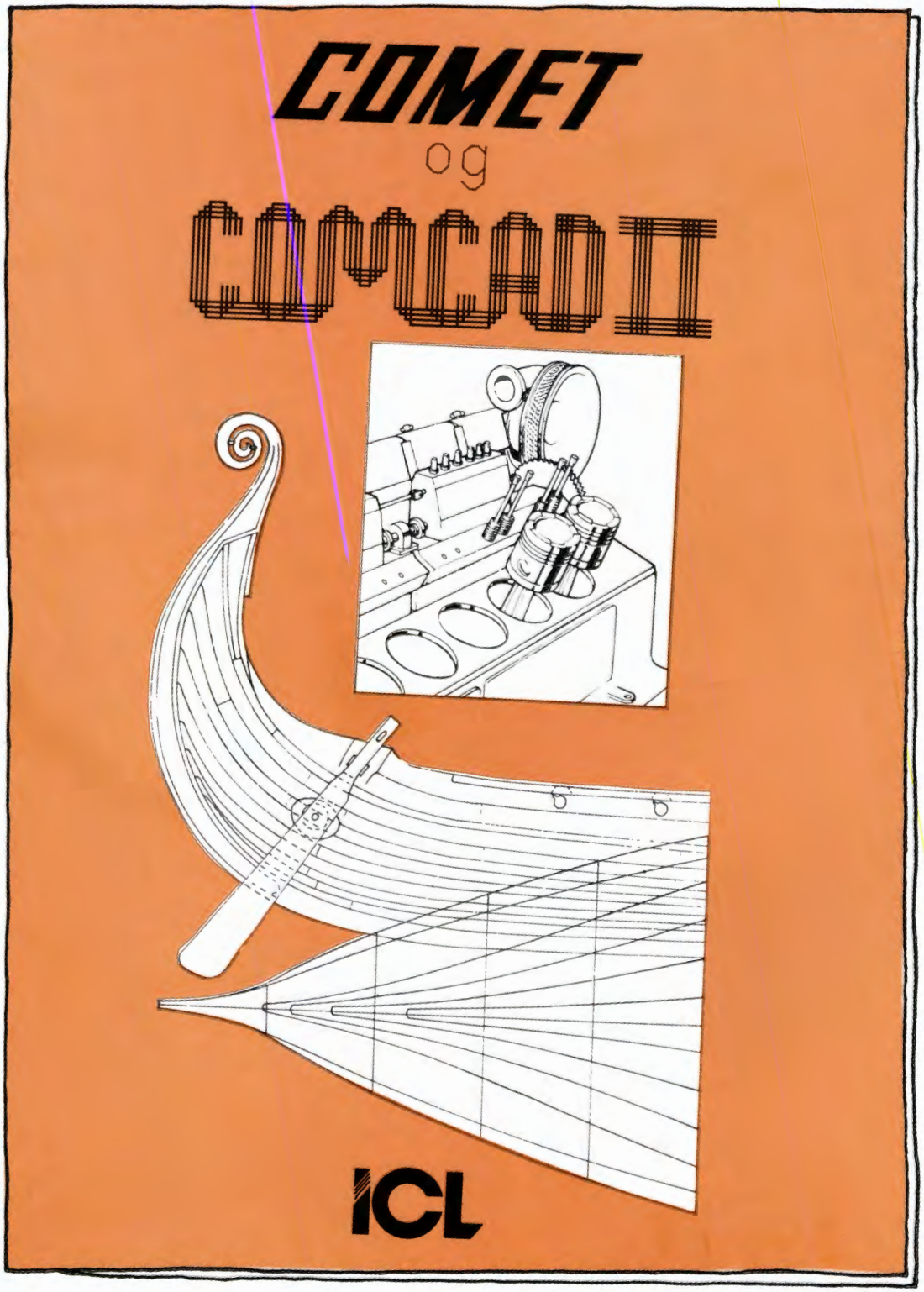
COMETEN

Nr. 2. Sept. 1986

omCad II
vanceret tegnings-
onstruktion.

Rekvirer den
nye brochure
på telefon
01 67 97 00
Comet afd.



Artikelserie
af Ulrich Lysdal Jensen

Behandling af regneudtryk i programmer

En væsentlig anvendelse af en datamat er beregning af regneudtryk/formler.

Først og fremmest skal fortolkere (COMAL-80) og oversættere (PASCAL) kunne tolke, lagre og udregne regneudtryk. Men også hvis man laver programmer til styrkeberegning, kurvetegning og lignende, kan man komme ud for tolkning af inddata i form af regneudtryk.

I det følgende beskrives en metode til behandling af regneudtryk, som ofte anvendes i fortolkere og oversættere, og som også let kan anvendes i andet programmel, hvor formler og regneudtryk skal kunne behandles.

Normal notation

I almindelighed beskriver vi, at tallene 285 og 517 skal lægges sammen (adderes) på følgende måde:

285 + 517 =

Tallene 285 og 517 kalder vi operander, medens plus-tegnet (+) kaldes en operator ligesom – (minus) * (gange) og / (dividere).

Ved opskrivning af regneudtryk kan man også anvende symbolske variabelnavne for operander – f.eks.:

a + b = eller hastighed * tid =

Vi kan som bekendt også skrive regneudtryk, der indeholder mere end 2 operander:

a + b * c – d / e =

og vi kender alle reglerne for udregningen: * og / udføres først (har højest prioritet).

I den almindelige skriveform – normal notation vil vi kalde det – skrives operatoren altså **mellem** de to operander, den skal virke på.

En konsekvens af normal notationen og de dermed forbundne regneregler er bl.a., at der må indføres parenteser i forbindelse med mere komplicerede udtryk:

(a + b) * c =

eller endog parenteser i flere niveauer (parentser i parenserer):

((a + b) * c + (d + e) / (f – g)) =

Når vi skal udregne et sådant udtryk 'på papiret', gør vi i høj grad brug af menneskelige evner til at skabe sig et overblik. Vi starter med at overskue hele regneudtrykket og derudfra træffer vi beslutning om rækkefølgen af beregning af de enkelte deludtryk.

En datamat har ikke evnen til overblik. I en datamat må elementerne i f.eks. et regneudtryk behandles et ad gangen i en ganske bestemt rækkefølge – f.eks. fra venstre mod højre. Derfor bliver det hurtigt en kompliceret sag at skulle behandle regneudtryk med parenteser i flere niveauer i et program. Hvad der måske er endnu mere beklageligt er, at der ofte vil blive indført visse begrænsninger – f.eks. i antallet af parenteser inden i parenteser – hvis man holder sig til regneudtryk i normal notation.

'Geografisk' er der to andre måder at skrive et regneudtryk på:

1. Operatoren kan skrives **foran** de to operander, den skal virke på:

   + a b =

   Dette kaldes for **polsk notation.**

2. Operatoren kan skrives **efter** de to operander, den skal virke på

   a b + =

   Dette kaldes ikke helt uventet for **omvendt polsk notation.**

Navnlig sidstnævnte notationsform har nogle egenskaber, der gør den velegnet til behandling i en datamat:

– I omvendt polsk notation er der ikke behov for parenteser.

– Udregning af et regneudtryk skrevet med omvendt polsk notation kan ske ved at behandle de indgående elementer (operander og operatorer) et ad gangen fra venstre mod højre ved brug af en stak (se senere).

– Det er enkelt at oversætte et regneudtryk skrevet med normal notation til omvendt polsk notation.

Omvendt polsk notation

Som sagt skrives i operatoren omvendt polsk notation efter de to operander, den skal virke på. F.eks. skrives udtrykkene:

(a + b) * c eller (285 + 517) * 13

således:

a b + c * eller 285 517 + 13 *

Ved udregning af et sådant udtryk, startes fra venstre, og det læses indtil en operator mødes. Operatoren lader vi virke på de to foranstående operander, og resultatet indsættes i stedet for de to operander og operatoren. Derefter arbejder man sig videre mod højre efter samme princip.

De enkelte trin i beregning af udtrykket '285 517 + 13 * =' ser således ud:

1. Vi læser operanden 285 og går videre.
2. Vi læser operanden 517 og går videre.
3. Vi læser operatoren '+' og udregner 285 + 517 = 802, som indsættes. Regneudtrykket ser nu således ud: '802 13 * ='.
4. Vi læser operanden 13 og går videre.
5. Vi læser operatoren '*' og udregner 802 * 13 = 10426, som indsættes. Regneudtrykket ser nu således ud: '10426 ='.
6. Vi læser lighedstegnet, som fortæller, at regneudtrykket er slut. Det sidst udregnede tal 10426 er resultatet.

BEMÆRK at vi under udregningen ikke har beskæftiget os med parenteser og prioriteten for regneoperatorer.

Prøv at udregne følgende udtryk i omvendt polsk notation:

13 285 517 + * =
(samme resultat som før)

26 30 10 – 5 / 15 + * =
(resultat: 494)

I normal notation ser sidstnævnte udtryk således ud:

((30 – 10) / 5 + 15) * 26 =

Prøv at oversætte følgende udtryk i normal notation til omvendt polsk notation:

(a + b) * (c – d)

Resultatet skulle gerne blive:

a b + c d – * eller c d – a b + * =

En stak

Ved udregning af regneudtryk skrevet med omvendt polsk notation på en datamat, kan man med fordel anvende begrebet 'en stak' (eller stack) til mellemlagring af de operander, vi ovenfor i første omgang sprang over.

En stak er (i hvert fald i denne sammenhæng) et dataområde, der anvendes til midlertidig lagring af data. Når man skal lagre et nyt dataelement (f.eks. et heltal) i stakken, placeres det nye element sidst i dataområdet. Når man skal hente et dataelement fra stakken, henter man altid det element, der ligger sidst i stakken, altså det, der senest er lagret.

En sådan stak kaldes også en 'Sidst-Ind-Først-Ud' stak (SIFU-stak).

Der findes også dem, der kalder en SIFU-stak for en cafeteriastak, fordi den fungerer på samme måde som stakken af tallerkener i et cafeteria. Gæsterne tager hele tiden den øverste tallerken i stakken. Når der kommer rene tallerkener fra opvasken, placeres disse øverst i stakken, hvorefter gæsterne tager af de sidst ankomne tallerkener.

Man kan oprette en stak i et program ved at oprette et array af samme type, som de dataelementer, man ønsker at lagre i stakken.

som altså kan indeholde op til 20 heltal nummereret fra 0 til 19.

Vi får også brug for en pegepind (stak-pointer) – lad os kalde den PP # – som hele tiden peger på den næste ledige plads i stakken. Fra starten er stakken tom, hvorfor PP skal være 0. Et sted i starten af programmet skal vi altså have en sætning PP := 0.

Endelig kan vi med fordel skrive et par rutiner til hhv. at lagre et nyt element i stakken og til at hente et element fra stakken.

Til lagring af et heltal kan vi f.eks. anvende følgende procedurer:

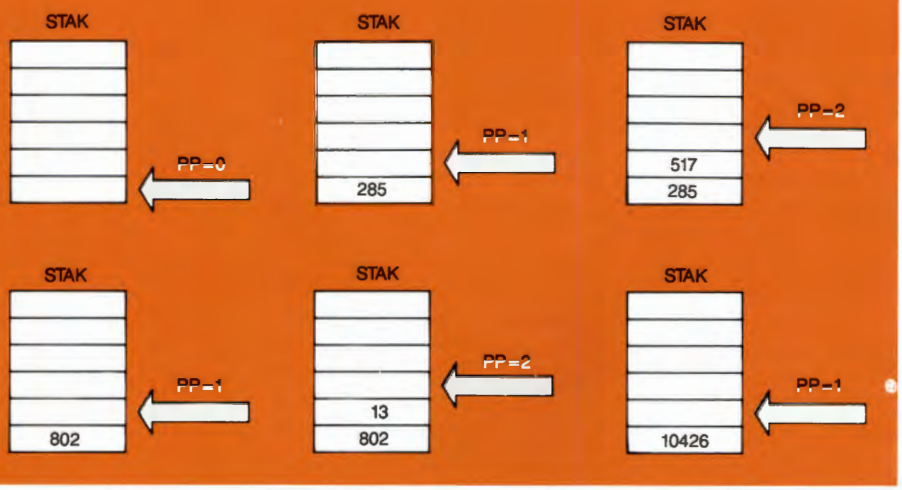
```
PROC PUSH (HT#)
  IF PP# < 20 THEN
    STAK#(PP#) := HT#          //HT# lagres sidst i stakken
    PP#+=1                     //og stakpointeren tælles op
  ELSE
    PRINT "Stakken er fuld"    //Hvis stakfejl stoppes
    END                        //programmet
  ENDIF
ENDPROC
```

Til at hente et element fra stakken, anvendes f.eks. følgende funktion:

```
FUNC POP ()
  IF PP      > 0 THEN
    PP#–=1                     //Stakpointeren tælles med
    RETURN(STAK#(PP#))         //Det udpegede element
  ELSE                         //returneres
    PRINT "Stakken er tom"     //Hvis stakfejl stoppes
    END                        //programmet
  ENDIF
ENDPROC
```

Vi starter med en tom stak:

```
PUSH(285)
PUSH(517)
A := POP() + POP()
eller
PUSH(285)
PUSH(285)
PUSH(POP() + POP())
```



Der er en stærk tradition for, at rutiner til at lagre og hente i stakken kaldes PUSH og POP. Sådan hedder de tilsvarende instruktioner i symbolsk maskinsprog (assembler) nemlig ofte.

Artikelserie
af Ulrich Lysdal Jensen

Omvendt polsk notation og stakken

Lad os nu se, hvorledes man beregner et regneudtryk, skrevet i omvendt polsk notation ved hjælp af en stak.

Udtrykket læses fra venstre mod højre.

1. Når der læses en **operand,** stakkes denne ved hjælp af PUSH.
2. Når der læses en **operator,** afstakkes de to øverste elementer i stakken ved hjælp af POP, og den læste operator udføres på de to afstakkede elementer. Resultatet stakkes atter ved hjælp af PUSH.
3. Når hele udtrykket er læst, findes resultatet af hele udtrykket øverst i stakken.

I COMAL-80 kan man anvende udtryk og funktionskald som parametre til et procedurekald. Derfor kan pkt. 2 for f.eks. addition se således ud:

PUSH(POP() + POP())

For udtrykket: 285 517 + 13 * = ser forløbet af udregningen således ud:

1. Læs 285:	PUSH(285)
2. Læs 517:	PUSH(517)
3. Læs +:	PUSH(POP() + POP())
4. Læs 13:	PUSH(13)
5. Læs * :	PUSH(POP() * POP())
6. Læs = –> slut	RESULTAT:= POP()

Det viste COMAL-80 program indlæser et regneudtryk skrevet i omvendt polsk notation. De indtastede tegn analyseres efterhånden som de indtastes, og beregninger udføres efter ovenstående metode. Når = læses, udskrives resultatet. Et tryk på 'C' annullerer det hidtil indtastede. Programmet afsluttes ved tryk på 'Q'.

For overskuelighedens skyld er programmet gjort så enkelt som muligt. F.eks. kan der kun indtastes heltal. En udvidelse til reelle tal kræver blot en udvidelse af proceduren GETOP og naturligvis ændring af variabelnavnene.

Oversættelse fra normal til omvendt polsk notation

Ved behandling af regneudtryk i programmer, kan man indlæse udtrykkene i normal notation, oversætte det indlæste til omvendt polsk notation og derefter foretage udregningerne ved hjælp af den ovenfor beskrevne metode.

Man kunne tænke på et program, der kan udskrive en tabel over en given formel i et givet interval.

En metode til oversættelse fra normal til omvendt polsk notation bygger på anvendelsen af to stakke. I den ene stak – STAK1 – lagres udtrykket på omvendt polsk notation, efterhånden som oversættelsen skrider frem. Den anden stak – STAK2 – anvendes til mellemlagring af operatorer under oversættelsen.

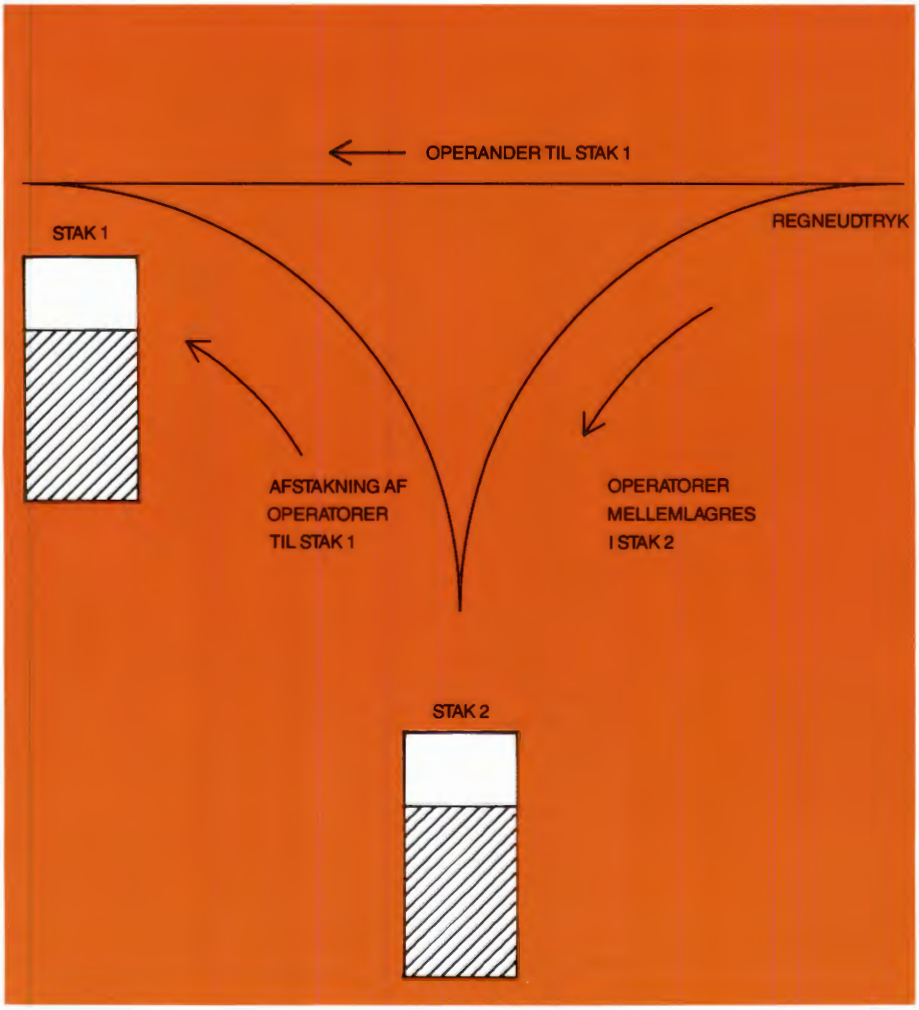
Det hele fungerer nogenlunde som et sidespor på en jernbane:

Allerførst skal vi dog lige indføre en prioritering for regneoperatorer og parenteser. Idet prioritet 0 er laveste prioritet, indføres følgende:

(og):	prioritet 0
+ og –:	prioritet 1
* og /:	prioritet 2

Oversættelsen forløber nu efter følgende regler, idet udtrykket i normal notation læses fra venstre mod højre:

1. Læses en **operand,** stakkes denne i STAK1.
2. Læses en **venstre parentes,** stakkes denne i STAK2.
3. Læses en **operator,** undersøges indholdet af STAK 2:
   - 3.1 Er STAK2 tom, stakkes operatoren i STAK 2.
   - 3.2 Indeholder STAK2 allerede operatorer og parenteser, afstakkes disse til STAK1, indtil STAK2 er tom eller indtil næste element i STAK2 har lavere prioritet end den læste operator, som derefter stakkes i STAK 2.



4. Læses en **højre parentes** afstakkes indholdet af STAK2 til STAK1 indtil øverste element i STAK2 er en venstre parentes. Denne afstakkes og glemmes ligesom den læste højre parentes.
   BEMÆRK: findes der ikke en venstre parents i STAK2, er der fejl i det læste udtryk (syntaks analyse).

5. Når hele udtrykket er læst, afstakkes et evt. indhold i STAK 2 til STAK1, som derefter indeholder det færdigt oversatte udtryk.
   BEMÆRK: Hvis der ved denne sidste afstakning findes en venstre parentes i STAK2, har der været fejl i det indlæste udtryk – ikke sammenhæng mellem venstre og højre parenteser.

Lad os prøve metoden på udtrykket:
((a + b * c – d) / (e + f * g) =

Ovenstående fremgangsmåde er beregnet for oversættelse og lagring af regnudtryk i et program, således at udtrykket kan beregnes igen og igen med forskellige værdier for de symbolske operander (variable).

Metoden kan også anvendes til direkte udregning af et udtryk i normal notation, som f.eks. indtastes til et program. Man skal da blot ikke stakke operatorer i STAK1 (men stadig mellem-stakke dem i STAK2). Derimod skal man lade operatorer, der ifølge ovenstående fremgangsmåde skulle have været stakket i STAK1, virke på de to øverste elementer (operander) i STAK1 og derefter stakke resultatet i STAK1. Når hele udtrykket er læst, står det endelige resultat som sidste element i STAK1.

Unitære operatorer

Hidtil har vi kun behandles operatorer, der virker på to operander – duale operatorer.

En anden type operatorer er unitære operatorer som f.eks. – (minus), kvadratrod, potensopløftning m.v.

De unitære operatorer kan udmærket medtages i ovenstående behandling, når man blot husker, at en unitær operator kun skal virke på det øverste element i en stak.

Trignometriske, eksponentielle, logaritmiske og andre matematiske funktioner kan behandles på samme måde.

Den omvendte polske notation gælder naturligvis også, hvis der er tale om logiske udtryk indeholdende logiske (boolske) variable (TRUE, FALSE, SAND USAND) og logiske operatoer (AND, OR, NOT osv.).

		STAK 1:	STAK 2:
1. Læs (	stak i STAK2	tom	(
2. Læs (	stak i STAK2	tom	((
3. Læs a	stak i STAK 1	a	((
4. Læs +	stak i STAK2	a	((+
5. Læs b	stak i STAK1	a b	((+
6. Læs)	afstak + fra STAK2	a b	((
	stak + i STAK1	a b +	((
	afstak (fra STAK2	a b +	(
7. Læs *	stak i STAK2	a b +	(*
8. Læs c	stak i STAK1	a b + c	(*
9. Læs –	afstak * fra STAK2	a b + c	(
	stak * i STAK1	a b + c *	(
	stak – i STAK2	a b + c *	(–
10. Læs d	stak i STAK1	a b + c * d	(–
11. Læs)	afstak – fra STAK2	a b + c * d	(
	stak – i STAK1	a b + c * d –	(
	afstak (fra STAK2	a b + c * d –	tom
12. Læs /	stak i STAK2	a b + c * d –	/
13. Læs (	stak i STAK2	a b + c * d –	/(
14. Læs e	stak i STAK1	a b + c d – e	/(
15. Læs +	stak i STAK2	a b + c d – e	/(+
16. Læs f	stak i STAK1	a b + c d – e f	/(+
17. Læs *	stak i STAK2	a b + c * d – e f	/(+ *
18. Læs g	stak i STAK1	a b + c d – e f g	/(+ *
19. Læs)	afstak * fra STAK2	a b + c * d – e f g	/(+
	stak * i STAK1	a b + c * d – e f g *	/(+
	afstak + fra STAK2	a b + c * d – e f g *	/(
	stak + i STAK1	a b + c d – e f g * +	/(
	afstak (fra STAK2	a b + c * d – e f g * +	/
20. Læs	afstak / fra STAK 2	a b + c * d – e f g * +	tom
	stak / i STAK1	a b + c * d – e f g * + /	tom

```
0010 DIM STAK# (0:19)
0020 DIM TALSTA$ OF 20
0030 DIM RETURVAL$ OF 1, C$ OF 1, L$ OF 1
0040 DIM GEMT$ OF 1
0050 //
0060 PROC POLSK // ---------- POLSK
0070   //
0080   // I denne procedure styres indlæsning,
         stakning, udregning og udskrift
0090   //
0100   REPEAT
0110     RETURVAL$:=GETOP$()
0120     CASE RETURVAL$ OF
0130     WHEN "Q", "q" //              Q stopper programmet
0140       END
0150     WHEN CHR$ (255) //            Tal indlæst
0160       TAL# :=IVAL (TALSTR$)
0170       EXEC PUSH (TAL# )
0180     WHEN "+" //                   Addition
0190       EXEC PUSH (POP# () +POP# () )
0200     WHEN "-" //                   Subtraktion
0210     OPERAND2:=POP ()
0220       EXEC PUSH (POP () -OPERAND2)
0230     WHEN "*" //                   Multiplikation
0240       EXEC PUSH (POP# () *POP# () )
0250     WHEN "/" //                   Division
0260       OPERAND2# :=POP# ()
0270       IF OPERAND2# =0 THEN
0280         PRINT CHR$ (10), CHR$, (13), "Division med 0"
0290         PP# :=0 //   Stakken tømmes ved division med 0
0300         PRINT "> > ";
0310       ELSE
0320         EXEC PUSH (POP# () DIV OPERAND2# )
0330       ENDIF
0340     WHEN "=" //                   Resultatet udskrives
0350       PRINT "   ", POP# ()
0360       PRINT "> > ";
0370       WHEN "C", "c"
0380         PP# :0 //          Regnemaskinen tømmes
0390         PRINT CHR$ (13), CHR$ (10) ;"> > ";
0400       OTHERWISE
0410         //
0420       ENDCASE
0430     UNTIL RETURVAL$="Q" OR RETURVAL$="q"
0440 ENDPROC POLSK
0450 //
0460 FUNC GETOP$ // ---------- GETOP$
0470   //
0480   // Denne funktion indlæser næste operand eller operator
0490   //
0500   REPEAT
0500     C$:=INDLES$ ()
0520   UNTIL C$ () CHR$ (32) //      Overspring blanke
0530   IF C$> ="0" AND C$ (="9 THEN
0540     TALSTR$:+C$
0550     REPEAT
0560       C$=INDLES$ ()
0570       IF C$ > ="0" AND C$ < ="9" THE TALSTR$:+C$
0580     UNTIL C$ ("0" OR C$) "9"
0590     GEMT$:=C$ //
         Sidst indlæste tegn gemmes til næste kald
0600     RETURN (CHR$ (255)) //
         Der returneres 255, hvis operand indlæst
0610   ELSE
0620     RETURN (C$) //
         ellers returneres indlæst karakter
0630   ENDIF
0640 ENDFUNC GETOP$
0650 //
0660 FUNC INDLES$ // ---------- INDLES$
0670   //
0680   // Indlæsning af tegn via adresse 256
0690   //
0700   IF GEMT$ () CHR$ (255) THEN //
         Først undersøges, om der er et gemt tegn
0710     L$:=GEMT$
0720     GEMT$:=CHR$ (255)
0730   ELSE
0740     POKE 256, 255
0750     REPEAT
0760       L$:CHR$ (PEEK (256))
0770     UNTIL L$ () CHR$ (255)
0780     PRINT L$:
0790   ENDIF
0800   RETURN (L$)
0810 ENDFUNC INDLES$
0820 //
0830 PROC PUSH (HT# ) // ---------- PUSH
0840   //
0850   // Stakker tallet HT# og tæller stakpointer en op.
0860   //
0850   IF PP# < 20 THEN
0860     STAK# (PP# ) :=HT#
0890     PP# :+1
0900   ELSE
0910     PRINT CHR$ (13), "Stakken er fuld"
0920     END
0930   ENDIF
0940 ENDPROC PUSH
0950 //
0960 FUNC POP# // ---------- POP
0970   //
0980   // Henter og returnerer øverste tal i stakken.
0990   // Stakpointer PP# tælles en ned.
1000   //
1010   IF PP# ) 0 THEN
1020     PP# :-1
1030     RETURN (STAK# (PP# ))
1040   ELSE
1050     PRINT CHR$ (13), "Stakken er tom"
1060     END
1070   ENDIF
1080 ENDFUNC POP#
1090 //
1100 //# # # # # # # # # # # # # # # # # # # # # # # # # # # #
1110 // HOVEDPROGRAM
1120 //# # # # # # # # # # # # # # # # # # # # # # # # # # # #
1130 //
1140 TRAP ESC-
1150 PP# :=0
1160 GEMT# :=CHR$ (255)
1170 PRINT "}) ";
1180 EXEC POLSK
1190 TRAP ESC+
1200 END
```

Som de fleste af dette blads læsere har set, er vi i fuld gang med planlægningen af ICL's efterårs-seminar. Vi har modtaget et forrygende antal tilmeldinger, og har derfor planlagt følgende seminarer:

SØNDAG D. 14.9.86 HOTEL
MANDAG D. 15.9.86 LANDSOLDATEN,
NORGESGADE 1,
7000 FREDERICIA.

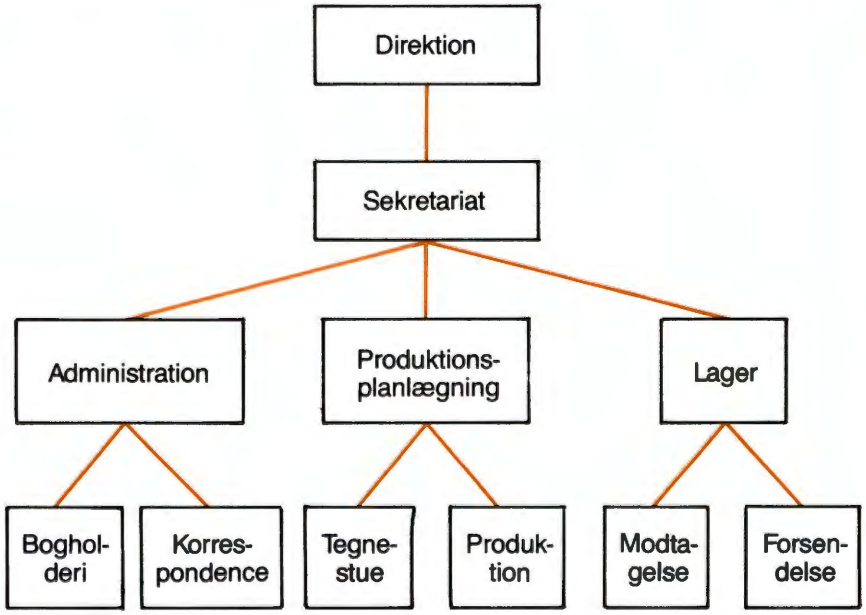
ONSDAG D. 17.9.86 HOTEL HADSUND,
VESTERGADE 4,
9560 HADSUND.

FREDAG D. 19.9.86 HOTEL HVIDE HUS
LØRDAG D. 20.9.86 STRANDVEJEN 111,
4600 KØGE.

TIRSDAG D. 23.9.86 BREGNERØD KRO
BREGNERØD BYVEJ
3520 FARUM.

Virksomhedsspillet:

Der oprettes en lille produktions-virksomhed med følgende struktur:



Sideløbende med virksomheden, afvikles følgende værksteder:

COMCADII	COMUS	KOMMUNIKATION DATABASER

Blandt de indkomne svar har interessen for emner været så bred, at vi, for at imødekomme flest mulige ønsker, har planlagt seminarerne med følgende program:

Kl.

9.00- 9.15: Velkomst – formiddagskaffe

9.15-10.00: Oplæg fra lokal erhvervsperson med emnet: Hvad er erhvervslivets behov for edb-uddannelse af den danske ungdom? Hvor skal niveauet være?

10.00-11.30: Indlæg fra deltagerne.

11.30-12.00: Oplæg til eftermiddagens aktiviteter

12.00-13.00: Frokost

13.00-14.30: Virksomhedsspil/ værksteder (se nedenfor)

14.30-15.00: Kaffepause

15.00-16.15: Virksomhedsspil/ værksteder fortsat

16.15-16.30: Afslutning.

SOFTWARE

HØSTPAKKE 1986

Der er efterhånden tradition for, at vi til hvert nummer af COMETEN dykker ned i vore disketter og finder programmer, som tilbydes læserne på absolut rimelige vilkår (uden beregning).

I december 1985 havde vi julepakken, som stadig kan rekvireres. I sidste nummer af COMETEN udbød vi COMCAD. Måske fik vi ikke helt klart understreget, at man kan rekvirere COMCAD ved at indsende en tydeligt mærket, formatteret diskette (2 stk. hvis 200Kb).

Her til høsten tilbyder vi en ny samling programmer af vidt forskellig art:

SPILOP:
Et program, der forener taltræning og logisk tænkning med et gran af spilleglæde.

ASTORID:
Astoride-spillet kræver næppe en nærmere præsentation. Her er en udgave, der benytter sig af COMET'ens højopløsningsgrafik.

COMPOLSK:
En regnemaskine, der regner med omvendt polsk notation – se artikel andetsteds i bladet. Programmet demonstrerer også stakkens opførsel.

PRTLOAD:
Med dette program kan man oversætte et karaktersæt, fremstillet ved hjælp af COMFONT (julepakken) til karakterer, der (ligeledes ved hjælp af PRTLOAD) kan indlæses i printere som Microline 84, 192 & 193, der har mulighed for brugerdefinerede karakterer.

COMPRINT:
Et generelt printprogram, som bl.a. kan:

- bruges til eksperimenter med printerens forskellige faciliteter.
- det samme som PRTLOAD.
- printe en tekstfil med udnyttelse af printerens faciliteter, herunder også brugerdefinerede karaktersæt.
- printe et grafikbillede (også på en Microline 84!).
- liste en tekstfil på skærmen med mulighed for at blade frem og tilbage (i modsætning til f.eks. TYPE).

FNTTOVEK:
Dette program kan oversætte et karaktersæt, fremstillet ved hjælp af COMFONT (julepakken) til en COMCAD vektorfil. Ofte kan det være nemmere at designe symboler og bogstaver i COMFONT. Med dette program kan man altså oversætte sådanne karaktersæt til at indgå i COMCAD vektortegninger, og dermed få udtegning på plotter. Hvad med at følge Gammel Dansk Standard, og anvende runer/gotisk skrift på tekniske tegninger?

De, der anvender fræseren fra IPC, Viborg tekniske Skole (omtalt i sidste nummer af COMETEN), vil sikkert hilse denne mulighed velkommen i forbindelse med f.eks. fremstilling af skilte.

SEARCH:
Dette er et program, som kan anvendes til at finde givne tegnstrenge i tekstfiler eller i binære programfiler. Sidstnævnte facilitet kan være en hjælp for de, der ønsker at rette i oversatte programmer (f.eks. ovesætte tekster o.l.).

Søgestrenge kan anføres både som tekststrenge og ved hjælp af decimale eller hexadecimale tal, så der er gode muligheder – også for hackere.

På høst-disketten findes også dokumentation og eksempler (listes/udprintes let ved hjælp af COMPRINT).
Ret til mindre ændringer i høstpakkens indhold forbeholdes.
Høst-pakken erhverves uden beregning på følgende måde:

SEND EN FORMATTERET DISKETTE TIL ICL A/S.

Disketten bedes mærket med:
- format (200 ell. 800 Kb)
- afsenders navn/adresse
- HØSTPAKKE

Vedlæg også venligst **adresseret & frankeret** (kr. 3,80) returkuvert/-emballage.
Så returnerer vi disketten med kopi af ovennævnte HØSTPAKKE i løbet af 1-2 uger.

SOFTWARE til COMETEN

For at hjælpe COMET-brugere til et større indblik i, hvad er findes på markedet af undervisningsprogrammer til COMET'en, starter ICL nu en artikelserie, omhandlende 'alt' software, som er udgivet til COMET.

Skulle du imidlertid kende til COMET-programmel, der ikke bliver nævnt, og synes du, de er gode, hører vi meget gerne fra dig.

Kunne du tænke dig at anmelde et eller flere programmer i COMET-avisen, eller har du selv lavet programmer, du gerne vil stille til rådighed for andre, er vi ligeledes interesserede.

I dette nummer findes en liste indeholdende programmer, der er gratis, som kan afvikles på COMET. Desuden findes en liste over programmer, som forhandles af eller via ICL, COMET-afdelingen. Vi vil siden følge op på dette med korte anmeldelser af kommercielle og halvkommercielle programmer.

Fra landscentraler, amtscentraler og pædagogiske centre vil der gratis kunne rekvireres følgende programmel:

NAVN		PROGRAMTYPE
Skinfo:	LK-PLAN	Simulationsprogram
	LK-STAT	Statistisk beskrivelse
	LK-LÆS	Træningsprogrammer til dansk
	LK-TEKST	Tekstbehandlingsprogram
Biologi:	Liv	Simulationsprogram
Dansk:	Diktat	Træn
	Alfabet	Træn
	Dan nogle ord " "	Træn
	Et ord forkert	Træn
	Ordbingo	Værktøj
	Del ord	Træn
	Find vokalen	Træn
Data:	Universalfile	Værktøj
	Boldklub	Indlær, værktøj
	Binær søgning	Indlær
	Blackjack	Spil
	Boblesortering	Indlær
	Editor	Værktøj
	Kalender	Værktøj
	Nim	Indlær, spil
	Store bogstaver	Indlær, værktøj
	Simulering af microprocessor	
	Simulering af datamaskine	
	Adresseprogram	Indlær, værktøj
	Breve	Indlær, værktøj
	Dessert	Indlær
	Doktor	Indlær
	Hjulspil	Indlær, spil
	Karakterbog	Indlæg, værktøj
	Menu program	Værktøj
	Æg	Indlær
	Valgmenu	Værktøj
	Skærmkopi	Værktøj
	Regnebog	Indlær
	Store bogstaver	Værktøj
Engelsk:	Put in the box	Træn
	New words	Træn
	Dictation	Træn
Fysik:	Hus	Simuler
	Kredsløb	Indlær, træn
Hjemkundskab:	Kostberegning	Værktøj
Regn. og mat:	Biorytme	Indlær, værktøj
	Dr. Faktor	Træn., spil
	Kast med terning	Indlær, simuler
	Der mangler et tal	Træn
	Pascals trekant	Simuler
	Pytagoræiske talsæt	Værktøj
	Relation	Indlær, træn
	Sum af to terninger	Indlær simuler
	Terninger	Indlær, simuler
	Hvem kommer først til Dronningborg	Træn + Spil
	Hvem kommer nærmest	Træn + Spil
	Addition 1	Indlær, træn
	Addition 2	Indlær, træn
	Addition 3	Indlær, træn
	Addition 4	Indlær, træn
	Addition 5	Indlær, træn
	Addition 6	Indlær, træn
	Addition 7	Indlær, træn
	Addition 8	Indlær, træn
	Addition 9	Indlær, træn
	Opgaveark Addition	Værktøj
	Opgaveark Substrakton	Værktøj
	Opgaveark Multiplikation	Værktøj
	Opgaveark Division	Værktøj
	Opgaveark Omsætning	Værktøj
	Kædealgoritmer	Værktøj
Samtid:	Befolknings tilvækst	Simuler
	CPR-nr. Generator	Simuler, værktøj
	Diktator	Simuler
	Monetarist	Simuler
	Grafik ill. af talmateriale	Værktøj
Øvrige:	Super Cloz	Spil
	Div. spil	
	Kommunikation	
	I base-læs	Værktøj
	Skriv	Værktøj

Fortegnelse over programmel til COMET, der forhandles af ICL A/S

PROGRAMMERING:
HH-Basic (Control-basic)
Basic fortolker
Basic oversætter
Comal-80 (Metanic)
Comal-80/Z80
DDE-Comal
Genix-Comal
Compas-pascal
Poly-pascal
Genix-pascal
Microsoft Cobol
Microsoft Fortran
Z80-macroassembler
Assembler plus-tools
HH-screen + HH-format

CAD/CAM:
Comcad
ComcadII

3D TEGNING:
Monster
TZ2 Digitaliseringsprogram.
Rukfly Sammenkoblingsprogram
Netmodel

TEKSTBEHANDLING:
Comtekst
HH-Protekst
Wordstar + Starindex

REGNEPROGRAMMER:
SupercalcII (avanceret regneark)
Myresnak
Skomal

ADMINISTRATION og DATABASE:
Office Power (efteråret 86)
Comet kontor (fra sept. 86)
HH-karto (kartoteksprogram)
DbaseII (databasesystem)
Polyfile (Databasesystem til PolyPascal)
Combase (fra august-86)

KOMMUNIKATION:
ComKom
HH-Dialog
HH-Skema

KOPIERING:
Comdisk
HH-Copy

FORFATTERSYSTEM:
Comus
Commenu

Endvidere findes en række større og mindre hjælpe- og serviceprogrammer.

EDB OG SAMFUND:

Er et system bestående af en diskette, en grundbog og en arbejdsbog samlet i et hele.

På programdisketten findes en metodisk vejledning til det samlede materiale, ligesom programmernes brugerflade gennemgås.

Systemet beskriver:

Hvordan sikrer man sig mod uautoriserede personers eller organisationers adgang til offentlige som private registre.

Hvad er samkøring af registre. – Brug og misbrug.
Brug og misbrug af EDB-behandlede prognoser.
Hvordan arbejder hackers'.

Systemet er et åbent system, hvor det er brugerne, der bestemmer, hvad programmerne skal lave. Det er eleverne der spørger datamaten, – ikke omvendt.

EDB og Samfund er lavet således, at eleverne eller lærergrupperne lærer ved at spille mod hinanden.

Der bliver kun anvendt datamaskine i en begrænset del af tiden, og skal materialet anvendes i datalærerundervisningen vil det derfor være nødvendigt at supplere med andet.

EDB og SAMFUND lægger op til, at undervisningsformen vil kunne være en kombination af såvel klasseundervisning som gruppearbejde.

VI HAR MODTAGET

COMET-afdelingen modtager med jævne mellemrum programmel, udviklet af personer rundt om i landet. Disse programmer kan i visse tilfælde også komme andre til nytte.

Vi vil gerne her i COMETEN udbrede kendskabet til programmerne.

SOLEDA

SOLEDA er et kartoteksprogram, fremstillet i Compas Pascal, og til brug af ledere i fritidsundervisningen til styring af:

1. Elever/lærere/hold
2. Udskrifter af kursusbekræftelser, protokol & labels.
3. Statistik.

I datalære-undervisningen kan SOLEDA anvendes som et godt eksempel på informationsbehandling. Der er bruger-kontrol ved hjælp af password og ved indlæsning/rettelse af elev-, lærer- og holddata er der fuld editering.

Sortering og søgning af elever/lærere foregår ved hjælp af index-variable.

En udvidet version til maskiner med 2 diskettestationer er under udarbejdelse, og forventes at være færdig ultimo oktober '86.

Skulle du være interesseret i yderligere oplysninger omkring SOLEDA KARTOTEKS-PROGRAM kan henvendelse ske til:

Ole Wrang, Skovvænget 20, 6100 Haderslev, Tlf. 04-52 99 55, eller til ICL A/S, COMET-afdelingen.

KONVERTERING AF DISKETTER

Nu kan vi tilbyde at konvertere disketter fra et vilkårligt (stort set) format til COMET-format. Naturligvis kan vi også tilbyde konvertering mellem forskellige COMET-formater (40 spor, 160 spor – 5¼" og 3½").

Ved indkøb af programmel, der f.eks. annonceres i diverse tidsskrifter, kan man altså forudsætte, at konvertering til COMET-format kan ske ved henvendelse til os.

Indsend disketterne til ICL, COMET-afdelingen med så mange oplysninger som muligt:

- formatets navn
- den såkaldte skrivefaktor
- hvilket COMET-format, der ønskes konverteret til.

Det er også en hjælp for os, hvis disketten indeholder en tekstfil med nummererede linier.

Priserne for konvertering er følgende:

Grundgebyr pr. gang:	kr. 100,-
Pris pr. diskette, der konverteres	kr. 50,-
Pris pr. diskette, der leveres:	
5¼", 40 spor	kr. 35,-
5¼", 160 spor	kr. 40,-
3½", 160 spor	kr. 70,-

Du er også velkommen til at medsende tomme, formatterede disketter (COMET-format). Så falder prisen for levering af disketter naturligvis bort.

NYT PROGRAMMEL

C

C-compiler

Vi har fundet en C-compiler – MIX C-compiler – til CP/M, som vi kan anbefale og tilbyde til COMET-brugere til en helt usædvanlig fordelagtig pris.

Compileren opfylder C-standarden og den indeholder også en del ekstra faciliteter (kald af CP/M-rutiner, assembler rutiner m.v.).

Vi har afprøvet MIX C og har bl.a. konstateret, at C-programmer udviklet med denne compiler under CP/M umiddelbart kan overføres til GENIX på COMET 32.

Programpakken omfatter:

- C-compiler
- biblioteker og run-time pakke
- linker
- overlay-loader
- optimeringsprogrammer, der optimerer m.h.t. størrelse og hastighed
- en skærmorienteret editor, som bl.a. giver mulighed for at arbejde med split-screen.

For de, der ikke interesserer sig for C, kan programpakken være attraktiv alene af hensyn til editoren.

Sammen med MIX C følger en omfattende lærebog og manual (ca., 450 sider – engelsk).

Hele denne program- & bogpakke kan vi tilbyde til en pris à kr. 1.150,- excl. moms.

DATABASER

for COMET-brugere

For mange COMET-brugere er databaser nok et relativt ubekendt begreb. Men her ligger faktisk en guldgrube af muligheder for ny udnyttelse af COMET'en.

Hvad er en database?

En database er i sin simpleste form blot en samling af data (f.eks. et navneregister) i en computer, et eller andet sted i verden. Disse data er gjort tilgængelige for andre via f.eks. en telefonlinie. Brugeren kan dermed, ved at slutte et modem til sin egen computer og telefonnettet, komme i forbindelse med databasen og trække oplysninger ud af denne. Blot behøves et program i brugerens computer, der kan kommunikere med modem'et.

Der findes i dag mange forskellige databaser, dækkende vidt forskellige aspekter – lige fra telefonbogsoplysninger til brevudveksling. Men desværre snakker de ikke alle "det samme sprog"; der skal kommunikeres på forskellig vis med baserne. Dette skal brugeren være klar over, før der evt. investeres i modem. De enkelte baser er normalt beskyttede via kodeord, så uautoriserede personer ikke uden videre kan få adgang til databasen. De fleste baser drives forretningsmæssigt, så der skal betales for evt. tidsforbrug/abonnement. Men der findes også databaser, som COMET-brugere uden videre kan kalde op, eller blot behøver at blive registreret i – og **derefter** frit kan trække oplysninger fra.

I COMET-afdelingen har vi kendskab til nogle enkelte databaser:

1. 0036

Telefonvæsenets oplysningsdatabase.
Indeholder alle telefonbogs-oplysninger og kan kaldes af alle. Kan kommunikere på flere måder (forskellige hastigheder (300/300, 75/1200 og 1200/1200 bagud).
For yderligere oplysninger kontakt JTAS på tlf. 06-29 33 66, lok. 3703.

2. POLTEKST

Politikens database.
Indeholder artikler bl.a. fra Politiken og Ekstra-Bladet. Abonnement skal tegnes, timeafgift betales.
For yderligere oplysninger kontakt Politikens Hus på tlf.: 01-11 85 11.

3. DATABOX

Postvæsenets databank/mail.
Mulighed for elektronisk post.
Der skal tegnes abonnement.
For yderligere oplysninger kontakt P & T.

4. CPI's opslagstavle

Dansk Center for Pædagogik & Informatik's elektroniske opslagstavle og udvekslingsbank.
Er gratis at anvende, men brugeren skal registreres.
For yderligere oplysninger kontakt CPI på tlf.: 01-23 71 11

5. TELEDATA

Det private erhvervslivs mulighed for at formidle oplysninger til brugerne. Administreres af telefonselskaberne og P & T. Indeholder et "væld af oplysninger".

Abonnement skal tegnes, og forbrug betales. For yderligere oplysninger kontakt en af de danske tele-administrationer.

6. BASIS

Bibliotekscentralens og Kommunedatas litteratursøgningsdatabase.
Abonnement skal tegnes.
For yderligere oplysninger kontakt Bibliotekscentralen på tlf.: 02-97 55 55.

7. ALIS

Danmarks Tekniske Biblioteks litteratur-database. Kan frit anvendes, brugeren skal registreres.
For yderligere oplysninger kontakt DTB på tlf.: 02-88 30 88.

Der findes sandsynligvis mange flere databaser. Har I kendskab til andre, bringer vi meget gerne disse oplysninger videre her i COMETEN.

Skriv til ICL, COMET-afdelingen, og forklar evt. ganske kort, hvad databasen bruges til, og hvordan yderligere oplysninger kan skaffes.

ÅBEN DATASTUE

Teknologi- og Informatikcenteret, København, havde i hele juli måned arrangeret en såkaldt »Åben Datastue« midt i København's centrum.

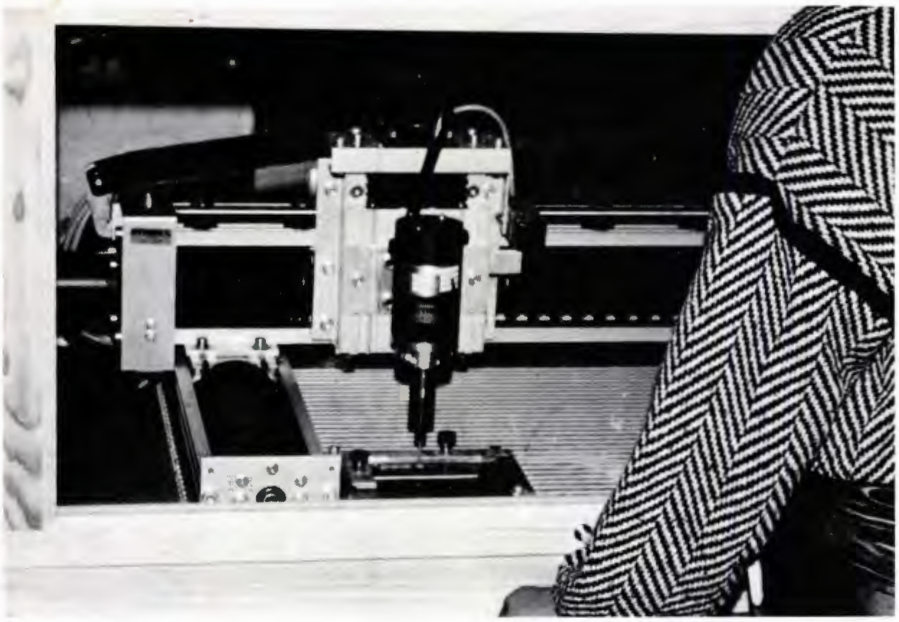
Formålet var, at både børn og voksne i ferietiden skulle have mulighed for at »lege« sig frem til større viden om EDB.

Datastuen blev en stor succes – det formelig vrimlede med mennesker dagen lang.

Bl.a. ICL's COMET'er var et stort trækplaster. Disse var opstillet til meget forskellige anvendelser, som f.eks. tegningskonstruktion/tekstbehandling/regneprogrammel m.v.

Specielt tiltrak COMET'en, der styrede en fræsemaskine, sig stor interesse. Der blev i løbet af måneden produceret mange navneskilte i plast.

Men også COMET-styrede kraner & transportbånd, opbygget af de besøgende af LEGO-moduler, var et tilløbsstykke.



Det Københavnske Tegneri. 9/86. 2000

AUTOBAUDSELECTOR

Har man ofte behov for at skifte baudrate på COMET 3400, kan der nu leveres en autobaudselector, således at man undgår at skulle stille på baudrateswitchene på MPS-27-kortet.

Autobaudselectoren er fremstillet i samarbejde med Viborg Tekniske Skole, og det er et lille printkort, som monteres på det eksisterende MPS-27 kort (d.v.s. det gælder kun for 4 Mhz maskiner!).

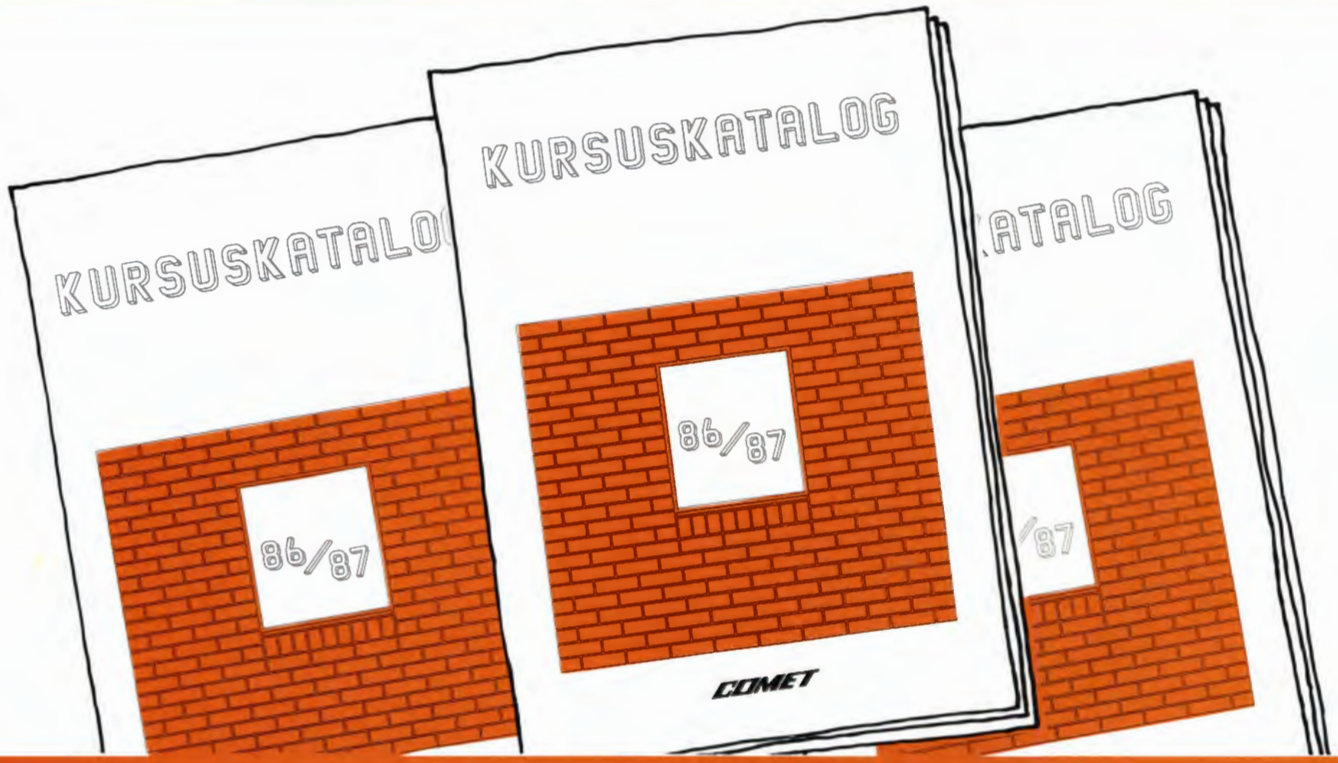
Med autobaudselectoren monteret kan man herefter rent softwaremæssigt sætte baudraten til ethvert tænkeligt forhold mellem 75 baud og 19200 baud.

Monteringen af autobaudselectoren foregår meget let, og selvom printet monteres på MPS-27 kortet vil MPS-27 kortet herefter alligevel ikke optage mere end 1 position i kortmagasinet på COMET 3400.

Autobaudselectoren kan rekvireres ved henvendelse til:

ICL A/S
Gladsaxevej 372
2860 Søborg
Tlf. 01 67 97 00

Prisen er kr. 500,- excl. moms og der følger monteringsvejledning og programeksempel med således at autobaudselectoren er klar til brug.



COMET-afdelingen tilbyder også i denne sæson afholdelse af kurser til COMET-brugere.

Disse kurser spænder lige fra det mest basale begynder-niveau, hvor brugeren bliver gjort bekendt med maskinel og styresystem (COMET grundkursus) – og til helt avancerede programmerings-kurser (GENIX). Nedenfor er nævnt ICL's COMET-kurser samt varighed, for sæsonen 1986/-87:

KURSUSTYPE:	VARIGHED
COMET grundkursus	½–1 dag
COMET flerbrugersystem	1 dag
Generelt om COMET	1 dag
COMAL-80 programmering	2 dage
CP/M operativsystemet	1 dag
ASSEMBLER programmering	2 dage
COMUS undervisningssystem	1-2 dage

KURSUSTYPE:	VARIGHED
COMUS med interaktiv video	1 dag
COMTEKST tekstbehandlingssystem	2 dage
COMCAD tegningskonstruktion	1 dag
SuperCalc2 regneprogram	1 dag
GENIX (UNIX)	3 dage
Introduktion til programmering med C.	3 dage
COMCAD II	2×3 dage

Yderligere oplysninger samt kursuskatalog vedrørende ICL's COMET-kurser kan rekvireres ved henvendelse til ICL, COMET-afdelingen, tlf.: 01-67 97 00.

Det skal bemærkes, at kursus i COMCAD II afholdes af Århus Tekniske Skole. Yderligere oplysninger ved henvendelse til:

INFORMATIKCENTERET, Århus Tekniske Skole, Halmstadgade 6, 8200 Århus N.

Der findes efterhånden et utal af forskellige vejledninger til hjælp for COMET-brugere – og der kommer stadig nye til.

Nedenfor findes en total liste over disse meget forskellige skrifter. Endvidere ligger COMET-afd. endnu inde med en del vejledninger til COMPAS PASCAL. Disse kan stadig rekvireres for kr. 50,-

BRUGERVEJLEDNINGEN/MANUALER vedr. COMET Mikrodatamaten og dens programmer:

Titel		Antal sider	Pris excl. moms
COMET	Brugervejledning (3000 & 3400)	80	kr. 200,-
COMET 3400	Flerbrugersystem, ver. 3.00-3.20	30	kr. 100,-
MPS-07	Digital/Analog konverter	16	kr. 50,-
MPS-08	Analog/Digital konverter	23	kr. 50,-
MPS-10	Input/Output modul	23	kr. 50,-
MPS-17	Eprombrænder. Indeholder endvidere manual for COMPROM	13	u/b
MPS-24	Grafikmodul incl. COMGRAPH (grafik i COMAL-80	53	kr. 100,-
MPS-25	Realtidsur-modul	15	kr. 50,-
MPS-28	192Kb RAM-modul	5	u/b
MPS-28A	512/768Kb RAM-modul	4	u/b
MPS-32	EPROM-bank	6	u/b
COMAL-80	ver. 2.0 (SPECIALTILBUD)	160	kr. 50,-
COMAL-80/Z80	Brugervejledning	186	kr. 200-
POLY PASCAL	Brugervejledning		kr. 200,-
COMCAD	ver. 2.0	123	kr. 150,-
COMUS	Datamatformidlet undervisning	63	kr. 100,-
MYRESNAK	ver. 1.1, indeholdende vejledning for SKILPADDE	40	kr. 75,-
COMTEKST	ver. 2.06, m. rettelsesblad for printerinstallation	58	kr. 100,-
COMET	HJÆLPEPROGRAMMER, ComDisk, ComMenu, ComKom	20	kr. 70,-
Tilbehørsprogrammer til Monster:			
	TZ-2	11	u/b
	Rukfly	9	u/b
	Netmodel	9	u/b
GENIX	Brugervejledning	50	kr. 100,-
GENIX	Programmers manual, 5 bind, Vol. 1, Section 1-8	ca. 500	kr. 1000,-
GENIX PASCAL	Brugervejledning	ca. 60	kr. 100,-
MUS/JOYSTICK	Brugervejledning for COMET 3400	3	u/b
EDB-ORDBOG		110	kr. 80,-

KORTE BESKRIVELSER/VEJLEDNINGER vedr. COMET Mikrodatamaten og dens programmer:

Titel	alle u/b	Antal sider
Kapitel 6	til COMET brugervejledning Datablade for Z-80 PIO MC6845 Video & MC6850 Kommunikationskontroller	55
Seriel printer	(tilføjelse til COMET manual). Benforbindelser når "2" vælges i startmenu	1
Portnumre	Anbefalede portnumre til forsk. moduler	1
Baudrate	Strapning af MPS-27	1
SUBMIT	– Et eksempel	1
Autoload	Frembringelse af en autoload diskette, beskrevet trin for trin	3
COMAL-80	& ASSEMBLER. Brug af maskinkode ved COMAL-80, ver. 2.02	2
BREDE BOGSTAVER I COMAL-80		
Mere end 2 disk-enheder i COMAL-80, ver. 2.02a		
	Ændringer, der skal foretages, når COMAL-80 anvendes på 3 eller flere disketteenheder. Beskrevet trin for trin	3
Brug af	SKÆRM- & ATTRIBUTRAM I COMAL 80/Z80	2
Monteringsvejledninger:		
	MPS-24	4
	– EPROM (CPU) t. COMET 3400/3000	3
	– KARAKTEREPROM i COMET 3400	5
	– Danske karakter på MP1000 Plotter	11
	– EPROM i MP1000 Plotter	4
	– MPS-16, understregnings-eprom i COMET 3000,	11
Printerinstallation i COMTEKST		7
Installationsvejledning COMUS-flerbruger		3
COMET	Stik & Kabler	10
OKI 182	Printer. Kort vejledning	3
OKI 192/193	Printer. Kort vejledning	4
Printerbuffer	Manual for 64Kb printer/plotter buffer	3

RETTELSESBLAD

COMETEN

Nr. 2. Sept. 1986

Der har desværre indsneget sig et par meningsforstyrrende fejl på siderne 3 & 4, samt i programmet side 6.

Side 3: I 1. spalte, nederste halvdel er et par linier faldet ud. Der skal stå:

I COMAL-80 kan vi f.eks. oprette
en stak til lagring af heltal:

```
DIM STAK# (0:19)
```

som altså kan indeholde op til
20 heltal nummereret 0 til 19.

Side 3: Teksten øverst i den nederste tegning hører til figurerne ovenover, og burde retteligen have været:

```
Vi starter med en tom stak
EXEC PUSH (285)
EXEC PUSH (517)
A:= POP() + POP()
```

eller

```
EXEC PUSH (285)
EXEC PUSH (517)
EXEC PUSH (POP() + POP())
```

Nederste del af tegningen (altså stakkene) hører sammen med teksten i nederste del af 1. spalte på side 4.

Side 6: Det er altid svært at få sat programudskrifter - måske fordi sættemaskiner i dag er programstyret - derfor er bagsiden af dette RETTELSESBLAD påtrykt en direkte udskrift af programmet.

```
0010 DIM STAK#(0:19)
0020 DIM TALSTR$ OF 20
0030 DIM RETURVAL$ OF 1, C$ OF 1, L$ OF 1
0040 DIM GEMT$ OF 1
0050 //
0060 PROC POLSK // ------------------------------------ POLSK
0070   //
0080   // I denne procedure styres indlæsning, stakning, udregning og udskrift
0090   //
0100   REPEAT
0110     RETURVAL$:=GETOP$()
0120     CASE RETURVAL$ OF
0130     WHEN "Q", "q" //                 Q stopper programmet
0140       END
0150     WHEN CHR$(255) //                Tal indlæst
0160       TAL#:=IVAL(TALSTR$)
0170       EXEC PUSH(TAL#)
0180     WHEN "+" //                      Addition
0190       EXEC PUSH(POP#()+POP#())
0200     WHEN "-" //                      Subtraktion
0210       OPERAND2:=POP()
0220       EXEC PUSH(POP()-OPERAND2)
0230     WHEN "*" //                      Multiplikation
0240       EXEC PUSH(POP#()*POP#())
0250     WHEN "/" //                      Division
0260       OPERAND2#:=POP#()
0270       IF OPERAND2#=0 THEN
0280         PRINT CHR$(10),CHR$(13),"Division med 0"
0290         PP#:=0 //                   Stakken tømmes ved division med 0
0300         PRINT ">> ";
0310       ELSE
0320         EXEC PUSH(POP#() DIV OPERAND2#)
0330       ENDIF
0340     WHEN "=" //                      Resultatet udskrives
0350       PRINT "  ",POP#()
0360       PRINT ">> ";
0370     WHEN "C", "c"
0380       PP#:=0 //                     Regnemaskinen tømmes
0390       PRINT CHR$(13),CHR$(10);">> ";
0400     OTHERWISE
0410       //
0420     ENDCASE
0430   UNTIL RETURVAL$="Q" OR RETURVAL$="q"
0440 ENDPROC POLSK
0450 //
0460 FUNC GETOP$ //------------------------------------ GETOP$
0470   //
0480   // Denne funktion indlæser næste operand eller operator
0490   //
0500   REPEAT
0510     C$:=INDLES$()
0520   UNTIL C$<>CHR$(32) //      Overspring blanke
0530   IF C$>="0" AND C$<="9" THEN
0540     TALSTR$:=C$
0550     REPEAT
0560       C$:=INDLES$()
0570       IF C$>="0" AND C$<="9" THEN TALSTR$:+C$
0580     UNTIL C$<"0" OR C$>"9"
0590     GEMT$:=C$ //              Sidst indlæste tegn gemmes til næste kald
0600     RETURN (CHR$(255)) //     Der returneres 255, hvis operand indlæst
0610   ELSE
0620     RETURN (C$) //            ellers returneres indlæst karakter
0630   ENDIF
0640 ENDFUNC GETOP$
0650 //
0660 FUNC INDLES$ //---------------------------------- INDLES$
0670   //
0680   // Indlæsning af tegn via adresse 256
0690   //
0700   IF GEMT$<>CHR$(255) THEN // Først undersøges, om der er et gemt tegn
0710     L$:=GEMT$
0720     GEMT$:=CHR$(255)
0730   ELSE
0740     POKE 256, 255
0750     REPEAT
0760       L$:=CHR$(PEEK(256))
0770     UNTIL L$<>CHR$(255)
0780     PRINT L$;
0790   ENDIF
0800   RETURN (L$)
0810 ENDFUNC INDLES$
0820 //
0830 PROC PUSH(HT#) //-------------------------------------- PUSH
0840   //
0850   // Stakker tallet HT# og tæller stakpointer en op.
0860   //
0870   IF PP#<20 THEN
0880     STAK#(PP#):=HT#
0890     PP#:+1
0900   ELSE
0910     PRINT CHR$(13),"Stakken er fuld"
0920     END
0930   ENDIF
0940 ENDPROC PUSH
0950 //
0960 FUNC POP# //----------------------------------------- POP
0970   //
0980   // Henter og returnerer øverste tal i stakken.
0990   // Stakpointer PP# tælles en ned.
1000   //
1010   IF PP#>0 THEN
1020     PP#:-1
1030     RETURN (STAK#(PP#))
1040   ELSE
1050     PRINT CHR$(13),"Stakken er tom"
1060     END
1070   ENDIF
1080 ENDFUNC POP#
1090 //
1100 //############################
1110 // HOVEDPROGRAM
1120 //############################
1130 //
1140 TRAP ESC-
1150 PP#:=0
1160 GEMT$:=CHR$(255)
1170 PRINT ">> ";
1180 EXEC POLSK
1190 TRAP ESC+
1200 END
```