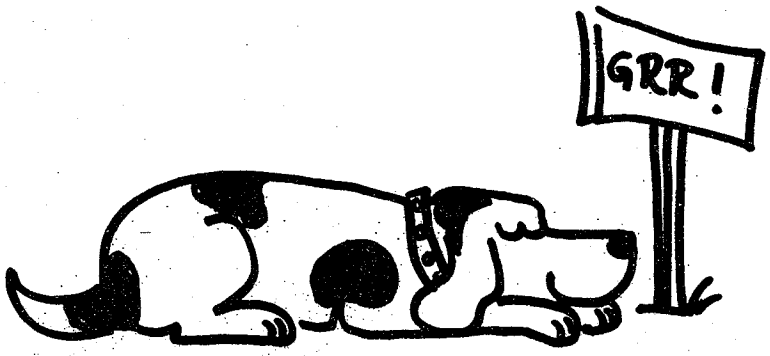
LATHUND

I ELVIRA - BASIC



INNEHÅLL:

4:e upplagan november 1978

1. ALLMÄNT OM SYSTEMET

ELVIRA-systemet består av en dator PDP-11/45 från Digital Equipment Corporation, operativsystemet heter RSTS/E och språket vi skall skriva i är BASIC-PLUS. Denna lathund sammanfattar de vanligaste kommandona och skrivsätten i BASIC-PLUS. Mera detaljerade upplysningar finns att få i handböcker (manualer) som finns tillgängliga i terminalrummen.

Till ELVIRA kan 32 terminaler samtidigt vara anslutna, dvs att 32 personer samtidigt kan använda henne. Som användare får du en area i arbetsminnet, där du kan skriva in och exekvera ett program. Du kan också hämta dit program- eller datafiler från ditt bibliotek på skivminnet, eller omvänt sända filer från arbetsarean till ditt bibliotek för senare användning. Arbetsarean är vid programmering i BASIC-PLUS 16K ord, dvs 16 x 1024 ord om 16 bitar. Det räcker för ganska stora program, om du inte rör dig med för stora matriser.

Varje användare loggar in under ett visst konto, och varje konto har sitt bibliotek. Filer som är lagrade under ett visst konto är tillgängliga för alla som kör under samma konto, men normalt ej för andra användare.

2. TERMINALERNA

Terminaler av flera olika typer är anslutna till Elvira. De är placerade enligt följande. (Reservation för tillfälliga omplaceringar!)

I sal D41, Lindstedtsvägen 15, 3tr, finns 15 textskärmar av typ Elite som kan kopplas direkt till Elvira. När schemalagd övning inte pågår är dock normalt bara tre kopplade till Elvira.
I Elvira:s labrum, rum 206, Osquars backe 14, 1 tr, finns tre skrivande terminaler av typ Decwriter och en textskärm, typ Elite.
I rum 213C, samma adress, finns två grafiska terminaler, typ Tectronix som vi dock lämnar utanför denna lathund. I rum 20 A,B, Lindstedtsvägen 3, 1 tr ned, finns sex skrivande terminaler av typ Teletype.

Teletype-terminalerna disponeras av E-teknologer och är tillgängliga dygnet runt i princip. Terminalerna i Elviras labrum är också tillgängliga för E-teknologer i den mån de inte reserverats av personer från E-institutioner eller med undervisning på E. (Övriga som vill använda dessa terminaler bör kontakta driftledningen för Elvira, tel 7814).

Terminalerna i D41 är i princip tillgängliga dygnet runt för alla teknologer. Undantagna är tider, då salen och terminalen reserverats för schemalagda övningar. Bokningsläget anslås för var vecka utanför sal D41 samt i korridoren nb, Lindstedtsvägen 15.

2.1 INSTÄLLNING AV TERMINALERNA.

Elite:

Slå på terminalen med knappen ON/OFF på höger sida.
Knapparna på framsidan skall vara enligt följande:
TAPE ute
FULL DUP intryckt
DATA RATE intryckt
EIA intryckt

Decwriter:

Slå på terminalen med knappen POWER ON/OFF t.v. på tangentbordet.

Teletype:

Starta genom att vrida knappen framtill t.h. till läget LINE.

Elite- och Decwriterterminalerna kan skriva både stora och små bokstäver. För att få enbart stora bokstäver in kan man efter inloggningen ge kommandot SET NO LC INPUT. På Elite-terminalerna kan man i stället trycka på knappen ALPHA LOCK, vilket medför att endast stora bokstäver matas in.

2.2 SPECIALTECKEN

Vissa tangenter på terminalerna har särskilda funktioner.

RETURN	(höger sida, kan också vara märkt CR) Efter varje meddelande till systemet - sats, kommando eller inmatning av data måste du trycka på RETURN. Först då tar ELVIRA emot meddelandet.
RUBOUT	(Märkt DELETE på vissa terminaler) Tar bort det sist skrivna tecknet. Två stycken tar bort de två sista tecknen etc. (BACKSPACE på Elite-terminalen tar INTE bort det sist skrivna tecknet, även om det ser ut så på skärmen).
CTRL	(vänster sida) hålles nedtryckt då en viss annan tangent trycks; dessa kombinationer är i det följande märkta CTRL-(bokstav). CTRL-X betecknas ofta ^X.
CTRL-C (^C)	Avbryter ett program under körning eller vid väntan på inmatning. Körningen kan återupptas från det ställe där avbrottet skedde med kommandot CONTINUE.
CTRL-I (^I)	Se TAB!

CTRL-O (^O) Användes för att slippa se långa utskrifter. En CTRL-O stänger av utskriften på terminalen. Utmatningsprogrammet fortsätter dock internt att gå igenom listan, och med ett nytt CTRL-O kan utskriften åter sättas i gång längre fram. (Om du är tillräckligt snabb).

CTRL-Q (^Q) Se CTRL-S!

CTRL-R (^R) Skriver ut den senast skrivna men ännu inte inmatade raden.

CTRL-S (^S) Avbryter en utskrift temporärt. CTRL-Q sätter igång den igen. Nödvändiga kommandon då man sitter vid textskärm och vill hinna läsa utskriften!

CTRL-U (^U) Tar bort den rad du håller på att skriva in. (om du inte hunnit trycka på RETURN)

TAB eller
CTRL-I (^I) Flyttar skrivhuvudet till nästa tabulatorstopp. Tabulatorstopp finnes på positionerna 8,16,24,32.... och kan inte flyttas.

CTRL-Z (^Z) Avbryter körning av systemprogram.

MELLANSLAG Mellanslagstangenten sitter längst ned på tangentbordet. Mellanslag och TAB saknar betydelse - utom i två fall: 1) i EXTEND MODE (se sid) måste de speciella BASIC-orden skiljas från angränsande tecken med mellanslag, och 2) i textsträngar (se sid). Använd mellanslag för att göra programmen mera lättlästa!

3. INLOGGNING OCH UTLOGGNING

Varje gång du kommer till din terminal är den "utloggad" för att den inte skall stjäla tid för andra användare. För att kunna köra måste du "logga in". När du är färdig med ditt pass skall du ALLTID logga ut.

3.1 INLOGGNING

Skriv:

HELLO nr1,nr2

åtföljt av tangenten märkt "RETURN" eller "<CR>".
nr1,nr2 är två nummer som identifierar kontot du kör på.

Elvira svarar då med att skriva

Password:

varvid du skall skriva in det lösenord som hör till ovannämnda konto, avsluta med tangenten "RETURN". Detta lösenord är hemligt

och du skall inte tala om det för någon annan. När du skriver in lösenordet ska det inte synas på pappret, om det gör det så sluta genast skriva in det, gör CTRL-C (se 2.2) och försök från HELLO igen. Om inloggningen gjorts riktigt svarar Elvira:

READY

som visar att hon är beredd att ta emot inmatning eller kommandon.

3.2 UTLOGGNING

Skriv:

BYE

Maskinen svarar då:

Confirm:

Om du nu svarar I, så skriver Elvira ut namnen på de filer du lagrat på ditt konto i tur och ordning med ett frågetecken efter. Skall filen vara kvar så trycker du på RETURN, behövs den inte längre så svarar du K (för "KILL"). OBS! DET ÄR VIKTIGT ATT DU INTE SPARAR FILER I ONÖDAN, Elviras minnesutrymme är begränsat! Om det inte är aktuellt att städa bort filer kan du i stället svara F (snabbast) eller Y (varvid Elvira skriver ut litet statistik om din körning). Om du svarar H, får du en lista över de möjliga svaren.

Du kan också direkt skriva:

BYE/I eller BYE/F (t.ex.)

4. INSKRIVNING OCH ÄNDRING AV PROGRAM

När ELVIRA skrivit READY är hon beredd att ta emot antingen inmatning av programrader eller kommandon. Om raden börjar med ett nummer uppfattas den som en programrad, annars som ett kommando. Båda slagen av inmatning verkställs först då du trycker på RETURN. Innan dess befinner sig det du skrivit i terminalens buffert, och du har möjlighet att korrigera eventuella fel med RUBOUT eller CTRL-U. (Se avsnitt 2, TERMINALERNA).

Ändra en rad	Skriv radnumret och radens nya innehåll.
Tillfoga en rad	Skriv raden med lämpligt valt radnummer. Den sorteras automatiskt in i programmet på sin plats enligt numret.
Radera en rad	Skriv bara radnumret. Då RETURN trycks raderas raden.

5. KOMMANDON

Kommandon är order till datorn om vad den omedelbart skall göra. Ett kommando ska stå ensamt på raden och inte ha radnummer.

5.1 Programhantering

CONT		Kort för CONTINUE. Återstartar ett program som avbrutits med CTRL-C från det ställe där det bröts.
DELETE		BÖR ANVÄNDAS VARSAMT! En rad raderas enklast genom att man skriver dess radnummer samt RETURN.
	DELETE m	Tar bort rad numrerad m. Flera radnummer kan specificeras med kommatecken emellan.
	DELETE	Delete utan argument tar bort ALLA rader i programmet!!!!
	DELETE m-n	Tar bort rad m till och med rad n.
	DELETE m-n,m1-n1	Tar bort rad m t.o.m. n OCH m1 t.o.m. n1.
LENGTH		Programmets längd i K ord skrivs ut.
LIST		Datorn skriver ut programmet du har i arbetsarean.
	LIST m	Datorn skriver ut rad m.
	LIST m-n	Datorn skriver ut rad m till och med rad n.
	LIST m-n,m1-n1	Datorn skriver ut rad m t.o.m. n OCH rad m1 t.o.m. n1
LISTNH		(List No Header) Som LIST fast överskrift med filnamn och klockslag skrivs inte ut.
NEW		Raderar arbetsarean och ger den namnet "NONAME".
	NEW filnamn	Raderar arbetsarean och ger den namnet "filnamn".
		RENAME
	RENAME filnamn	Döper om ditt program till "filnamn".
RUN		Programmet som finns i arbetsarean exekveras.
	RUN filnamn	Arbetsarean raderas och programmet med namn "filnamn" läses in och exekveras.
	RUN $PROG	Hämtar och kör programmet "PROG" från Elviras systembibliotek.
RUNNH		(Run No Header) Som RUN fast överskrift med programnamn och klockslag skrivs inte ut.

5.2 REMSHANTERING

KEY, PUNCH, TAPE: se avsnitt 17 ANVÄNDNING AV REMSA.

5.3 BIBLIOTEK

APPEND

APPEND filnamn	Sammanlänkar programmet i arbetsarean med programmet med namnet "filnamn". Om samma radnummer finns i båda programmen så försvinner det från programmet som ursprungligen fanns i arbetsarean och det från programmet som man länker ihop det med blir kvar istället.

CATALOG

	Ger en lista över BASIC-program i användarens bibliotek. Raderar inte arbetsarean.
CAT	Förkortning för CATALOG

DIRECT

DIR	Ger en lista över alla program i kontots bibliotek. Raderar arbetsarean.
DIR filnamn	Ser efter om filen med namnet "filnamn" finns på skivminnet och ger data för den. Kommandot raderar arbetsarean.

OLD

OLD filnamn	Hämtar programmet med namnet "filnamn" från användarens bibliotek och raderar dessförinnan arbetsarean.
OLD #PROG	Hämtar programmet PROG från det för en kurs gemensamma biblioteket.

REPLACE

REPLACE filnamn	Det program du har i arbetsarean ersätter program i ditt bibliotek med namn "filnamn". Om det program man vill ersätta inte finns i biblioteket så skapas ett nytt program med det namnet.

SAVE

SAVE filnamn	Sparar programmet som finns i arbetsarean i användarens bibliotek och ger det namnet "filnamn".

UNSAVE

UNSAVE filnamn	Raderar programmet "filnamn" från användarens bibliotek.

6. TAL

Två typer av tal förekommer, flyttal och heltal.

6.1 Flyttal i området (10E-38 => 10E38) samt 0 kan lagras. Talen lagras med 56 binära siffror, vilket motsvarar 15-16 siffror i taldelen.
Decimalkomma skrivs som decimalpunkt.
10-exponent anges med E. Exempel: 0.000001 kan skrivas som 1E-6. Observera att E-et måste föregås av en siffra (annars skulle man få värdet av variabeln E minskat med 6).

6.2. Heltal representeras med 16 binära siffror och ligger mellan -32768 och +32767. Heltal har alltid tecknet procent (%) efter sig och får inte innehålla decimalpunkt eller exponent. Ex. heltalet 2 skrivs som 2%.

7. VARIABLER

Innan en variabel tilldelas något värde i programmet är dess värde 0. Alla variabler nollställs och alla strängar sätts till tomma strängen (längd noll) när man ger kommandot RUN.

Variabler skall ha namn, och reglerna för namngivning beror av i vilken mod (tillstånd) Elvira befinner sig i.

7.1. EXTEND MODE (som är standard vid Elvira)

Flyttal	Namn på flyttalsvariabler får bestå av en bokstav följd av upp till 27 tecken - bokstäver, siffror eller punkt. Ex:A, A1, ALFA6.PX
Heltal	Samma regel som för flyttal, men namnet skall följas av tecknet %. EX: A%, A1%, ALFA6.PX%
Strängvariabler	Samma regel som för flyttal, men namnet följs av tecknet $. Ex: A1$, ALFA6.PX$ Om användningen av strängar och strängvariabler, se avsnitt 15.
Vektorer och matriser	Namn på vektorer och matriser följer samma regler som namn på enkla variabler. Matriser får ha en eller två dimensioner. Även strängmatriser får förekomma.

7.2. NO EXTEND MODE

I detta tillstånd tillåts namn endast bestå av en bokstav eller av en bokstav + en siffra. Övergång till detta tillstånd erhålles genom att satsen NO EXTEND infogas i programmet, återgång till det normala genom satsen EXTEND. Ändringarna gäller endast inom det aktuella programmet.

I EXTEND MODE måste BASIC-ord, ss LET, IF, NEXT m.fl. skiljas från omgivningen med mellanslag eller tabulatortecken.

8. OPERATORER OCH UTTRYCK

Operatorer i prioritetsordning:

1.	^ eller **	Upphöjning
2.	-	Tecken framför tal
3.	* /	Multiplikation, division
4.	+ -	Plus, minus

Med ovanstående operatorer bildas ARITMETISKA UTTRYCK av tal variabler och funktioner (se avsnitt 9).

5.	= <> < > <= >=	Jämförelseoperatorer

Två aritmetiska uttryck med en jämförelseoperator emellan utgör ett VILLKORSUTTRYCK.

6.	NOT	Logisk negation
7.	AND	Logiskt OCH
8.	OR	Logiskt ELLER
9.	XOR	Logiskt exklusivt ELLER

Med dessa operatorer kan man bilda nya villkorsuttryck av befintliga villkorsuttryck.

9. FUNKTIONER

9.1 STANDARDFUNKTIONER

Samtliga standardfunktioner utom RND, RANDOMIZE och PI skall ha ett argument - variabel eller uttryck - omgivet av parenteser.

Exempel: LET Y = ABS(X)

ABS(X)	Ger absolutbeloppet av argumentet
EXP(X)	Ger e upphöjt till argumentet
INT(X)	Ger det största heltalet <= argumentet. Exempel: INT(-5.5) ger -6.
LOG(X)	Ger argumentets naturliga logaritm. Argumentet skall vara positivt.
LOG10(X)	Ger 10-logaritmen av argumentet.
RND	Ger ett slumptal mellan 0 och 1. Argument behövs inte. Kör man programmet flera gånger fås samma slumptalssekvens igen (bra vid avlusning av program)
RANDOMIZE	Ger ny sådd till slumptalsgeneratorn. Dvs. nya slumptal ges varje gång man kör programmet.
SGN(X)	Signumfunktionen, = 1 om argumentet > 0 = 0 om argumentet = 0 = -1 om argumentet < 0
SQR(X)	Ger kvadratroten av argumentet. Argumentet får inte vara negativt.
SIN(X)	Ger sinus för argumentet. Argumentet i radianer.
COS(X)	Ger cosinus för argumentet. Argumentet

i radianer.

TAN(X) Ger tangens för argumentet. Argumentet i radianer.

ATN(X) Ger arcustangens för argumentet. Resultatet är uttryckt i radianer, och ligger alltid mellan -Pi/2 och Pi/2

TAB(X) Flyttar skrivhuvudet till den position på pappret, som anges av argumentet. Se vidare avsnitt 13, Utskrift.

PI Är en konstant som har värdet av Pi. ex. X=PI ger X värdet av Pi.

9.2 ANVÄNDARDEFINIERADE FUNKTIONER

9.2.1 Funktionsdefinition

DEF FNxxx(<Lista med argument>)=uttryck,

där xxx är vilket variabelnamn som helst (ex. FN.INRE.PRODUKT,FNE$,FNT8%)

TAG
Bort denna punkt

<lista med argument> är från 0 till 5 variabelnamn åtskilda av komma. Motsvarande variabler är funktionens argument (oberoende variabler), och de namn som används i definitionen kallas funktionens FORMELLA PARAMETRAR. Ett argument kan vara vilken variabel som helst dock inte en matris.

Funktionens värde är ett flyttal, ett heltal eller en sträng beroende på namnet.

MAX SJÄLV PÅ ELVIRA

DEF FNxxx(<Lista med argument>)

Definitionen av en funktion kan också sträcka sig över flera rader och avslutas då med FNEND ensamt på en rad. Före dess måste själva funktionsvariabeln FNxxx ha tilldelats ett värde.

Exempel:

```
15000 DEF FNMAX(X,Y,Z) !BERÄKNAR MAX AV X, Y, Z
15010 LET FNMAX = X
15020 IF Y > FNMAX THEN LET FNMAX = Y
15030 IF Z > FNMAX THEN LET FNMAX = Z
15040 FNEND
```

FFLAKTIGT SKRIVET

PGA att FN ANROPAR SIG SJÄLV

```
DEF FNMAX(A,B)
SUB.MAX=A
IF SUB.MAX<B THEN SUB.MAX=B
FNMAX=SUB.MAX
FNEND
```

9.2.2 Funktionsanrop

FNxxx(argument) kan ingå i uttryck precis på samma sätt som standardfunktionerna. Datorn beräknar värdet av uttrycket i funktionsdefinitionen med "argument" insatta i stället för de formella parametrarna. De talvärden, variabler eller uttryck som sålunda insättes kallas AKTUELLA PARAMETRAR, och varje användning av funktionen på detta sätt kallas ett FUNKTIONSANROP.

Exempel:

```
10 DEF FNF(X)=SIN(X)+X^2
.
.
50 LET Y = 2+ FNF(A+B)
```

Här är alltså uttrycket A+B den aktuella parametern, vars värde vid anropet sätts in i stället för den formella parametern X.

```
.
.
25 DEF FNLENGD(X1,Y1,Z1)=SQR(X1^2+Y1^2+Z1^2)
.
.
.
100 T=FNLENGD(100,250,500)
.
```

Denna funktion beräknar avståndet från origo till en punkt i ett tredimensionellt koordinatsystem. På rad 100 anropas funktionen och T sätts till värdet av avståndet mellan origo och punkten (100,250,500). Här är alltså de aktuella parametrarna talen 100, 250, 500.

OBS! Värdet av en aktuell parameter ändras ICKE genom ett funktionsanrop även om funktionsdefinitionen innehåller en sats som ger motsvarande formella parameter ett värde. Vid anropet överföres de aktuella parametrarnas värden till "lokala" variabler som deltar i beräkningarna. (S.k. "värdeanrop")

10. SATSER

Ett BASIC-program består av en följd av satser. Man skriver en sats per rad och inleder varje rad med ett radnummer.

(I BASIC-PLUS kan man också skriva flera satser per rad, varvid de åtskils av kolon (:). I denna lathund går vi dock inte in på dessa möjligheter, utan förutsätter att en sats skrivs per rad.)
Om en sats är för lång för att rymmas på en rad på terminalen, kan den fortsättas på nästa rad om man trycker LINE FEED i stället för RETURN. LINE FEED får inte komma mitt i ett BASIC-ord, ett variabelnamn, ett tal eller dylikt. Radnumret skrivs endast på den första raden. Maximal längd för en sats är 255 tecken.

De olika slagen av satser karakteriseras av särskilda ord som vi här skall kalla BASIC-VERB, fast inte alla är verb i grammatiken.

10.1. ELEMENTÄRA SATSER

BASIC-VERB EXEMPEL	ALLMÄN FORM. KOMMENTAR
DATA	Anger data, som skall läsas med READ.
DATA 33,4.7,"ADJÖ"	Data kan vara av typ flyttal, heltal eller strängar.
DIM	Reserverar utrymme för vektorer och matriser. Flera dimensioneringar kan stå på samma rad.
DIM A(3,13)	Observera att dimensioneringen E$(100) reserverar plats för hundra strängar - inte för en med längden 100.
DIM X(3),Y(8,19),E$(100)	
END	Skall vara programmets sista sats. Elvira lägger automatiskt ut ett END vid kommandot RUN.
FOR...NEXT	Repetitionsslinga. FOR variabel = A TO B STEP C ..NEXT variabel
FOR I = 1 TO N STEP 2	där A, B, C kan vara tal, variabler eller aritmetiska uttryck.
: :	
NEXT I	Satserna mellan FOR och NEXT utförs upprepade gånger till dess "variabel" passerat värdet av B - på väg uppåt eller neråt. För varje varv ökas "variabel" med värdet av C. (OBS att de värden av B och C som sålunda styr slingan är de som gällde då slingan startades!) Om B redan vid starten är passerat utförs slingan ingen gång. Om C = 1 får STEP-delen utelämnas.

GOTO

Hopp sker till den rad som angivits efter GOTO.

```
GOTO 400
```

GOSUB

Hopp till subrutin som börjar på angiven rad.

```
GOSUB 1000
```

IF...THEN

IF villkor THEN sats.
Om villkoret är uppfyllt utförs satsen efter THEN. Annars exekveras nästa rad i programmet. Satsen till höger om THEN kan vara av godtycklig typ, även en ny IF-sats.

```
IF A = 1 THEN GOTO 65
```

Denna speciella IF-sats kan förkortas till:

```
IF A = 1 THEN 65
```

eller till:

```
IF A = 1 GOTO 65
```

INPUT

Då en INPUT-sats exekveras, skrivs ett frågetecken ut på terminalen. Användaren ska då mata in värden på de variabler, som står efter INPUT. Värdena skall åtskiljas av komma (,) RETURN, LINE FEED eller ESCAPE.

```
INPUT A,B,D$
```

Även strängar kan inmatas (exempel ja-nej frågor.)

```
INPUT "VAD ÄR X";X
```

En ledtext kan också tas med i INPUT-satsen och skrivs då ut före frågetecknet.

LET

LET variabel = uttryck.

```
LET A = 3
```

Variabeln tilldelas värdet av uttrycket. LET får utelämnas.

```
LET A,B = 0
```

ger både A och B värdet noll.

NEXT

Se FOR...NEXT

PRINT

Se avsnitt 13. Utskrift

READ

READ var1,var2,var3....var Läser från DATA-sats och tilldelar den första variabeln det första värdet från den första DATA-satsen i programmet osv.

```
READ A,B,C$
```

REM

Kommentar; godtyckliga tecken får förekomma.

```
REM BERÄKNING AV Y
```

Utropstecken

Kommentar kan även skrivas med hjälp av utropstecken.

```
50 LET A = 1 !A FÅR NYTT VÄRDE
```

Allt som står efter utropstecknet uppfattas av BASIC-systemet som en kommentar. Den skrivs ut då programmet skrivs ut, men påverkar inte programmet. Gör ditt program lättbegripligt med väl valda kommentarer!

RESTORE

Datorn har en pekare, som pekar på det första datum i DATA-satserna, som inte lästs. RESTORE flyttar denna pekare till första DATA-satsens början så man kan läsa samma data igen.

RETURN

Återhopp från subrutin till den sats som följer det GOSUB som subrutinen anropades från.

STOP

Exekveringen avbryts. Flera STOP-satser får förekomma i samma program.

10.2. "AVANCERADE SATSER".

FOR...UNTIL...NEXT

FOR variabel = A STEP B UNTIL villkor

```
FOR I = 1 STEP 2 UNTIL I > 8
 :
 :
NEXT
```

Repetitionsslingan upprepas så länge som "villkor" inte är uppfyllt.

FOR...WHILE...NEXT

FOR variabel = A STEP B WHILE villkor

```
FOR J = 3 WHILE J < M
 :
 :
NEXT
```

Repetitionsslingan upprepas så länge som "villkor" är uppfyllt.

I båda dessa repetitionssatser används variablernas aktuella värden vid utvärdering av villkoret.
Som vid den vanliga FOR-satsen får STEP 1 utelämnas.

IF...THEN...ELSE...

IF villkor THEN jasats ELSE nejsats

```
50 IF A > B THEN MAX=A ELSE MAX=B
```

ON...GOTO

ON uttryck GOTO nr1,nr2,....nrk

```
ON N GOTO 10,110,10,30
```

Om "uttryck" = 1 sker hopp till rad nr1, om "uttryck" = 2 sker hopp till rad nr2, etc.

ON...GOSUB

ON uttryck GOSUB nr1,nr2,....nrk
Analogt med ON...GOTO vid hopp till subrutiner.

UNTIL...NEXT

```
190 N = 1
200 UNTIL SUM > 10
210 SUM = SUM+1/N
220 N = N+1
230 NEXT
```

UNTIL villkor ..... NEXT

Satserna mellan UNTIL och NEXT upprepas så länge "villkor" inte är uppfyllt.

WHILE...NEXT

```
390 N = 1
400 WHILE SUM <= 10
410 SUM = SUM+1/N
420 N = N+1
430 NEXT
```

WHILE villkor ..... NEXT

Satserna mellan WHILE och NEXT upprepas så länge som "villkor" är uppfyllt.

11. TALVÄRDEN AV VILLKORSUTTRYCK

Logiska värden representeras i Elvira med tal.
Sant representeras av -1 och falskt med 0.

I aritmetiska uttryck får också logiska uttryck ingå. Ett logiskt uttryck ger då -1 om det är sant och 0 annars.

Exempel: PRINT X>7 ger utskrift 0 eller -1

LET A=(B=0) ger A värdet -1 om B=0 annars 0

12. FÄLT

Varje fältelement hanteras som en variabel, men det finns speciella satser för att hantera hela fält.

DIM

Reserverar utrymme för vektorer och matriser. Vektorer som ej nämnts i DIM-sats har dimensionen 10. Matriser som ej nämnts i DIM-sats har dimensionen 10x10. Undre indexgräns är alltid 0. Dock används inte index noll vid matrisoperationer. Exempel: DIM A(3,15)

För varje vektor och matris reserveras ett visst utrymme - genom DIM-sats eller implicit. Varje vektor och matris har även en viss AKTUELL DIMENSION som från början är den reserverade, men sedan kan vara mindre (ej större). m och n nedan får vara uttryck. Om uttryckens värde ej är heltal sker avhuggning.

MAT A = IDN	A blir en enhetsmatris. A måste vara kvadratisk.
MAT A = IDN(m,m)	A blir en enhetsmatris med aktuell dimension m x m.
MAT B = ZER	B blir en nollmatris resp. nollvektor.
MAT B = ZER(m,n)	B blir en nollmatris med aktuell dimension m x n.
MAT C = CON	C blir en matris med endast ettor.
MAT C = CON(m,n)	C blir en matris med endast ettor och aktuell dimension m x n.
MAT INPUT D	Inläsningen av matris D från terminal.
MAT INPUT D(m,n)	Inläsning av matrisen D med aktuell dimension m x n från terminal. (Inläsningen sker radvis D(1,1), D(1,2),...)
MAT PRINT E	Utskrift av matrisen E med 1 element per rad.
MAT PRINT E,	Utskrift av matrisen E med 5 element per rad.
MAT PRINT E;	Utskrift av matrisen E tätpackad. 1 radframmatning görs då raden tar slut. 2 radframmatningar görs för varje matrisrad.
MAT PRINT E(m,n)	Utskrift av delmatrisen m x n från matris E
MAT READ F	Inläsning av matrisen F från DATA-sats
MAT READ F(m,n)	Inläsning av matrisen F med aktuell dimension m x n från DATA-sats. (Inläsningen sker radvis D(1,1), D(1,2),...)
MAT G = H + K	Addition. Aktuella dimensioner måste överensstämma.
MAT J = K - L	Subtraktion. Aktuella dimensioner måste överensstämma.
MAT M= N * P	Multiplikation. Om N är m x p och P är p x n så måste M vara m x n.
MAT Q =(uttryck)*R	Multiplikation med skalär. Aktuella dimensioner måste stämma.
MAT S = T	Kopiering. Matris S måste vara minst lika stor som T, den får aktuella dimensionen som T.
MAT U = TRN(V)	Transponering. Om U är m x n så måste V vara n x m.
MAT X = INV(Y)	Invertering. Aktuella dimensioner måste stämma. X och Y måste vara kvadratiska. Efter inverteringen finns determinanten i variabeln DET.

OBS! Endast vid addition, subtraktion, multiplikation med skalär och invertering får man ha samma fältvariabel i högra och vänstra ledet.

13. UTSKRIFT

Man kan skriva ut dels värden av uttryck dels textsträngar.

```
PRINT A                 ger utskrift av A:s värde
PRINT "A"               ger utskrift av texten A
```

Med en PRINT-sats kan man skriva ut flera värden och/eller textsträngar.

13.1. VAR PÅ PAPPRET?

En rad består av 72-80 positioner (beroende på terminal) numrerade från 0 och uppåt. Den är indelad i kolumner, som har sina vänsterkanter i positionerna 0,15,30,45,60 resp 75.

, Skriver man komma efter en variabel eller sträng i en PRINT-sats flyttas skrivhuvudet till nästa kolumn, så att nästa värde skrivs i ny kolumn.

; Skriver man ; efter en variabel eller sträng i en PRINT-sats flyttas skrivhuvudet till nästa position, så att nästa värde kommer så tätt som möjligt. Dvs. med ett mellanrum mellan.

Ny rad Om sista uttrycket eller strängen i en PRINT-sats inte följs av komma eller semikolon, kommer nästa utskrift på ny rad.

När raden tar slut kommer automatiskt ny rad.
Ny rad får man även så fort man försöker utnyttja 15 positioner i sista kolumnen.

Tom rad PRINT ensamt ger radframmatning.

TAB En standardfunktion (se avsnitt 9.1) som används i PRINT-satser.
Om argumentet är < skrivhuvudets nuvarande position, står skrivhuvudet kvar.
Om argumentet är < 0, står skrivhuvudet kvar.
Om argumentet ej är heltal tas heltalsdelen av argumentet. Genom att öka argumentet med 0.5 får man avrundning.

Exempel: PRINT TAB(X);"*"

13.2 UTSKRIFTSFORMAT

Om man använder vanlig PRINT-sats så upptar talet 1-8 positioner (maximalt sex decimaler). Vill man ha flera eller färre decimaler i utskriften kan man använda en funktion med namn PRINT USING

```
PRINT USING "xxx.yyy" A
```

Denna sats där xxx och yyy vardera består av 0-20 nummertecken (Item, "#"), talar om att A skall skrivas ut med xxx siffror före decimalpunkt och yyy decimaler. Siffror fylls ut till vänster med blanktecken, och decimaler till höger med nollor.

Exempel:

```
PRINT USING "#.##########",1/3
```

ger utskriften:

```
0.3333333333
```

Ett exempel till:

```
A$ = #.###
PRINT USING A$,1/3
```

ger utskriften:

```
0.333
```

Fyra uppåtpilar (^) ger utskrift med 10-exponent.

Exempel:

```
PRINT USING "#.##^^^^",1/3
```

ger utskriften:

```
3.33E-01
```

Får utskriften inte rum i det angivna fältet skrivs den på vanligt sätt med ett %-tecken före.

Text får inkluderas i formatnotisen, ex. "###.## KR"

13.3 UTSKRIFT FRÅN SKÄRMTERMINALER.

I Elviras labrum och i terminalsal D41 finns radskrivare dit utskrifter kan dirigeras från skärmterminalerna. Detta görs med kommandot:

SKRIV filnamn,antal

För BASIC-program räcker programnamnet utan extension. "antal" anger antalet kopior; utelämnas det får man 1 ex.

14. FELMEDDELANDEN FRÅN SYSTEMET

Systemet kan ge en mängd olika felmeddelanden för olika fel som kan uppstå (närmare bestämt 127 stycken). Här nedan är en förteckning på de vanligaste:

14.1 Fel vid programskrivning

Illegal expression	Operander saknas eller är dubblerade, höger- och vänsterparenteser matchar inte varandra, eller något liknande.
Illegal line numbers	Radnummer måste vara heltal mellan 1 och 32767.
Illegal mode mixing	Strängoperander och tal är blandade med varandra.
Illegal symbol	Ett felaktigt tecken finns på raden
Illegal verb	Ett ord som inte hör till BASIC är hittat.
Syntax error	Syntaxfel av annan art än ovanstående.

14.2 Felutskrifter vid inmatning

?	Inmatning väntas.
Data format error	Felaktig inmatning - försök igen.

14.3 Felutskrifter vid exekvering

Många olika felutskrifter kan förekomma, ett urval är:

Floating point error	Försök har gjorts att räkna med tal vars absolutbelopp är större än 10E38 eller mindre än 10E-38. 0 är returnerat som svar.
Arguments too large in exp	Argumentet måste ligga mellan ca -89 och +88. 0 är returnerat som svar.
Illegal statement	Ett fel i programmet är inte rättat.

14.4 Felutskrifter övriga

Här kommer några förklaringar på övriga vanliga felutskrifter.

No room for user on device	Skivenheten full, försök ta bort några filer (särskilt stora).
Can't find file or account	Filen eller biblioteket (kontot) du söker finns inte.
End of file on device	END saknas när program kallas in för exekvering, Skriv RUN igen, utan filnamn.
File exists-rename/replace	Program finns redan med det namn som du använder. Ändra namnet på ditt program eller använd REPLACE kommandot istället för SAVE (vid SAVE).

15. STRÄNGAR

Sträng

En sträng är en följd av 0 till 30055 tecken omgiven av citationstecken eller apostrofer.
Exempel: "NEJ" 'JA'

Strängvariabel

En strängvariabel är en variabel vars värde är en sträng. Namn på strängvariabler lyder samma regler som namn på andra variabler (se avsnitt 7), men avslutas alltid med tecknet $.

IF...THEN

IF sträng relationsoperator sträng THEN sats.
där relationsoperatorerna är =, <, >, <=, >=, <>
NOT,AND och OR får ej förekomma.
Strängarna jämförs tecken för tecken från vänster till höger. Siffror anses mindre än bokstäver och en bokstav, som kommer tidigare i alfabetet anses mindre än en som kommer senare. (Obs dock att Ä < Ö < Å)
"JOHAN" anses mindre än "JOHANSSON" etc.
Exempel: IF A$="JA" THEN 70

```
IF B$<C$ THEN B$=B$+C$
```

INPUT

Inläsning från terminal. Om flera strängar matas in med samma INPUT-sats åtskiljs de med kommatecken.
Exempel: INPUT A$

```
INPUT A,A$,B$,B7$
```

INPUT LINE

Läser in en hel rad från terminalen som en sträng, inklusive det avslutande CR, även om raden innehåller kommatecken.

Exempel: INPUT LINE A$

LEFT

LEFT är en standardfunktion. LEFT har två argument den första en sträng och det andra ett heltal. Som svar ges de n vänstraste tecknen i strängen.
Exempel: LEFT("STRÄNG",3%) Ger som svar "STR"

A$=LEFT(A$,A) Tar bort alla tecken i A$ som är efter det av A angivna.

RIGHT

RIGHT är också en standardfunktion som har två argument. Det första är en sträng och det andra ett heltal. RIGHT ger som svar strängen från och med det n:te tecknet till slutet (där n är heltalet).

Exempel

A$=RIGHT(A$,2%) tar bort första bokstaven från A$.

MID

MID Är en standardfunktion med tre argument. Det första är en sträng och de två andra heltal. MID ger som svar en delsträng, som börjar med det tecken som utpekas av det andra argumentet och vars längd=det tredje argumentet.
Exempel: MID("DETTA ÄR EN STRÄNG",7%,2%) Ger som svar "ÄR" (Ä: et är det sjunde tecknet).

LEN

LEN är en standardfunktion. LEN skall ha ett argument, som skall vara en sträng. LEN ger som svar den aktuella längden av strängen.
Exempel: PRINT LEN(A$)
Skriver ut längden av strängen A$

LET

Tilldelning. (LET kan liksom vid aritmetisk tilldelning utelämnas.)
Exempel: LET A$="ABCDEFG"

LET A$=A$+MID(D$,A,Y)+R5$

READ

Inläsning från DATA sats.
Exempel: READ Y$,T9$

PRINT

Skriver ut värdet av en sträng. Sträng och numeriska värden får blandas i en PRINT-sats.
Exempel: PRINT A$,X

PRINT "VA KUL"; Y; MID(B$,1%,4%)

CVT$$

Strängkonvertering.
T$ = CVT$$(S$,M%) gör att T$ blir lika med en modifierad S$. Modifikationen styrs av de binära potenser som ingår i heltalet M% enligt följande:

1% Ta bort paritetsbitarna
2% Ta bort alla mellanslag och tabulatortecken
4% Ta bort styrtecknen CR,LF,FF,ESC,RUBOUT och NULL
8% Ta bort inledande mellanslag och tabulatortecken
16% Dra ihop grupper av mellanslag och tabulatortecken till ett mellanslag
32% Omvandla små bokstäver (gemena) till stora (versaler)
64% Omvandla ` till (och ~ till)
128% Ta bort avslutande mellanslag och tabbar
256% Men - ändra inte tecken som står inom citationstecken.

Typiskt BRA!! allt utom 256, 64, 2

Exempel: IF CVT$$(A$,189%) = "JA"THEN 7000

16. FILER

Fil

En fil är ett minnesutrymme, som kan läggas upp på datorns skivminne och där nås av program och kommandon. Data kan läsas från och skrivas på filer. Ett sparat program är också en fil.

Filnamn

En fil har ett namn för att identifiera filen. Under RSTS har ett filnamn formen:

devn:filnamn.ext(proj,prog)<protection>

De olika delarna i filspecifikationen beskrivs nedan. De delar som inte behövs behöver inte skrivas ut. Exempelvis behöver "(proj,prog)" inte skrivas ut om filen ligger på eget konto, "dev:" inte om filen

ligger på skivenhet. "Protection" behöver endast anges då man ändrar den.

devn: Representerar vilken perifer enhet man vill att filen skall hamna på. Enhet har formen två bokstäver eventuellt följt av en siffra avslutat av ett kolon. Siffran talar om vilken enhet man vill ha (om det finns flera av samma typ). Siffran kan utelämnas om man menar enhet noll. Enhetsspecifikation kan utelämnas helt om man vill öppna en fil på system-disk (DP0). De på detta system förekommande enheterna är:

DPn: (Disk Pack). Skivenhet.
DKn: (Disk cartridge). Skivenhet, finns två, nr. 0 och 1.
KBn: (Keyboard) Terminal, finns fn. 32 st. numrerade 5 till 36
KB: Egen terminal specificeras utan nummer.
LP: (Line printer) Radskrivare, finns en.
MT: Magnetbandsstation, 9-kanals, 800bpi
NL: Nollenheten (Svarta hålet!)
PP: (Paper Punch) Pappersstans, finns en.
PR: (Paper Reader) Remsläsare, finns en.
OBS! Kolon ingår i namnet!

"filnamn" är själva namnet på filen. Detta namn ignoreras för alla enheter utom (DK), (DP) och (MT).

Filnamnet får bestå av från 1 till 6 alfanumeriska tecken.

"ext"(extension) är en utvidgning av filnamnet vid diskfiler.

Om filen är ett program, anger extension programspråket:
.BAS = BASIC, .FOR = Fortran och .MAC = MACRO-11(Assemblyspråket).

Under RSTS/E är extension .BAS underförstådd; namnen på BASIC-program kan därför ges utan extension.

Extension får bestå av från ett till tre alfanumeriska tecken. När filen inte är ett program kan den också utelämnas, men om den är med måste punkten och extension ansluta omedelbart till filnamnet.

(proj,prog)(projekt-, programmerarnummer) är ett nummer som talar om på vilket konto filen skall finnas. Detta nummer används bara då man skall använda filer på andra konton än det man själv kör på. Ignoreras för andra enheter än disk. Kontonummer består av två nummer från 0 till 255 åtskiljda med ett kommatecken och med hakparenteser omkring (vanliga parenteser går också).

Vissa konton får specialbeteckningar. Sålunda betecknar

#PROG

den fil med namnet PROG som finns på kontot med samma projektnummer som mitt, men med programmerarnumret 0. Dessa konton används med fördel som gemensamma bibliotekskonton för alla konton inom ett projekt (t ex kurs).

Och

$PROG

betecknar filen med namnet PROG i det för alla användare gemensamma biblioteket.

Vissa areor på skivminnen har fått egna namn som om de vore självständiga enheter. Det gäller t ex HLP:, DOC: och PUB: som innehåller resp. instruktionsfiler, dokumentation och program av allmänt intresse.

Kanal

För att programmet skall veta var filen man vill använda finns någonstans så finns det kanaler. När man vill tala om att man vill läsa/skriva data till en fil så öppnar man en kanal.

Det finns 13 kanaler till förfogande, ett program kan alltså läsa/skriva på 13 filer samtidigt. Kanalerna är numrerade från 0 till 12 där dock kanal 0 är reserverad. Kanal 0 är alltid öppen på egna terminalen, detta innebär att man alltid kan läsa/skriva på egen terminal.

OPEN

används för att öppna en kanal. Open fungerar i en sats med formen:
satsnr OPEN "dev:filnamn.ext" AS FILE <kanalnummer>
Exempel: 170 OPEN "ALLAN.DAT" AS FILE 1

CLOSE

används för att stänga en kanal. Alla kanaler måste stängas före END i programmet för att de data man skrivit ut skall behållas. Close ingår i satser med formen:

radnr CLOSE <kanalnummer>

Exempel: CLOSE 1 !Denna sats stänger kanal 1

KILL

används för att ta bort en fil från disk.
Det kan användas både som ett kommando och en sats.
Exempel: KILL "ALLAN.DAT"

Skriva på en fil

för att skriva på en fil används en specialform av PRINT, med kanalnummer:
PRINT #<kanalnummer>,<lista med variabler>
Exempel: PRINT #2%,A,B,C$
Skriver värdet av variablerna A,B och C$ på den fil som är öppen på kanal 2. I detta fall representerade tvåan kanalnummer, vilket kan vara ett uttryck. Värdena skrivs direkt efter senast skrivna värden, eller först om det inte är skrivet något förut på filen. Vill man börja skriva från början av filen igen får man öppna kanalen igen med OPEN-satsen. Även fält och matriser kan skrivas på fil.

PRINT USING kan användas här också på samma sätt som vid utskrift.

Läsa från fil

Läsning från fil sker med satsen INPUT. Form:
INPUT#<kanalnummer>,<lista med variabler>
Exempel:
INPUT #2%,A,B,C7$
Läser värdet av variablerna A,B och C7$ från filen öppen på kanal 2. Filnumret kan vara ett uttryck, som automatiskt avrundas nedåt till närmaste heltal. Värdena läses direkt efter det sist lästa eller skrivna. Vill man börja att läsa filen från början igen får man öppna samma fil igen med OPEN-satsen. Även fält och matriser kan läsas från en fil.

OBS! För att läsa text från fil används lämpligen

INPUT LINE #<kanalnummer>,<strängvariabel>

t.ex. INPUT LINE #2%,A$

17. ANVÄNDNING AV REMSA

Vid stansning av programmet se först till att du har sparat det på disk, då det annars går förlorat då du försöker stansa det.

Om du vill ha kvar de program du gör så skall du alltid spara dem på remsa. Remsstansningen/läsningen försigår lite olika beroende på om du stansar på snabba remsstansen (i Elviras labrum) eller på TELETYPE. Observera att remsor stansade vid Teletype INTE får läsas in med snabba remsläsaren, av det skälet att teletyperemsor är oljade och snabba remsläsaren läser remsorna optiskt, så dess lins skulle bli igensmetad av oljan från teletyperemsorna. Däremot får remsor stansade uppe på snabba remsstansen läsas in på Teletype.

17.1 STANSNING VID TELETYPE

Tillse först att det finns remsa i stansen.

a. Skriv:
   PUNCH
   direkt åtföljt av namnet på det program du vill stansa. (Och tangenten "RETURN") Svara med RETURN på frågan som kommer.
b. Tryck på ON på stansen när uppmaningen:
   SÄTT PÅ STANSEN
   kommer.
c. Stäng av stansen genom att trycka på OFF när remsan är färdigstansad
d. Riv av remsan.

När programmet stansas så får du underlig utskrift på terminalen men det är precis som det ska vara. Remsan märks automatiskt med programmets namn och dagens datum.

17.2 LÄSNING AV REMSA PÅ TELETYPE

a. Skriv: NEW
   åtföljt av det namn du vill ha på programmet. (och tangenten "RETURN")
b. Lägg remsan i läsaren och kläm fast den i spännet.
c. Skriv TAPE på terminalen.
d. Tryck på RETURN
e. Ställ spaken på läsaren i läge START.
f. Se till att remsan inte trasslar till sig.
g. När läsningen är färdig, skriver du KEY åtföljt av tangenten märkt ESCAPE (ALTMODE på en del terminaler).
h. Nu har du programmet i din användar-area

17.3 STANSNING PÅ SNABBA REMSSTANSEN

Stansa remsa på snabba remsstansen kan du bara göra om du sitter vid en av terminalerna i Elviras labrum.

a. Skriv PUNCH åtföljt av namnet på programmet samt RETURN.
b. Gå och hämta programmet vid snabbstansen.

   OBS! Punchprogrammet raderar användararean.

17.4 LÄSNING AV REMSA FRÅN SNABBA REMSLÄSAREN

Läsning av remsa från snabba remsläsaren kan bara ske om man sitter vid terminalerna i Elviras labrum.
OBS! Ej oljad remsa från teletyperna.

a. Tillse först att remsläsaren är påslagen, det ses genom att den lyser. Strömbrytaren sitter baktill.
b. Sätt knappen märkt READER i läge ON.
c. Fäll upp bygeln som ska hålla kvar remsan, genom att vrida den svarta ratten åt höger.

d. Sätt in remsan i högra facket, så att den går under den svarta plåtbygeln och kuggarna på hjulet passar in i hålen (de små) på remsan.
e. Fäll ned plåtbygeln genom att vrida ratten moturs, tillse att kuggarna kommer rätt.
f. Gå till terminalen och skriv:
   OLD PR:
   åtföljt av namnet på programmet och tryck ned tangenten RETURN.
g. Skynda till remsläsaren och se till att remsan inte trasslar sig.
h. Programmet finns nu i din arbetsarea.

(707) Tel: 7814 787 8922