '84

I TILSLUTNING TIL EUROMICRO '84, DER AFHOLDES PÅ DTH I LYNGBY DEN 27. TIL 30. AUGUST, ARRANGERER EUROMICRO (THE EUROPEAN ASSICIATION FOR MICROPROCESSING AND MICROPROGRAMMING) IGEN 'MICROMOUSE' FINALER

Hvad er en mikromus?

Mikromusen er en lille selvstændig robot, der skal finde vej gennem en labyrint. Labyrinten er organiseret som 16 x 16 kvadrater i et net med 18 cm som basis. Væggene i labyrinten er 5 cm høje og 12 mm tykke. Mikromusen startes i labyrintens 'nederste venstre' kvadrat og skalsà selv finde vejen ind til midten af labyrinten. Den mus, der klarer det på kortest tid, er vinder. Da der er 15 minutter til rådighed, giver dette mikromusen mulighed for at lære den korteste vej.

'Micromouse Maze Contest' blev først lanceret af IEEE Spectrum og er så taget op af EUROMICRO i forbindelse med afholdensen af det årlige EUROMICRO symposium.

Madrid.

Sidste mesterskab blev afholdt i Madrid i 1983. Der deltog 12 mikromus i konkurrencen, og af disse kom 8 i finalen. Der var stor variation i udformningen af de mikromus, der stillede op - både tekninsk og i kvalitetsmæssig udførelse - ligesom der var stor spredning i deltagernes tekninske baggrund. Det mest bemærkelsesværdige i denne forbindelse var nok, at et engelsk hold skoledrenge i 12 års alderen stillede op med deres egen konstruktion, omfattende både mekanik, mikrodAtamat og programmel. Det lykkedes dog ikke for dem at kvalificere sig til finalen.

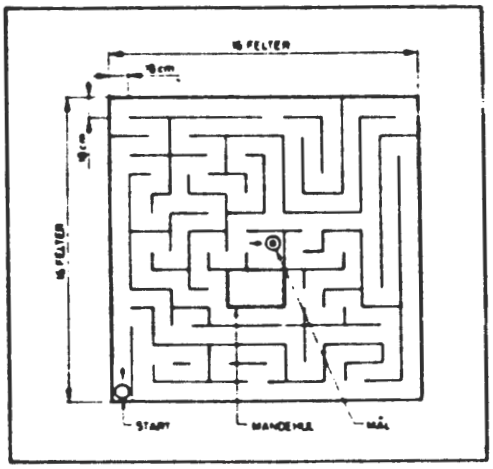
Resultatet blev at et finsk hold vandt i tiden 31,6 sek. Andenprisen gik også til Finland. tredjepræmien gik til England. (Præmierne var: 1000$, Spectrum og 120 pund).

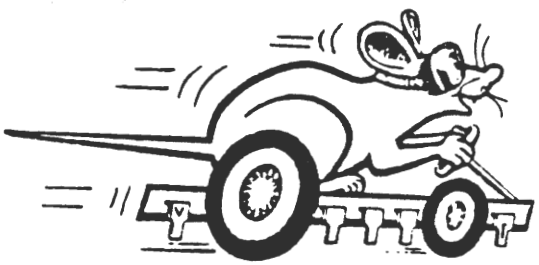
København

I år er der store forventninger til finalen i Lyngby, da førsteprisen er en rejse til Tokioi Japan for at deltage i det mesterskab, der afholdes dér senere på året. Det kunne være interessant, om der også var dansk deltagelse ved denne lejlighed, således at den finsk dominans kunne brydes.

Z80 brugergruppen

Det ville måske være en god lejlighed til at vise, hvad gruppens medlemmer kan præstere - lad ikke denne fantastiske lejlighed gå fra dig.





RECURSION

af Asbjørn Lind

Dette emne, har for mange været en stadig kilde til forvirring. Hvad er det, der sker? Er det særlig fint at skrive recursive programmer? Hvorfor gør man det? Det er nogle af de spørgsmål, jeg vil forsøge at besvare her.

Som jeg skrev i forrige nummer under næste nummers indhold: Recursion: Se recursion! Der er en lille sandhed i denne forklaring, men den er mere at betegne som en morsomhed (lige som den der findes nederst på siden). Men recursion er en rutine, procedure, funktion eller subrutine, der kalder sig selv. Enten direkte eller gennem en anden rutine. Det sidste kaldes så indirekte recursion.

En direkte anledning til recursion er f.eks. matematiske udtryk, der beskriver betingelser, det ville være meget vanskeligt at klare uden at bruge dette begreb. Et af de mest fortærskede emner er funktionen: Fakultet, som kort kan beskrives sådan, at 4 fakultet (skrives 4!) er lig med produktet 1*2*3*4 eller 6!=1*2*3*4*5*6 osv. Men hvordan bringer man det på en matematisk formel? Det kan gøres på to linier for n>=0:

```
Fak(0)=1
Fak(n)=n*Fak(n-1)
```

eller Fibonacci-tal, som er tal, der fremkommer, når man i en talrække addere de to foregående tal for at få det næste. F.eks. vil Fibonacci med to ettaller som start blive: 1 1 2 3 5 8 13 21

Hvis vi overspringer de to første tal i definitionen bliver de resterende som følger:

```
Fib(n)=Fib(n-1)+Fib(n-2)
```

I mange sprogbeskrivelser er det direkte nødvendigt at benytte sig af recursive definitioner, da det ellers ville være en uoverkommelig opgave at beskrive det pågældende sprog.

Det er også ofte benyttet i forbindelse med grafik og ikke at forglemme i det meget 'berømte' LOGO - på dansk 'myre-snak'.

Vi skal nu komme ind på lidt teoretisk stof. Når en funktion kalder en rutine (kald den, hvad du vil alt efter programmeringsprog), så skal rutinerne efterlade sig en returadresse, hvortil den skal vende tilbage, når rutinen er slut. Denne returadresse lagres på en stack, som er en stabel af returadresser, hvis man kalder mange funktioner inden i hinanden. Det kan sammenlignes med den røde tråd, som blev rullet ud i labyrinten, for at give mulighed for at vende tilbage til udgangspunktet!

I Basic, som er det mest almindelige sprog, er det ikke almindeligt at bruge recursion. Det skyldes variablernes rækkevidde! Hvad er nu det? Ja, vi bliver nød til at beskæftige os med lidt uden for emnet, for at forstå den dybere mening med det.

Når vi bruger en variabel i Basic, forventer vi, at vi har værdien til rådighed i alle dele af programmet. Det kan have sine fordele og ulemper (dog mest det sidste: vi skal sørge for, at der ikke optræder utilsigtede sideeffekter på vores variable. Det vil sige, at vi skal sikre os, at ikke to variable for samme variabelnavn, for så går det galt.) Hvis vi kalder en subrutine i Basic fra samme subrutine, vil variablen være tilgængelig i alle niveauer. Dette er en begrænsning, som man har prøvet at rode bod på i andre sprog. F.eks. er det ikke tilfældet i Comal-80, som man siger er en udvidet Basic - vrøvl!

```
**********************************
*   Se på den anden side         *
*   af papiret.                  *
*   Det der står der,            *
*   er ikke                      *
*            SANDT !!!!          *
*                  (vend)        *
**********************************
```

I Pascal, som er det mest anvendelige sprog til forklaring af emnet, er der et fænomen, der kaldes scope eller variablernes rækkevidde. I underprogrammer eller rutiner kan man definere variable, der kun har virkning, når rutinen kaldes. Det skal forstås således, at hovedprogrammet ikke har tilgang til disse variable. De har en begrænset levetid, nemlig den tid hvor i programmet er i rutinen! Det vil sige, at man kan have flere variable med samme navn, bare de ikke eksisterer i samme rutine. Rækkevidden er bestemt af, hvor definitionen foretages. Hvis den foretages lige efter programhovedet er variablen global, hvis den foretages i et procedurehoved er den lokal, hvis der ikke senere i proceduren defineres nye procedurer, for så er den global i forhold til disse. Selv om der eksistere to variable med samme navn, men defineret både i starten af programmet og i proceduren er det to helt forskellige variable, der intet har med hinanden at gøre.

Når man kalder et underprogram i pascal, kan man forestille sig, at en kopi af rutinen bliver lagt ud i lageret, hvorefter det bliver udført med sine variable. Efter endt udførelse bliver lagerområdet stillet frit og derved tabes indholdet af rutinens variable. I en parentes kan det nævnes, at datatransport mellem de forskellige rutiner foregår via den før nævnte stack, herved adskiller Pascal sig fra Basic, der har sine variable som adresser i lageret hele tiden.

Når nu en rutine kalder sig selv, kopierer den sig til et ledigt lagerområde. Det vil sige, at hvis en rutine kalder sig selv 5 gange, ligger der 5 ens udgaver af rutinen i lageret. Man kunne f.eks. forestille sig dem i fem forskellige farver.

For at belyse emnet, vil jeg nu komme med en hel del eksempler, der bliver kommenteret mere eller mindre. De fleste vil du være i stand til at prøve programmeerne, hvis du har en Pascalcompiler. En god idé, når man vil undersøge recursive funktioner, er at indsætte fortællende tekster og variable i funktionen på 'passende' steder. Det kan f.eks. være i starten og i slutningen af funktionen.

Det første recursive program er en simpel ordtæller i en given sætning. Det kunne selvfølgelig gøres ved at tælle mellemrum og lægge en til - men nu skal det s'gu være recursivt!

```
PROGRAM RCOUNT;
TYPE STR40=STRING(.40.);
VAR S:STR40;

PROCEDURE PRIK(N:INTEGER);
VAR I:INTEGER;
BEGIN
  FOR I:=1 TO N DO WRITE('. ');
END;
```

Programmet fortsætter på næste side.

```
*******************************
*   Se på den anden side      *
*   af papiret.               *
*   Det der står der,         *
*   er ikke                   *
*           SANDT !!!!        *
*                 (vend)      *
*******************************
```

```
FUNCTION TAEL_ORD(S:STR40;NIVEAU:INTEGER):INTEGER;
VAR ANTAL,K,L:INTEGER;
BEGIN
  K:=POS(' ',S);L:=LEN(S);
  PRIK(NIVEAU);WRITELN(S,', L=',L);
  IF K>0 THEN ANTAL:=1+TAEL_ORD(COPY(S,K+1,LEN(S)-K),NIVEAU+1)
  ELSE ANTAL:=1;
  TAEL_ORD:=ANTAL;
  PRIK(NIVEAU);
  WRITELN('Slut på denne farve, antal ord indtil nu =',ANTAL,', L=',L);
END;

BEGIN
  WRITELN('Indtast en linie tekst');
  READLN(S);
  WHILE LEN(S)>0 DO
    BEGIN
    WRITELN(TAEL_ORD(S,0),' ORD');
    WRITELN;
    WRITELN('Indtast en ny sætning (stop med <CR> alene)');
    READLN(S)
    END;
END.
```

Indrykningen skal fortælle os, i hvilken dybde recursionen er nået. Lige langt indrykkede linier hører til samme 'farve' af funktionen

```
 dette er en test på ordtælleren, L=31
 . er en test på ordtælleren, L=25
 . . en test på ordtælleren, L=22
 . . . test på ordtælleren, L=19
 . . . . på ordtælleren, L=14
 . . . . . ordtælleren, L=11
 . . . . . Slut på denne farve, antal ord indtil nu =1, L=11
 . . . . Slut på denne farve, antal ord indtil nu =2, L=14
 . . . Slut på denne farve, antal ord indtil nu =3, L=19
 . . Slut på denne farve, antal ord indtil nu =4, L=22
 . Slut på denne farve, antal ord indtil nu =5, L=25
 Slut på denne farve, antal ord indtil nu =6, L=31
 6 ORD
```

Vi skal ikke glemme fakultet, så den komme her. Der er også brugt indrykning, så vi bedre kan følge med.

```
program fakultet;
var x: integer;

procedure prik(n:integer);
var i:integer;
begin
  for i:=1 to n do write('. ')
end;

function fakultet(n,niveau:integer):integer;
var rfak:integer;
begin
  prik(niveau);
  writeln('start på fakultet, n=',n);
```

```
  if n>1 then rfak:=fakultet(n-1,niveau+1)*n
         else rfak:=1;
  fakultet:=rfak;
  prik(niveau);
  writeln('slut på fakultet, n=',n,' Fakultet=',rfak);
end;

begin
  writeln('Indtast tal mellem 0 og 7');
  readln(x);
  while (x>=0) and (X<=7) do
    begin
    writeln(fakultet(x,0));
    writeln;
    writeln('Indtast et nyt tal');
    readln(x);
    end;
end.
```

Det producerer følgende udskrift på skærmen:

```
 Indtast tal mellem 0 og 7
 0
 start på fakultet, n=0
 slut på fakultet, n=0 Fakultet=1
 1
 Indtast et nyt tal
 2
 start på fakultet, n=2
 . start på fakultet, n=1
 . slut på fakultet, n=1 Fakultet=1
 slut på fakultet, n=2 Fakultet=2
 2
 Indtast et nyt tal
 4
 start på fakultet, n=4
 . start på fakultet, n=3
 . . start på fakultet, n=2
 . . . start på fakultet, n=1
 . . . slut på fakultet, n=1 Fakultet=1
 . . slut på fakultet, n=2 Fakultet=2
 . slut på fakultet, n=3 Fakultet=6
 slut på fakultet, n=4 Fakultet=24
 24
 Indtast et nyt tal
 7
 start på fakultet, n=7
 . start på fakultet, n=6
 . . start på fakultet, n=5
 . . . start på fakultet, n=4
 . . . . start på fakultet, n=3
 . . . . . start på fakultet, n=2
 . . . . . start på fakultet, n=1
 . . . . . slut på fakultet, n=1 Fakultet=1
 . . . . . slut på fakultet, n=2 Fakultet=2
 . . . . slut på fakultet, n=3 Fakultet=6
 . . . slut på fakultet, n=4 Fakultet=24
 . . slut på fakultet, n=5 Fakultet=120
 . slut på fakultet, n=6 Fakultet=720
 slut på fakultet, n=7 Fakultet=5040
 5040
```

```
Indtast et nyt tal
-9
```

Så kommer vi til Fibonacci-tallene. Her viser det sig, at det ikke altid kan betale sig at bruge recursion, selv når definitionen lægger op til det. Det vil både kræve mere lagerplads og tage længere tid end den direkte metode i dette tilfælde. Hvis du kører programmet, vil du opdage, at den recursive del kalder sig selv flere gange, selv ved små værdier af fibonacci-tal. En beregning vil vise, at proceduren vil blive kaldt 1024 gange, hvis Fib(11) skal beregnes - det er spild af memory og tid!! Men prøv selv at køre programmet og bliv overbevist, hvis du er recursiv-fan.

```
program test_fibonacci;
var x,fibo:integer;

procedure prik(n:integer);
 var i:integer;
begin
  for i:=1 to n do write('. ')
end;
function fibonacci(n,niveau:integer):integer;
var rfib:integer;
begin
  prik(niveau);
  writeln('ind, niveau=',n);
  if n>1 then rfib:=fibonacci(n-1,niveau+1)+fibonacci(n-2,niveau+1)
         else
         if n=1 then rfib:=1
                else rfib:=0;
  fibonacci:=rfib;
  prik(niveau);
  writeln('ud, niveau=',n,' fibonacci=',rfib)
end;

begin
  writeln('Indtast tal mellem 0 og 7');
  readln(x);
  fibo:=fibonacci(x,0);
  writeln('Det ',x,'. fibonaccital=',fibo)
end.
```

Du kan selv konstruere en funktion, der vil beregne de enkelte fibonaccital direkte uden recursion. Derved vil du blive overbevist om, at man skal tænke sig godt om, før man bevæger sig ind på det recursive.

Inden jeg forlader det matematiske, vil jeg lige tilføje to programmer skrevet i COMAL-80, der også bruger recursion. Det ene bruger Euklids algoritme til at finde største fælles divisor, og det andet er et konstrueret eksempel på en funktion, der spejlvender en tekststreng.

```
1000 FUNC SFD(X,Y)
1010   IF X MOD Y=0 THEN
1020     RETURN Y
1030   ELSE
1040     RETURN SFD(Y,X MOD Y)
1050   ENDIF
1060 ENDFUNC SFD
```

```
1000 FUNC SPEJL$(T$)
1010   L:=LEN(T$)
1020   IF L=1 THEN
1030     RETURN T$
1040   ELSE
1050     RETURN T$(L:L)+
           SPEJL(T$(1:L-1))
1060   ENDIF
1070 ENDFUNC SPEJL$
```

Et meget betydeligt område for recursion er det grafiske område. Det er måske helt umuligt at producere disse figurer uden brug af recursion.

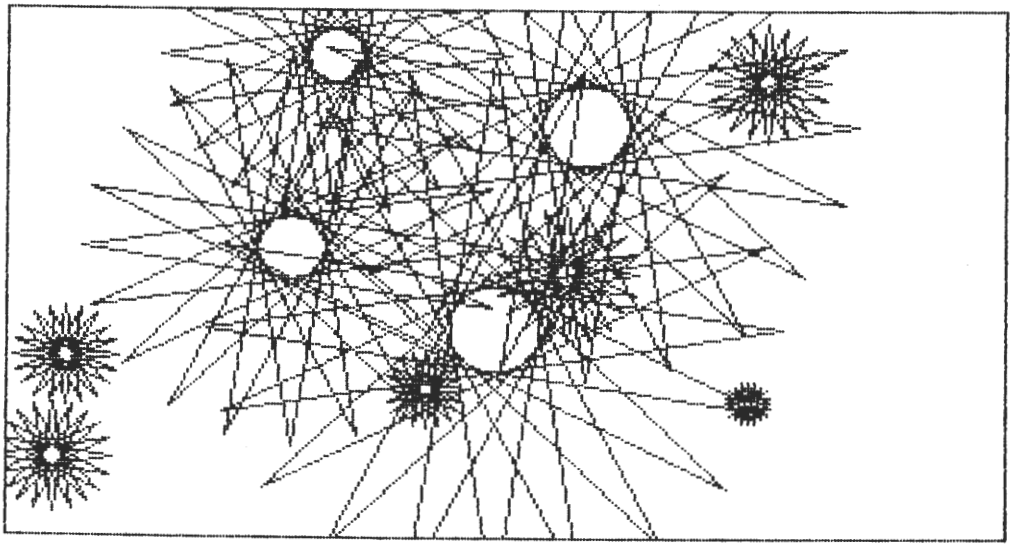
Men jeg vil starte forfra igen og lede jer gennem et grafik eksempel, hvorefter det er op til jer, at prøve at finde ud af, hvordan de øvrige eksempler fungerer!

Men først en lille snak om LOGO. En del af det undervisningssprog består af en tegnemaskine (turtle), der er i stand til at tegne på papir. I stedet for dette er der mange maskiner, der bytter papir ud med dataskærm og den 'levende' turtle ud med en figur på skærmen, der efterlader et spor, hvor den har gået. Et sådant sprog kan hedde MYRE-snak på dansk. Der er nogle ganske få ordre, der får myren til at bevæge sig rundt på skærmen. Det kan være frem(3) eller drej(90) osv. Men se i programmer, hvor jeg også har brugt de amerikanske betegnelser.

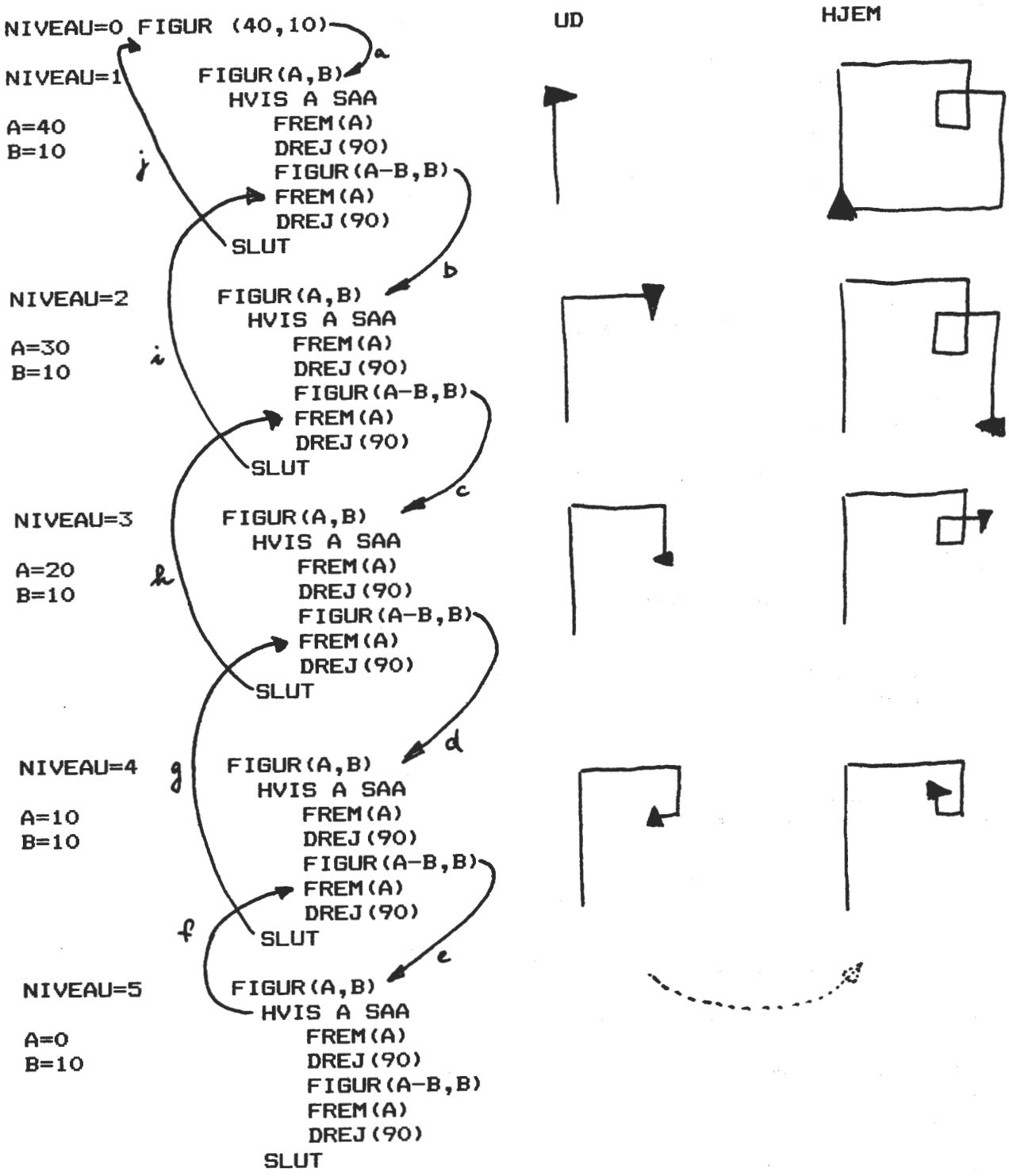
I parentes skal det nævnes, at der er konstrueret en simpel ikke helt afpudset MYRE-fortolker til Gemini's IVC-kort. Hvis du er interesseret, så henvend dig til Asbjørn Lind eller Peter Villadsen. Jeg har ikke fået skrevet nogen manual til sproget endnu, så det er stadig prerelase!!

Det skulle være let at fatte simpel geometri og mønsteropbygning med dette sprog. Her kommer et eksempel på rekursiv myresnak:

```
FIGUR(A,B)
  HVIS A SAA
     FREM(A)
     DREJ(90)
     FIGUR(A-B,B)
     FREM(A)
     DREJ(90)
  SLUT
```



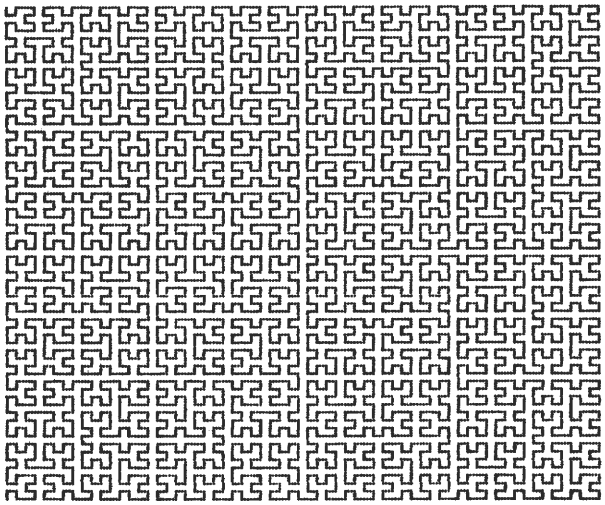
ET ANDET TURTLE PGR.



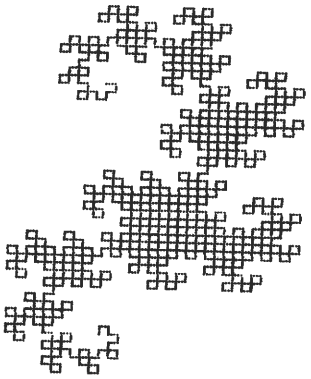
Her følger så to programmer, som ikke bliver kommenteret, men hvor det grafiske resultat er sakset ind.

```
program hilbert;
var
  size,delta,n: integer;
  order: integer;

(*$I grafikin *)

procedure hil(i:integer);
var
  a,b:integer;

  procedure hil1;
  begin
    turn(a);hil(-b);turn(a);
  end (*hil1*);
  procedure hil2;
  begin
    move(size);
    hil(b);
    turn(-a);move(size);turn(-a);
    hil(b);
    move(size);
  end (*hil2*);
begin (*hil*)
  if i=0 then turn(180)
  else
  begin
    if i>0 then
    begin
      a:=90; b:=i-1;
    end
    else
      begin
        a:=-90;b:=i+1;
      end;
    hil1; hil2; hil1;
  end;
end(*hil*);

(*fortsættes i næste spalte*)
```

```
begin (*main*)

  coaddr:=addr(prchr);
  write(chr(27),'sC');

  write('size: ');
  readln(size);
  write('order: ');
  readln(order);

  grafikon;

  n:=order-1;
  delta:=size;
  while n>0 do
  begin
    delta:=delta*2;
    n:=n-1;
  end;
  moveto(-150,100);
  hil(order);

  grafikoff;
  write(chr(27),'sC')
end.
```



```
 program dragon;
var
  size:integer;

(*$I grafikin *)

procedure dragon(length:integer);
begin
  if length=0 then move(3)
  else
    if length>0 then
      begin
        dragon(length-1);
        turn(90);
        dragon(-(length-1));
      end
    else
```



```
      begin
        dragon(-(length+1));
        turn(270);
        dragon(length+1);
      end;
end;

begin
  coaddr:=addr(prchr);
  write(chr(27),'sC',clrhom);
  write('Size of dragon: ');
  readln(size);
  grafikon;
  moveto(40,70);
  dragon(size);
  grafikoff;
  write(chr(27),'sC')
end.
```

Jeg kan ikke nære mig for at bringe et program, som vi i pascal-studiekredsen gennemgik en af de sidste gange. Det illustrerer POINTER-begrebet og LISTE-begrebet, samt men ikke mindst recursion i særdeleshed.

```
program liste_pointer_demo;
type
  string24 = string(.24.);
  personpeger = ^person;

  person = record
             navn: string24;
             haegte: personpeger
           end;
var
  komtil,rod,ppil1: personpeger;

procedure opbyg(var lrod: personpeger; pnavn: string24);
begin
  new(lrod);
  with lrod^ do
    begin
    navn:= pnavn;
    haegte:=nil
    end
end;

procedure dan_liste(rod: personpeger);
(*Denne procedure danner listen med navne. Der returneres naar brugeren
  taste <CR> for et personnavn*)
var
  navn: string24;
  pil: personpeger;
begin
  write ('Indtast navnet '); readln(navn);
  while len(navn)<>0 do
    begin
    opbyg (pil,navn);
    komtil^.haegte:=pil;
    komtil:=pil;
    write(' Indtast navnet '); readln(navn)
```

```
    end;
end; (*dan_liste*)

procedure udskriv_liste (pil: personpeger);
begin
  while pil<>nil do
    begin
    writeln(pil^.navn);
    pil:=pil^.haegte
    end;
  writeln; writeln('*** ikke flere navne ***')
end; (*udskriv_LISTE*)

procedure bagfra(pil:personpeger);
begin
  if pil^.haegte <> nil then bagfra(pil^.haegte);      (* Her recursiv *)
  writeln(pil^.navn)
end;

(*hovedprogram*)
begin
  (*opbyg dummy post*)
  new(ppil1);
  ppil1^.navn:='$$$$$$$$$$$$$$$$$$$$$$$$';
  rod:=ppil1;
  komtil:=ppil1;
  dan_liste(ppil1);
  udskriv_liste(rod^.haegte);
  bagfra(rod^.haegte)
end.
```

Z80-brugergruppen 'Sydfyn'

søger kontakt med interesserede i bl.a. maskinkodeprogrammering ved hjælp af Z80 assembler og/eller datatransmission ved anvendelse af MODEM's, samt evt. Basic-kursus, på New Brain eller andre.
Afhængig af tilslutningen er det hensigten, at oprette studiegrupper indenfor de forskellige interesseområder, såvel "begynderen" som øvede vil kunne få udbytte af at medvirke.
Brugergrupper fra andre dele af landet er velkommen til, at reagerer på nærværende med henblik på, at hjælpe os igang med f.eks. materialer og erfaringer.
Kontaktperson er Poul E. Thestrup, Avernakøvej 13, 5700 Svendborg.
Husk at vedlægge adresseret og frankeret svarkonvolut, det sikre hurtig orientering om tid, sted og mødeprogram.

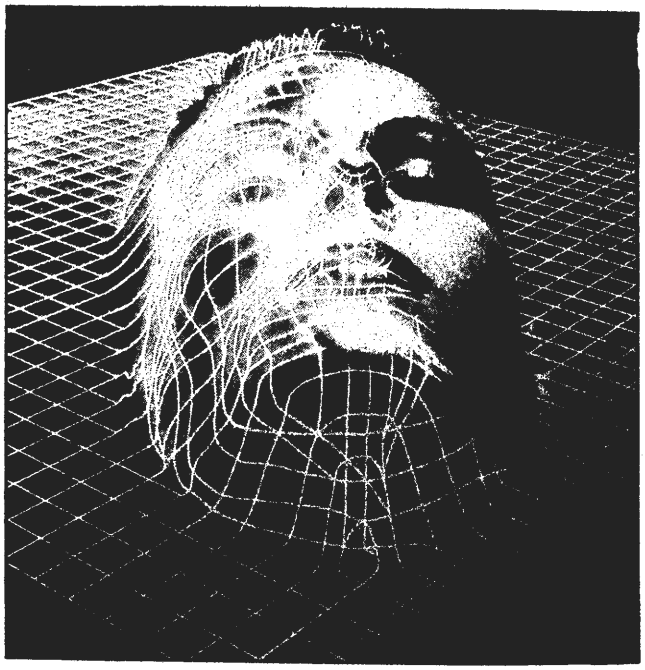
Med venlig hilsen
P. E. Thestrup (391)

Programbiblioteket

Der er kommet yderligere 3 formater til: Decision Mate IV, Osborne SSSD og ATEC-80. Husk,hvis du har programmer, at sende dem ind til programbiblioteket. Alle slags programmer modtages - eneste begrænsning er - at de skal kunne køre på dit CP/M anlæg. Det ville være dejligt, hvis også sourceteksterne blev medsendt. Lidt dokumentation ville så sandelig heller ikke være at foragte, men det er ikke alt afgørende.

HUSK HUSK PRÆMIEMULIGHED ! !

Har du husket den gule seddel om medlemsundersøgelsen. Hvis du har indsendt den inden 1.6.1984, vil du være med i konkurrencen om Jesper Skavins bog: 'Maskinkodprogrammering med Z80'. Resultatet af lodtrækningen i næste nummer af Z80 NYT.



COMPUTER-STRESS NYÅRSAG TIL SKILSMISSER

Frank, en 36-årig programmør, har været gift og er blevet skilt igen to gange. Frank har for mere end 100.000 kr. computere og udstyr - og han elsker det. Efter en 8-timers arbejdsdag tager han hjem og tilbringer endnu 8-10 timer med sine computere hjemme. Hans første kone forlod ham, fordi - som han siger - »hun forstod mig ikke«. Hans anden kone tog mere håndfast på tingene, og anbragte to af hans computere i udhuset. Så blev de skilt.

Frank er et ekstremt tilfælde, men han er udtryk for et voksende problem: Mennesker, der tilbringer det meste af deres tid sammen med en computer, har svært ved at omgås andre mennesker.

I USA er problemet blevet så påtrængende, at en psykolog i Texas, Thomas McDonald, har specialiseret sig i behandling af computer-plagede patienter.

- Tusinder af mennesker har problemer med at være sammen med andre på grund af computere, og der vil blive flere og flere efterhånden som computerne bliver en almindelig del af vores liv, siger Thomas McDonald. Han tilføjer, at computere er fåmælte og dejligt logiske. Mange mennesker oplever det som en befrielse sammenlignet med det mere komplekse samvær med andre mennesker.

Thomas McDonald fortæller, at de første computer-ramte par kom til hans konsultation for et par år siden. Siden har han behandlet flere end 1000 tekno-stressede mennesker.

- Det drejer sig næsten altid om mænd, der omgås computere på deres arbejde, siger han. De virkelig hårdnakkede tilfælde, som knap nok holder computer-pause for at spise eller sove, har heller ingen problemer med deres omgangskreds. De har ingen.

Illustration Barry Blackman

SÆLGES / ØNSKES

Print og dokumentation til det engelske ANIMATION GRAPHICS BOARD uden bånd (der kan lånes). Jeg har modtaget det i maj og skal kun bruge båndet. Pris 260 kr.
Ønsker NASSYS 3 med 'indbygget' ANIMATION GRAPHICS evt. i forbindelse med ASCII keyboard. Jeg er ved at lave en maskine af en ZX81, NASSYS 3, TMS9129 på europakort med en ECB-bus.
Henvendelse til Hans Tuxen på telefon: 09 72 19 08

SÆLGES

Nascom 2, 64K RAM, diskkontroller, TEAC 55E, 8 tommers monitor/grøn Strømforsyning. Polydos + software. I 19 tommer rack med selvstændigt keyboard. Pris: ca. 12.000 kr.
Henvendelse hos Kim Fuglsang efter kl. 17.30 på 07 14 22 51

ANNONCE/EFTERLYSNING.

Er der en der har en listning af Polysys 2 Prom version, da min ene PROM er brændt af. Det er adresse B400H til B7FFH.
René Olesen; Gammel Mosevej 107G,1MF; 2800 Lyngby

Sælges

Midicos 2 system for Nascom 1/2 med 2 stk. Philips mini digital båndoptagere, interfacekort, operationssystem og bootprom. Der medfølger 6 bånd med mange programmer bl.a. skakprogram. Pris komplet incl. manual kr. 3000,- eller bedste bud.
NCR printer type C260-100 termisk kører 150 eller 300 baud. Printeren kører serielt. Beregnet til papir i ruller uden perforering. Sælges med manual for kr. 950,-
Henvendelse: J. Hørring (347) på tlf.: 02 87 63 01

Annonce

Nascom 2 computer, indbygget i brun/beige metalkabinet, incl motherboard med plads til 4 kort. Seperat keyboard kabinet. Gemini GM 802 64K RAM-kort, danske tegn, grafik, Nassys 1/3, BLS pascal, HP4T Pascal, Forth, Basic, ExBasic, Nap/Nip, Naspen, Wordease, div. spil. Sælges samlet for kr. 6000,-
Seikosha GP 250 X printer. Centronics og RS 232 interface, brugerdefinerbare tegn, mange printstørrelser, 80 tegn/linie, grafik, ekstra farvebånd samt 1/2 kasse papir. Sælges for kr. 3000
Zenith video mointor, grøn skærm med composite VDU indgang 1000 kr.
Astec AC 9231 switch mode strømforsyning. Helt ny, giver 5V/6A, 12V/2.5A, -5V/0.5A og -12V/0.5A. Svarer til Gemeni GM 817 1200 kr.
Henvendelse Jesper L. Olesen (250) tlf.: 03 70 64 12

Annonce — Sælges

Gemini 802 64K RAM sælges for 1500 kr. Henv. Ole Kammersgård på telefon 01 42 53 60

SÆLGES.
Memotech MTX 512 fabriksny med fuld garanti sælges. BrugerRAM 64K,og VidioRAM 16K, Z80A processor. BASIC og NODDY, Assembler og Disassembler. Register-, memory- og program-manipuleringsrutiner. Dansk tegnsæt, cassetteudgang (50-2400 Baud). Farve-TV og monitorudgang, PIO og 2 Joystick udg. Centronics printerudgang, 3 tonekanaler + støjkanal samt avanceret farvegrafik med 32 uafh. definerbare "Sprites".
Kan udbygges til max. 512K intern memory samt floppy-/harddisk indtil 32MB !! Kan udvides med 2 x RS 232C udg. på internt Comm. Bord, samt anvende MTX FORTH, PASCAL og NODE RING flerbrugersystem. MTX disksystemer kører under CP/M 2.2. Den har et lækkert stort ASCII skrivemaskinetastatur, separat 9/12V strømforsyning, demobånd og kabler. EN "STÆRK" SORT PERLE FOR KR. 5245,00.
Finn Christensen, Lauravej 5, 2500 Valby. Telf. 01 307810