BØRGE R. CHRISTENSEN

COMAL 2

(TILLÆGSMANUAL TIL
EXTENDED BASIC USER's MANUAL)

UDGIVET AF

A/S REGNECENTRALEN

I SAMARBEJDE MED DATAAFDELINGEN VED TØNDER SEMINARIUM

COMAL II er anden version af sproget COMAL, beskrevet af undertegnede i samarbejde med lektor Benedict Løfstedt DAIMI, Aarhus Universitet. COMAL II er implementeret til kørsel på RC 7000, SOS og RDOS, af Per Christiansen og Knud Christensen ved Tønder Statsseminariums Dataafdeling. Som værtssprog er benyttet DGC's X.BASIC, revision 08, der er indeholdt i COMAL II som et ægte delsprog.
For at få en fuldstændig beskrivelse af COMAL II skal man læse nærværende tillægsmanual i sammenhæng med DGC's EXTENDED BASIC USER'S MANUAL (093-000065-04).

Tønder, februar 1976.

Børge R. Christensen.

1. Variabelnavne.

I COMAL II angives variable af alle typer efter forskriften:

$$x_1 x_2 \ldots x_i,$$

hvor $i \leq 8$. Tegnet x_1 skal være et bogstav, og tegnene $x_2, \ldots, x_i$ skal være bogstaver eller cifre.

Eksempel.

```
LET RENTE=KAPITAL*RENTEFOD*ANTDAGE/36000
LET NAVN$="OLE OLSEN"
LET TABEL(NUMMER,AARG)=126
```

□

2. Tildelinger.

LET-sætninger kan opbygges således:

LET $\underline{var}_1=\underline{udtryk}_1$; $\underline{var}_2=\underline{udtryk}_2$; ...; $\underline{var}_n=\underline{udtryk}_n$

hvor $\underline{var}_1$, $\underline{var}_2$, ..., $\underline{var}_n$ er en række variabelnavne, og $\underline{udtryk}_1$, $\underline{udtryk}_2$, ..., $\underline{udtryk}_n$ er konstanter, variable eller formler. En LET-sætning kan indeholde så mange tildelinger, som linjelængden tillader.

Eksempel.

```
LET RENTEFOD=12; KAPITAL=15000; ANTDAGE=145
LET TAL=45; ANTAL=10; NAVN$="ANTON OLSEN"
LET SIDST=I*25; FØRST=SIDST-24; I=I+1
LET TELLER1=0; TELLER2=0; TELLER3=1
```

□

3. Styrestrukturer.

Ud over styrestrukturerne i BASIC findes følgende i COMAL II:

(1) IF p THEN DO .. ELSE DO .. ENDIF

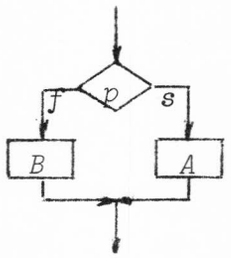
(2) IF p THEN DO .. ENDIF

(3) REPEAT .. UNTIL p

(4) WHILE p DO .. ENDWHILE

(1) er opbygges således:

```
IF p THEN DO
  +-----------+
  |     A     |
  +-----------+
ELSE
  +-----------+
  |     B     |
  +-----------+
ENDIF
```



Idet p er et Boolsk udtryk (et åbent udsagn), virker styresætninger således: Hvis p har værdien sand, udføres programdelen A mellem IF p THEN DO og ELSE, og hvis p har værdien falsk, udføres programdelen B mellem ELSE og ENDIF. I begge tilfælde fortsættes programudførelsen med linjen efter ENDIF.

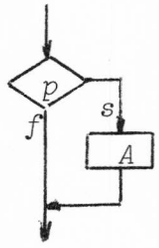
Eksempel.

```
LET ENHPRIS=13.85
INPUT "ANTAL SOLGTE ENHEDER: ",ENHEDER
INPUT "HAR KUNDEN KONTO HOS OS (JA=1, NEJ=0): ",KONTO
LET SALGSPR=ENHEDER*ENHPRIS
IF SALGSPR<100 AND KONTO=0 THEN DO
  PRINT "KUNDEN SKAL BETALE: "; SALGSPR+15;"KR."
ELSE
  PRINT "KUNDEN SKAL BETALE: "; SALGSPR;"KR."
ENDIF
□
```

Ved listning af programmet sker automatisk indrykning af teksten mellem IF p THEN DO og ELSE og mellem ELSE og ENDIF (jvf. eksemplet).

(2) er opbygget således:

```
IF p THEN DO
  [----------]
  [    A     ]
  [----------]
ENDIF
```



Idet p er et Boolsk udtryk, virker styresætningerne således: Hvis p har værdien sand, udføres programdelen A, og derpå fortsættes udførelsen af programmet med linjen efter ENDIF. Hvis p har værdien falsk, udføres A ikke, men udførelsen af programmet fortsættes uden videre med linjen efter ENDIF.

Eksempel.

```
READ TAL1, TAL2
IF TAL1>TAL2 THEN DO
  LET GEM=TAL1; TAL1=TAL2; TAL2=GEM
ENDIF
PRINT TAL1, TAL2
□
```

Ved listning af programmet sker automatisk indrykning af teksten mellem IF p THEN DO og ENDIF.

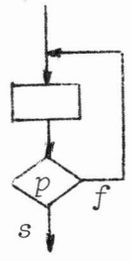
Eksempel.

```
INPUT "ANTAL KOPIER: ",ANTAL
IF ANTAL>=6 OR ANTAL<=25 THEN DO
  KOPIPRIS=180+(ANTAL-5)*15
ELSE
  IF ANTAL<6 THEN DO
    KOPIPRIS=36*ANTAL
  ELSE
    KOPIPRIS=480+(ANTAL-25)*6
  ENDIF
ENDIF
PRINT "DER SKAL BETALES:";KOPIPRIS/100;"KR."
□
```

I COMAL II kan man have op til 7 IF p THEN DO .. (ELSE) .. ENDIF forgreninger inden i hinanden.

(3) er opbygget således:

```
REPEAT
  +----------+
  |    A     |
  +----------+
UNTIL p
```



Idet p er et Boolsk udtryk, virker styresætningerne således: Programdelen A gentages, indtil p har værdien sand. Når p er sand, fortsættes programudførelsen med linjen efter UNTIL p.
Man bør være opmærksom på, at programdelen A altid udføres mindst én gang, hvis fortolkeren når frem til sætningen REPEAT, idet den egentlige styring er anbragt sidst i forløbet.

Eksempel.

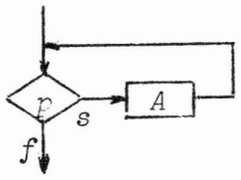
```
LET KAPITAL=0; TERMIN=0
INPUT "HVOR MEGET VIL DE INDBETALE PR. TERMIN? ",INDBET
REPEAT
  LET KAPITAL=KAPITAL*103/100+INDBET; TERMIN=TERMIN+1
UNTIL KAPITAL>=25000
PRINT "OPSPARINGEN VARER: ";TERMIN;"TERMINER."
□
```

Ved listning af programmet sker automatisk indrykning af teksten mellem REPEAT og UNTIL p.
I COMAL II kan man have op til 4 REPEAT .. UNTIL p løkker inden i hinanden.
I de enkelte løkker kan man have forgreninger, uanset på hvilket niveau løkken findes. Hvis man har forgreninger inden i løkker, som selv er dele af forgreninger, skal man blot sørge for, at der ikke bliver mere end ialt

4 REPEAT-løkker og 7 IF-forgreninger i dybden.

(4) er opbygget således:

```
WHILE p THEN DO
  +----------+
  |    A     |
  +----------+
ENDWHILE
```



Idet p er et Boolsk udtryk, virker styresætningerne således: Programdelen gentages, sålænge p har værdien sand. Hvis p er falsk, fortsættes udførelsen af programmet med linjen efter ENDWHILE.
Det kan bemærkes, at hvis p har værdien falsk, når udførelsen af programmet kommer frem til WHILE-sætningen, udføres A overhovedet ikke, men udførelsen fortsættes uden videre med sætningen efter ENDWHILE.

Eksempel.

```
INPUT "ANTAL AEG: ",ANTALAEG
LET ANTBAKKR=0
WHILE ANTALAEG>=6 THEN DO
  LET ANTALAEG=ANTALAEG-6; ANTBAKKR=ANTBAKKR+1
ENDWHILE
PRINT "DER BLEV PAKKET: ";ANTBAKKR;"BAKKER MED AEG."
PRINT "OG DER BLEV";ANTALAEG;"AEG TIL OVERS."
□
```

Ved listning af programmet sker automatisk indrykning af teksten mellem WHILE p THEN DO og ENDWHILE.
I COMAL II kan man have indtil 4 WHILE .. ENDWHILE-løkker inden i hinanden.

I COMAL II findes altså ialt 3 løkkestrukturer: FOR .. NEXT, REPEAT .. UNTIL og WHILE .. ENDWHILE, og man kan have indtil 4 løkker af hver type inden i hinanden, uafhængig af rækkefølgen. Endvidere kan kan man på et

hvilket som helst niveau have IF .. (ELSE) .. ENDIF forgreninger i løkkerne, blot man ser efter, at der ikke er mere end sammenlagt 7 forgreninger i samme rede.

Det skal bemærkes, at brugen af GOTO-sætninger i forbindelse med styrestrukturerne (1) - (4) kan være problematisk. I almindelighed bør man ikke bruge GOTO's inde i de i oversigten nævnte programdele A og B, hvis disse GOTO's henviser til sætninger, der ligger uden for den pågældende styrestrukturs begyndelses- eller slutsætning.

Eksempel.
Følgende program er uproblematisk (men ikke videre smart):

```
010 INPUT "ANTAL KOPIER: ",ANTAL
020 IF ANTAL>=6 AND ANTAL<=25 THEN DO
030   KOPIAFG=180+(ANTAL-5)*15
040 ELSE
050   IF ANTAL<6 THEN GOTO 80
060   KOPIAFG=480+(ANTAL-25)*6
070   GOTO 90
080   KOPIAFG=36*ANTAL
090 ENDIF
100 PRINT "ANTAL KOPIER: ";ANTAL;" AFGIFT: ";KOPIAFG/100
```

Følgende program vil give anledning til en fejlmelding under kørslen, hvis der optræder mere end 7 navne:

```
010 DIM NAVN$(50)
020 INPUT "NAVN: ",NAVN$
030 LET KARSUM=0; ANTAL=0
040 IF NAVN$<>"SLUT" THEN DO
050   INPUT "KARAKTER: ",KAR
060   LET KARSUM=KARSUM+KAR; ANTAL=ANTAL+1
070   IF ANTAL=9 THEN GOTO 110
080   GOTO 50
090 ENDIF
100 STOP
110 PRINT "GENNEMSNIT FOR ";NAVN$;" ER: "; KARSUM/9
120 GOTO 20
```

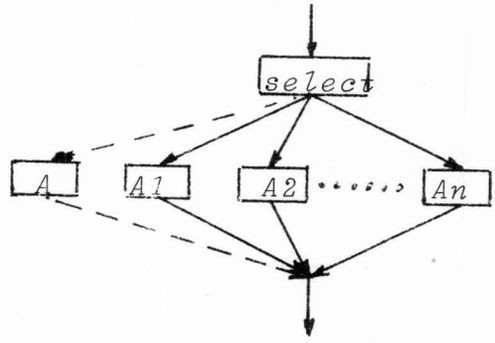
Hver gang, der er indtastet 9 karakterer, vil der ske et hop fra linje 70 til linje 110. Herved bliver ENDIF overspunget, og fortolkeren registrerer ikke, at blokken IF .. ENDIF er udført. Hvis dette sker mere end 7 gange, vil systemet opfatte det, som om der var mere end 7 IF .. ENDIF forgreninger inden i hinanden, hvilket ikke er tilladt i COMAL II.

□

Brugen af GOTO bør i det hele taget undgås. Hvis det forekommer programmøren vanskeligt at undgå GOTO, vil dette som regel skyldes, at vedkommende ikke er blevet fortrolig med tankegangen bag COMAL II's styrestrukturer.

Der findes endnu en styrestruktur i COMAL II, nemlig en multiforgrening, som er opbygget således:

```
SELECT udtryk THEN DO
  [    A    ]
CASE <liste af hele tal>1
  [    A1   ]
CASE <liste af hele tal>2
  [    A2   ]
...
CASE <liste af hele tal>n
  [    An   ]
ENDCASES
```



Styresætningerne virker således: udtryk er en variabel eller en formel, som kan antage heltallige værdier. Hvis den værdi, der tildeles udtryk findes i en liste efter et CASE, udføres den programdel A_i, som er anført mellem det udpegede CASE og det efterfølgende CASE, eller - hvis det udpegede CASE er det sidste i rækken - mellem det udpegede CASE og ENDCASES. Derefter fortsættes udførelsen af pro-

grammet med sætningen efter ENDCASES. Der kan indsættes et delprogram A mellem SELECT-sætningen og den første CASE-sætning. Hvis værdien af udtryk ikke er anført i nogen liste efter CASE, vil delprogrammet A blive udført og derefter sætningen efter ENDCASES. Hvis delprogrammet A ikke findes, og der ikke udpeges noget CASE, fortsættes uden videre med sætningen efter ENDCASES.
Der er ingen begrænsninger i rækken af CASEs efter SELECT, og man kan have så mange SELECT .. ENDCASES inden i hinanden, som man ønsker.
Ved listning af programmet sker automatisk indrykning af alle teksterne til A, A_1, A_2, ..., A_n.

Eksempel.

```
ON ESC THEN LET FYRAFTEN=1
REPEAT
  INPUT "KØB (TAST 1), SALG (TAST 2), INGEN AF DISSE (TAST 0)",STYR
  IF STYR=0 THEN DO
    INPUT "NY VARE (TAST 3), VARE UDG. (TAST 4)",STYR
  ENDIF
  SELECT STYR THEN DO
    PRINT "DET TILFÆLDE FINDES IKKE"
  CASE 1
    REM //KØB TIL LAGER//
    ------------
  CASE 2
    REM //SALG FRA LAGER//
    ------------
  CASE 3
    REM //REG. AF NY VARE//
    ------------
  CASE 4
    REM //VARE UDGAAR OG SLETTES//
    ------------
  ENDCASES
UNTIL FYRAFTEN
END
```

□

4. Boolske udtryk.

I COMAL II findes de Boolske operatorer AND og OR.

Eksempel.

```
IF ANTAL>=0 OR SIDST=10 THEN DO
IF NAVN$<>"SLUT" AND TOTAL3>100 THEN DO
```

□

Det bemærkes, at Boolsk algebra ikke findes fuldt implementeret i COMAL II. Der må således ikke sættes parenteser i Boolske udtryk.

Eksempel.

Sætningen

```
UNTIL HØJRE(NODE)=0 OR HØJRE(NODE)=SIDST AND NODE=1
```

er syntaktisk korrekt, og under udførelsen af programmet vil det Boolske udtryk blive udregnet på sædvanlig måde fra venstre mod højre.
Sætningen

```
UNTIL NODE=1 AND (HØJRE(NODE)=0 OR HØJRE(NODE)=SIDST)
```

kan ikke udføres af COMAL II fortolkeren og vil fremkalde en syntax-fejlmelding.

□

Det skal også bemærkes, at COMAL II har arvet de pseudo-Boolske variable fra X.BASIC fortolkeren: En numerisk variabel opfattes i rette sammenhæng som en Boolsk variabel, hvis sandhedsværdi er "sand", hvis den variable er tildelt en værdi forskellig fra nul, og "falsk", hvis den variable er tildelt værdien nul.

Eksempel.

Hvis de variable FYRAFTEN og PASOP har fået tildelt værdierne hhv. 3 og 0, vil det Boolske udtryk i sætningen:

```
WHILE FYRAFTEN OR PASOP THEN DO
```

være syntaktisk korrekt og have værdien sand.

□

5. Kommentarer.

I COMAL II har ordene: ELSE, ENDIF, REPEAT og ENDWHILE samme status som REM med hensyn til kommentarer. Man kan altså efter de pågældende ord anbringe en kommentar, dvs. en streng, som er uden indflydelse på programmets udførelse og kun optræder ved listning af programmet.

Eksempel.

```
INPUT "ANTAL AEG: ",ANTALAEG
IF ANTALAEG<6 THEN DO
  PRINT "DER ER IKKE AEG NOK TIL EN BAKKE."
  STOP
ENDIF //SLUT, HVIS DER IKKE ER AEG NOK//
LET ANTBAKKR=0
REPEAT //SAA PAKKES DER 6 AEG AD GANGEN//
  LET ANTALAEG=ANTALAEG-6; ANTBAKKR=ANTBAKKR+1
UNTIL ANTALAEG<6
PRINT "DER BLEV PAKKET: ";ANTBAKKR;"BAKKER,"
PRINT "OG DER ER";ANTALAEG;"TILBAGE I LØS VAEGT."
END
□
```

6. Kommandoer.

Ud over kommandoerne i X.BASIC findes i COMAL II følgende kommandoer:

LRUN virker som RUN, men alle PRINT-sætninger bliver udført på linjeskriveren.

AUTO bevirker, at COMAL selv udstyrer sætningerne med numre således: 0010, 0020, 0030, ...
AUTO n,m (n og m positive, hele tal og n mindre end 9999) bevirker, at COMAL-fortolkeren udstyrer de efterfølgende sætninger med numre begyndende med n. Derpå følger n+m, n+2m, n+3m, ...
AUTO-tilstanden afbrydes ved, at man trykker på ESC-tasten.

BATCH og BATCH "$LPT" sætter den terminal, fra hvilken de

afgives, i batch-tilstand. Hvis kommandoen BATCH bruges, fremkommer udskrifterne på den terminal, fra hvilken kommandoen afgives; hvis man bruger kommandoen BATCH "$LPT", fremkommer udskrifterne på linjeskriveren. Batch-tilstanden kan afbrydes ved, at man trykker ESC på batch-terminalen. Sædvanlig tilstand opstår automatisk, når systemet modtager et EOF-signal fra kortlæseren.
En kortstak (en "batch") kan indeholde flere jobs. Et job begynder med et SCRATCH-kort og afsluttes med et END-OF-JOB (EOJ) kort. Hvert job får tildelt en kørselstid på 60 sek. Dette kan hvor som helst i stakken ændres ved, at man på et kort markerer JOB=<helt tal>. Så snart et job-kort detekteres, sættes jobtiden lig med det antal sekunder, som er angivet ved <helt tal>. Den sidst definerede jobtid gælder, indtil et evt. nyt job-kort detekteres, eller kørslen standser. I sidste tilfælde retableres standard-jobtiden på 60 sek.
Operatøren kan afbryde et job, som er under afvikling, ved at trykke CRTL A. Fortolkeren fortsætter så med det næste job.
Efter hvert job udsendes et FF-signal (linjeskriveren skifter side).
ON ESC-sætninger bliver overset i BATCH-tilstanden. INPUT-sætninger giver ERROR 07. Kommandoen findes ikke i RDOS.

7. Procedurer i COMAL II.

Hvis et program er indledt med sætningen PROC <navn>, hvor <navn> er en streng, der har samme format som et variabelnavn (jvf. pkt. 1), og afsluttes med sætningen RETURN, kan dette program kaldes af et andet program ved sætningen EXEC <navn>. Når det opkaldte program er udført vender fortolkeren tilbage til sætningen umiddelbart efter den EXEC-sætning, fra hvilket proceduren blev kaldt.

Eksempel.

Sætningen

```
EXEC UDSKRIFT
```

bevirker, at programmet

```
PROC UDSKRIFT
  ------------
  ------------
  ------------
RETURN
```

bliver udført inden fortolkeren udfører sætningen efter

```
EXEC UDSKRIFT
```

□

8. Afsluttende bemærkninger.

I COMAL II skal assembler user-subroutines indeholde erklæringen .EXTN CALL. Dette hænger sammen med, at programellet til CALL-faciliteten kun indlæses, hvis CALL er erklæret som .EXTN før COMAL B.LB, dvs. hvis brugeren ønsker en user-subroutine i sit program.

EOF (END-OF-FILE) er for $CDR defineret som:
1. EOF-kort (markering af alle ruder i en søjle)
2. som 2 sek. timeout,

hvilket betyder, at man ikke behøver et særskildt EOF-kort som sidste kort i stakken.

SOS COMAL II og RDOS COMAL II SAVE-filer er ikke compatible.

Endelig bemærkes, at kommandoerne LRUN og BATCH samt sætningerne EXEC og PROC ikke er implementeret i RDOS.