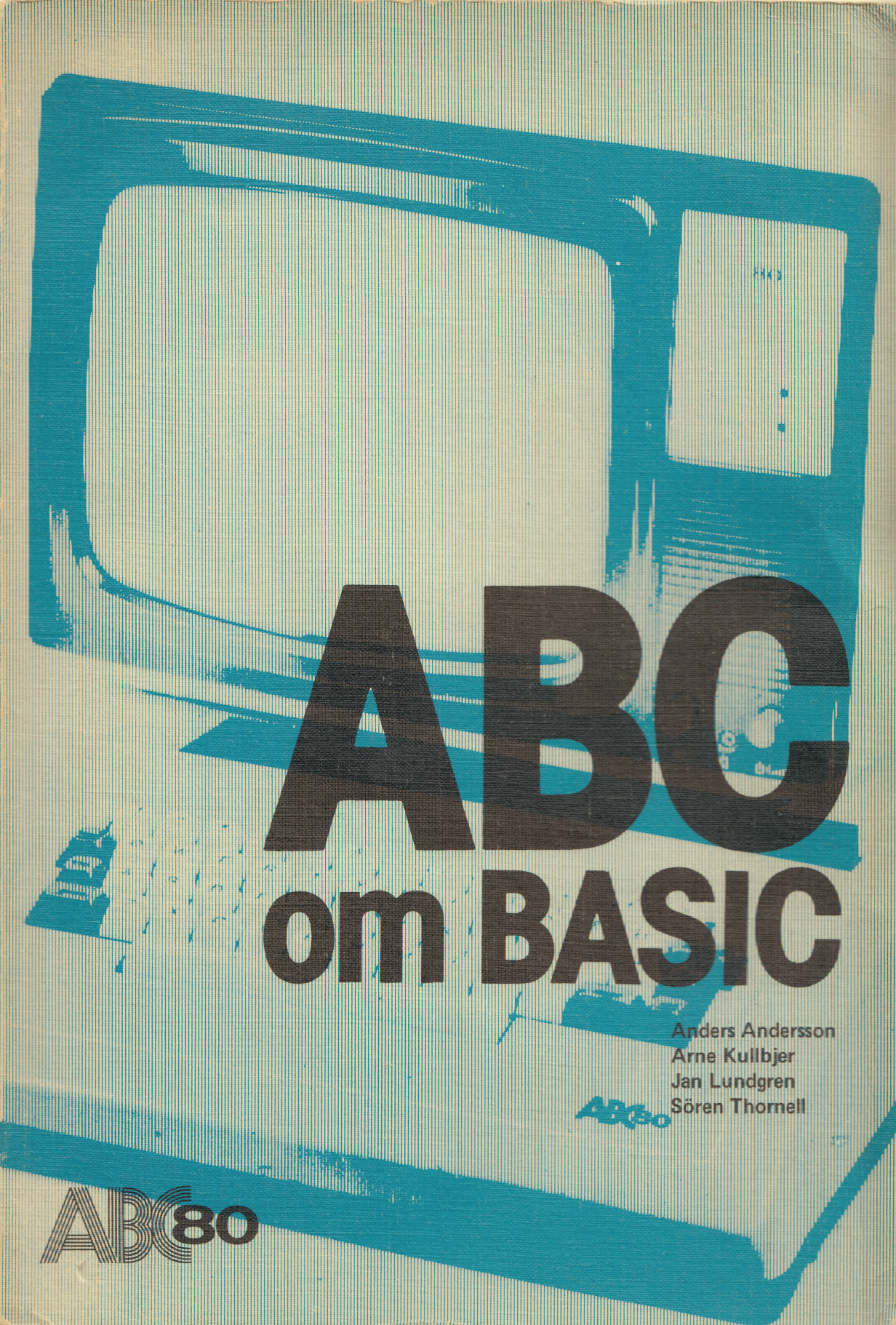




ABC om BASIC

Anders Andersson
Arne Kullbjer
Jan Lundgren
Sören Thornell

ABC80

ABC om BASIC

**Anders Andersson
Arne Kullbjer
Jan Lundgren
Sören Thornell**

ABC80

OBSERVERA den begränsning i rätten att kopiera ur denna skrift
som finns inskriven i Avtal om kopiering i skolorna och högskolorna
(SÖ-FS 1981:126).

ABC om BASIC

© 1982 Didact och LiberYrkesutbildning-Teknik Stockholm

ISBN 91-7260-254-6

Andra upplagan 1980 3 4 5 6 7 8 9 10

Figurer: Ulf Nilsson, Marianne Lundin

Fotografier: AB Fotografen Bildbyrå

Tryck LiberTryck, Stockholm 1983

INNEHÅLL

KAPITEL 1. Introduktion till ABC	80	1
KAPITEL 2. Datorn som intelligent skrivmaskin	7	
KAPITEL 3. Datorn söker i register	19	
KAPITEL 4. Datorn utför beräkningar	31	
KAPITEL 5. Datorn fattar beslut	43	
KAPITEL 6. Lagra program på kassett	61	
KAPITEL 7. Flödesschema underlättar programmeringen	67	
KAPITEL 8. Datorn hanterar bokstavskombinationer	83	
KAPITEL 9. Datorn löser matematiska problem	101	
KAPITEL 10. Grafisk presentation på TV-skärmen	115	
KAPITEL 11. Lagra data på kassett	129	
KAPITEL 12. ASCII-aritmetik	139	
KAPITEL 13. Att avhjälpa programfel	149	
KAPITEL 14. Matrisberäkningar	163	
APPENDIX 1. Kommandon och instruktioner	175	
APPENDIX 2. Felmeddelanden	187	
APPENDIX 3. Svar till övningsexempel	193	
APPENDIX 4. Sakregister	213	

FÖRORD

Den snabba utvecklingen inom elektroniken har medfört att kraftfulla datorer nu kan användas i många nya tillämpningar. Datorer, som idag kan köpas av privatpersoner för en kostnad som motsvarar priset på en färg-TV, har samma kapacitet som de datorer, som för ett tiotal år sedan kostade en miljon kronor. En helt ny användarkategori, utan tidigare erfarenhet av datorer, har därför ett stort behov av utbildning för att kunna möta den nya tekniken. För att utnyttja datorer på rätt sätt, samt framför allt ha glädje av den stora mängd färdiga program som finns tillgängliga, krävs vissa grundläggande kunskaper om hur datorer programmeras. Denna bok ger dessa kunskaper. Boken är avsedd att användas tillsammans med den svenska datorn ABC 80. Formerna för utbildning kan variera, men denna bok kan med fördel användas både för självstudier och för gruppundervisning i gymnasieskolan, inom arbetsmarknadsutbildningen, för internutbildning på företag och för kurser i studieförbundens regi.

För att kunna tillgodogöra sig denna bok krävs inga förkunskaper om datorer eller deras programmering.

Bokens sju första kapitel behandlar allmänna användningsområden, medan de sju sista kapitlen beskriver mera speciella tillämpningar, t ex ordbehandling och grafisk presentation.

I denna andra upplaga har ett par tryckfel rättats till, i övrigt är innehållet oförändrat.

Vi tackar alla som medverkat vid tillkomsten av denna bok, bland andra medarbetarna på DIAB, LUXOR och SCANDIA METRIC, samt Elsiey och Lisa, som överfört våra handskrivna manus till läsbar form på composeern.

Linköping i februari 1980

Författarna

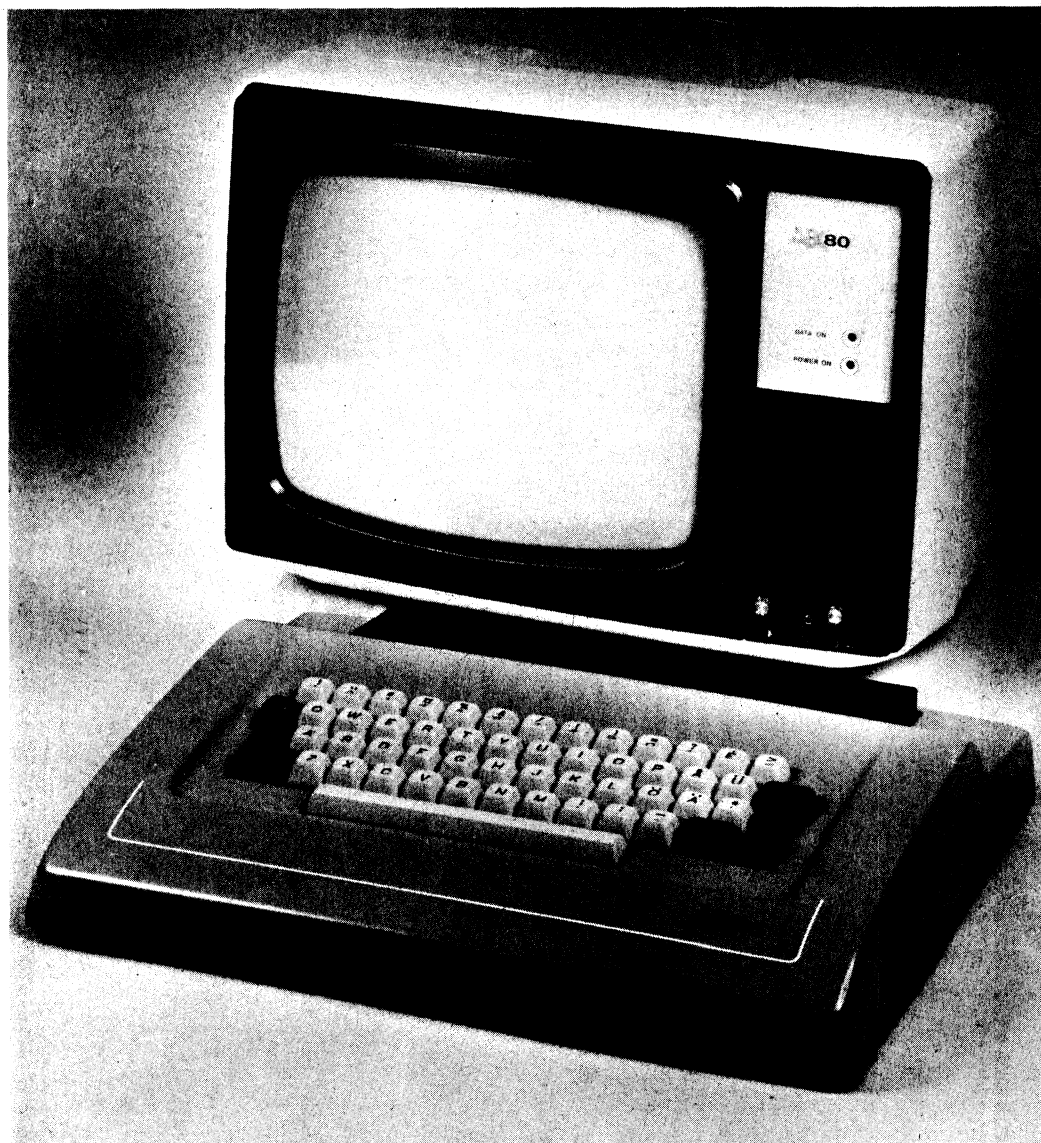
KAPITEL 1

Introduktion till ABC 80

ABC80

ABC 80, ETT FLEXIBELT DATORSYSTEM

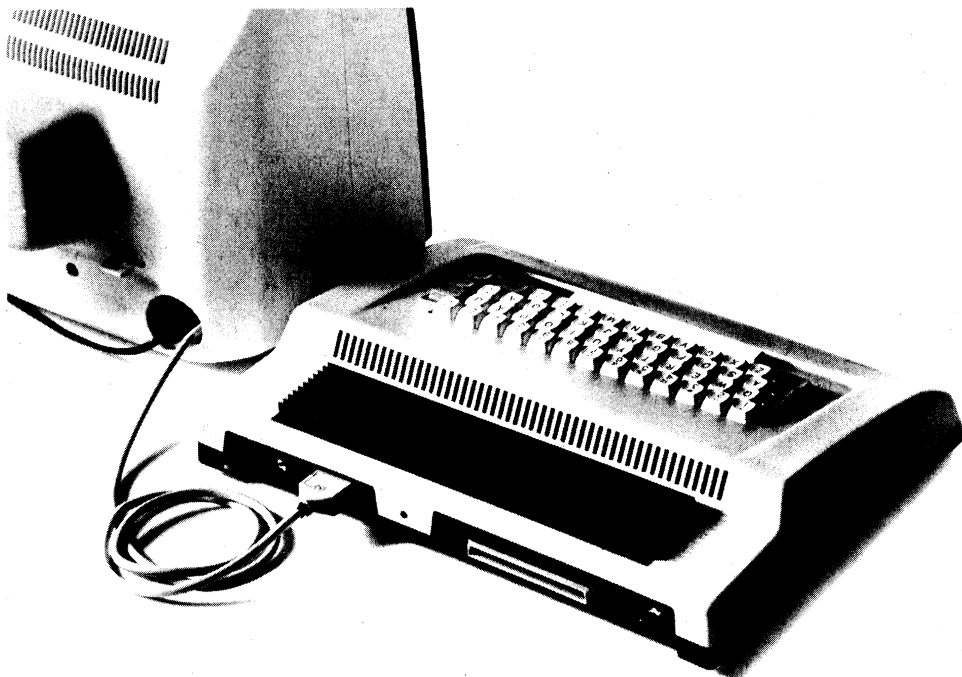
ABC 80 är ett mycket flexibelt datorsystem som kan bestå av ett stort antal delar. För de enkla tillämpningar, som vi behandlar i bokens fem första kapitel, utnyttjas dock bara två enheter, nämligen tangentbordet och TV-skärmen.



ABC 80 i grundutförande består av tangentbord och TV-skärm

DAGS FÖR START

Innan nätsladden ansluts måste du koppla ihop TV-skärmens andra kabel med tangentbordet.



Sammankoppling av enheterna

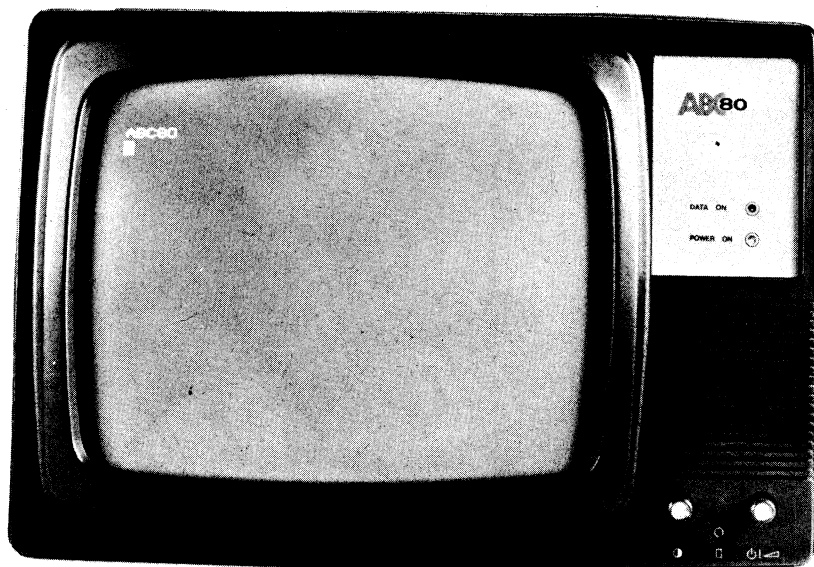
Nu kan du ansluta TV-skärmens nätsladd till vägguttaget. Starta datorn med nätströmbrytaren på TV-apparatens baksida. Ställ de båda rattarna för kontrast och volym i mittläge.

Datorn är klar för användning, när **ABC80** efter några sekunder visas i TV-skärmens övre vänstra hörn. Korrigera kontrasten så att bästa möjliga bild erhålls.

I fortsättningen använder vi följande skrivsätt för att visa utskrift på TV-skärmen.

ABC80

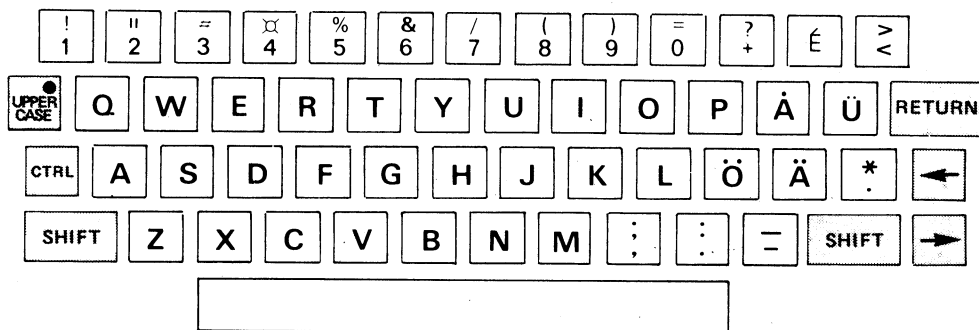
Den blinkande fyrkanten på TV-skärmen kallas markör (eng cursor). Markören anger var nästa tecken skrivs på skärmen.



ABC 80 klar för användning

TANGENTBORDET

Tangentbordet på ABC 80 fungerar som på en elektrisk skrivmaskin. Utskriften sker på TV-skärmen, som rymmer 24 rader om vardera 40 tecken. När du skriver flera rader, visas bara de 24 sist inskrivna. Alla tangenter är repeterande.



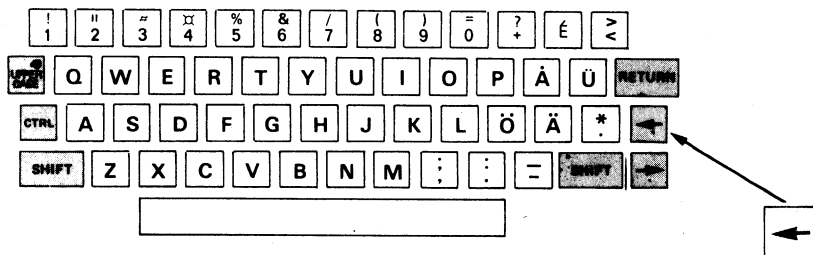
Tangentbordet

Prova nu ABC 80 genom att på tangentbordet skriva

PRINT

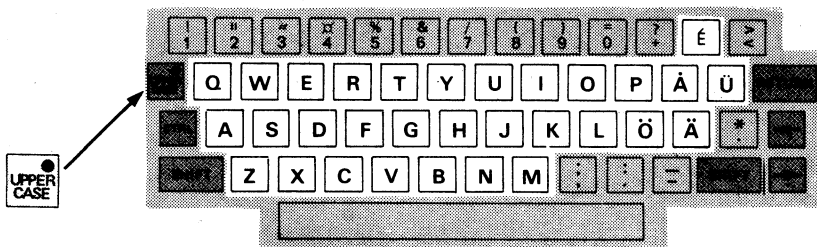
Skuggningen används i fortsättningen för att markera den text du själv ska skriva in.

Du kan lätt radera felslag genom att trycka på radertangenten 



Radertangentens placering

Radera det du skrivit genom att trycka ett antal gånger på radertangenten. Till en början ska vi för enkelhets skull bara använda stora bokstäver, något som du lättast åstadkommer genom att trycka på tangenten . Den lysdiod, som tänds i tangenten, betyder att små bokstäver inte längre skrivs. En ytterligare nedtryckning av tangenten  släcker lysioden och återställer tangentbordet till normal skrivmaskinsfunktion. Du kan då, som vanligt, tillfälligt välja stora bokstäver med skifftangenten .



Tangenter utan skuggning påverkas av tangenten 

BRUKSANVISNINGEN

Den här beskrivningen av ABC 80 är visserligen kortfattad, men du har nu förhoppningsvis lärt dig så mycket, att du kan börja använda maskinen. En utförligare teknisk beskrivning hittar du i den bruksanvisning som hör till maskinen.

KAPITEL 2

Datorn som intelligent skrivmaskin

ABC80

GE DATORN INSTRUKTIONER

ABC 80 är visserligen en tekniskt mycket fulländad skapelse, men för att den ska arbeta enligt dina önskemål måste du på något sätt förklara vad du vill ha utfört. ABC 80 är konstruerad så att den kan tolka vissa instruktioner, skrivna på tangentbordet. Instruktionerna är gemensamma för många minidatorer och består av korta engelska uttryck.

ETT FÖRSTA FÖRSÖK

Hur ska maskinen instrueras om du vill att den ska skriva ditt namn? Jo, skriv på tangentbordet det engelska ordet för "skriv", nämligen PRINT, följt av ditt namn inom citationstecken, t ex

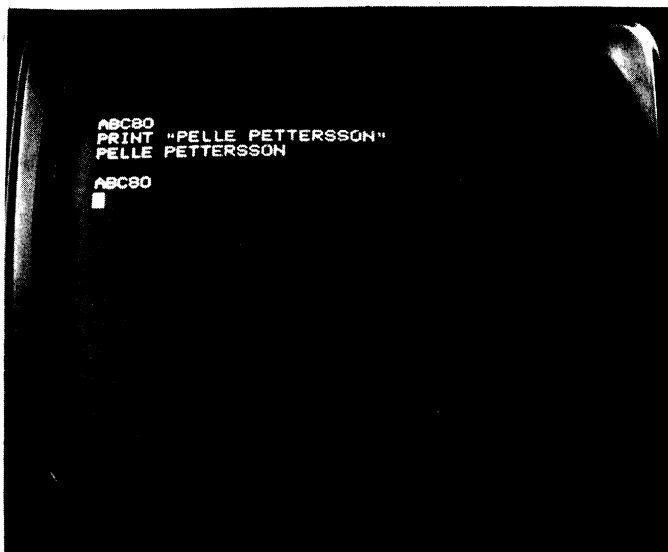
PRINT "PELLE PETTERSSON"

RETURN

Citationstecken erhålls om du håller tangenten SHIFT nedtryckt samtidigt som du trycker på tangenten "2".

När du vill avsluta raden, trycker du på returtangenten RETURN. Då tolkar ABC 80 din uppgift och skriver ditt namn på TV-skärmen.

På TV-skärmen står det nu



TV-skärmen efter första försöket

ABC80, som datorn skrev i sista raden, innebär att den utfört sitt uppdrag och är klar för nästa.

ETT LITET PROGRAM

När du vill skriva flera rader med instruktioner till datorn, måste varje rad förse med ett radnummer. ABC 80 får på detta sätt information om, i vilken följd instruktionerna ska utföras. Datorn utför normalt instruktionerna i stigande radnummerföljd. Ett antal numrerade rader med instruktioner till datorn kallas för program. Vi ska nu skriva ett litet program, genom att förse föregående instruktion med ett radnummer, t ex 10, och skriva den på nytt.

10 PRINT "PELLE PETTERSSON"

RETURN

För att du inte ska glömma att avsluta varje färdigskriven rad med returtaggen, markeras detta i hela kap 2 med RETURN i slutet på raden. Datorn vet ännu inte om programmet är färdigskrivet, eller om flera rader återstår. Därför utförs ännu inga instruktioner, utan du kan tillfoga ytterligare en rad, som instruerar datorn att skriva din postadress.

20 PRINT "580 05 LINKÖPING"

RETURN

Du måste nu förklara för ABC 80 att programmet är färdigskrivet och att det är dags för datorn att utföra programmet. Detta görs med kommandot RUN.

Skriv därför

RUN

RETURN

På TV-skärmen har nu tillkommit

10 PRINT "PELLE PETTERSSON"

20 PRINT "580 05 LINKÖPING"

RUN

PELLE PETTERSSON

580 05 LINKÖPING

ABC80

Ditt program består av de två raderna med radnummer 10 och 20. Programmet finns fortfarande kvar i datorns minne och du kan upprepa körningen genom att på nytt skriva

RUN

RETURN

Prova detta några gånger!

PROGRAMMET VÄXER

Du kan nu utöka programmet genom att tillfoga en rad för utskrift av din gatuadress.

```
15 PRINT "STORGATAN 10" RETURN
```

Kör det nya programmet på samma sätt som förut.

```
RUN RETURN
```

Vad händer? Jo, datorn utför instruktionerna i radnummerföljd och resultatet blir därför

```
PELLE PETTERSSON  
STORGATAN 10  
580 05 LINKÖPING
```

```
ABC80
```

Hela programmet finns fortfarande lagrat i minnet. Du kan kontrollera programmet med kommandot

```
LIST RETURN
```

TV-skärmen visar

```
10 PRINT "PELLE PETTERSSON"  
15 PRINT "STORGATAN 10"  
20 PRINT "580 05 LINKÖPING"
```

```
ABC80
```

Som du ser, har datorn automatiskt sorterat raderna i rätt radnummerföljd. Du kan fortfarande utöka programmet. Låt oss förändra utskriften genom att tillfoga en tom rad före och efter gatuadressen. Tillfoga därför raderna

```
12 PRINT RETURN  
18 PRINT RETURN
```

Kör programmet på vanligt sätt med

```
RUN RETURN
```

Som du ser har två tomma rader tillkommit.

PELLE PETTERSSON

STORGATAN 10

580 05 LINKÖPING

ABC80

TV-SKÄRMEN RADERAS

Du kan tömma TV-skärmen på text genom att hålla returtangenten nedtryckt några sekunder. Detta påverkar inte innehållet i minnet hos ABC 80, programmet finns fortfarande kvar.

Kontrollera genom att skriva

LIST **RETURN**

Ditt program visas då på TV-skärmen.

MINNET RADERAS

Om du vill skriva ett helt nytt program, måste du ange detta genom att skriva

NEW **RETURN**

Kommandot NEW raderar datorns minne. Förvissa dig om att det inte finns något program kvar i minnet genom att skriva

LIST **RETURN**

Endast **ABC80** skrivs på TV-skärmen, vilket innebär att det inte längre finns något program lagrat i minnet.

EN EVIGHETSMASKIN?

Som du har sett kan datorn på ett enkelt sätt fås att skriva text. Dessutom sorterar datorn själv raderna i rätt ordning. Följande exempel visar att datorn

dessutom kan utföra allt detta mycket snabbt. Börja ett nytt program med att skriva in ditt förnamn för utskrift.

10 PRINT "PELLE" **RETURN**

Tillfoga dessutom en rad, som är en uppmaning till datorn att gå tillbaka till rad 10 och utföra den raden på nytt.

20 GOTO 10 **RETURN**

Det program som du nu har matat in i datorns minne, kommer att utföras på följande sätt.

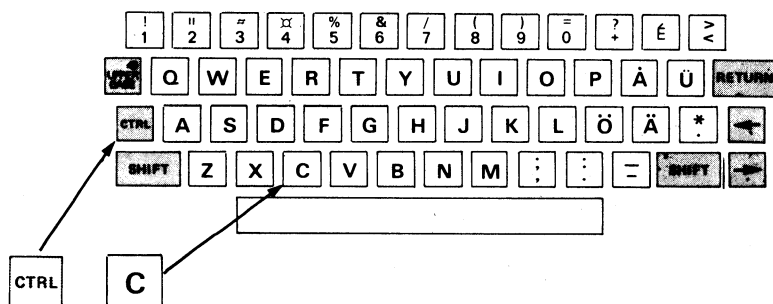
Rad 10 instruerar datorn att skriva ut ditt namn.

Rad 20 instruerar datorn att gå tillbaka till rad 10. Den skriver på nytt ditt namn, går sedan till rad 20 o s v.

Prova programmet genom att skriva

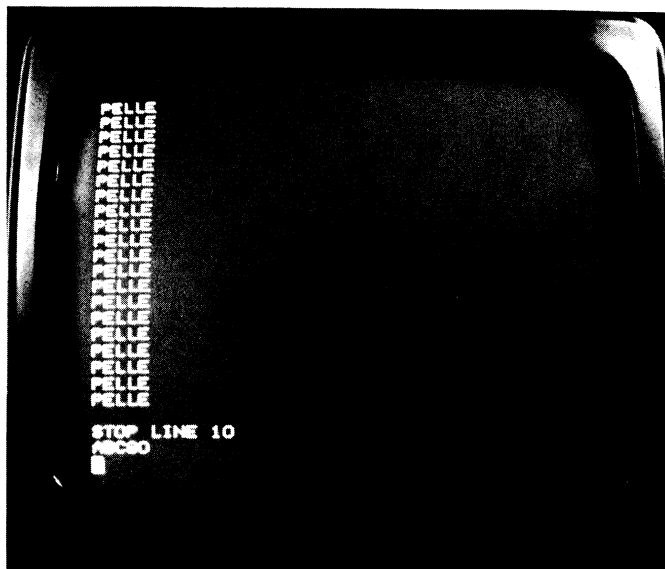
RUN **RETURN**

Som du märker, fyller datorn snabbt TV-skärmens vänstra kant med ditt namn och fortsätter med detta utan uppehåll. Datorn har fått en uppgift att utföra, som aldrig kan slutföras. Du kan avbryta utskriften genom att hålla kontrolltangenter **CTRL** nedtryckt samtidigt som du trycker på tangenten **C**.



Avbrott i programkörning

ABC 80 skriver då ut numret på den programrad, där den blev avbruten, samtidigt som den ger signal via högtalaren.



Avbruten utskrift

Utskriften STOP LINE 10 innebär att datorn avbröts, när den var sysselsatt med rad 10.

UTSKRIFT PÅ SAMMA RAD

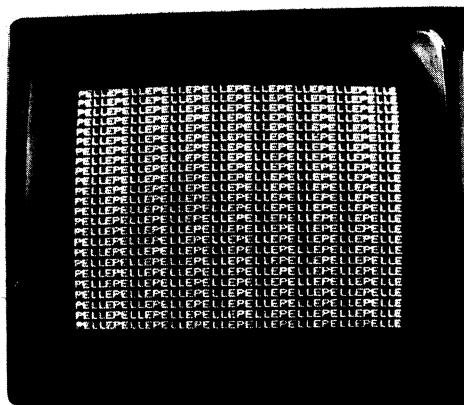
Om du inte vill att datorn ska börja skriva på ny rad, varje gång den utför uppgiften på rad 10 i föregående program, kan du lätt modifiera programmet.

Skriv om rad 10 igen men tillfoga nu ett semikolon sist i raden.

```
10 PRINT "PELLE"; RETURN
```

Observera vad som händer när du skriver

```
RUN RETURN
```



Semikolon ger packad utskrift

Utskriften går så snabbt, att den ibland kan vara svår att uppfatta. Avbryt programkörningen som tidigare med **CTRL** och **C**. Avslutar du en PRINT-rad med semikolon, sätts den normala radframmatningen ur spel och texten skrivs i en följd utan mellanrum.

UTSKRIFT I TABELLFORM

Skriv om raden 10 på nytt, men byt nu semikolon mot kommatecken.

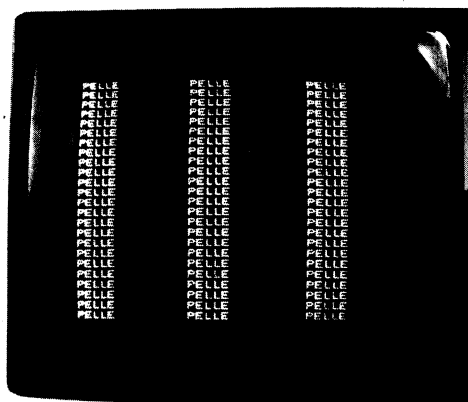
10 PRINT "PELLE",

RETURN

Kör programmet som vanligt.

RUN

RETURN



Kommatecken ger kolumner

Avbryt körningen med **CTRL** och **C**

Som du ser på skärmen, skrivs nu texten i kolumner. Om du avslutar en PRINT-rad med kommatecken, ersätts den normala radframmatningen med en framflyttning till efterföljande kolumn. Skärmen är uppdelad i 3 kolumner.

DATORSPRAKET BASIC

Ett datorspråk är en uppsättning regler och uttryck som används för att instruera datorer. Det datorspråk, som vi använder för att programmera ABC 80, kallas BASIC (Beginners All-purpose Symbolic Instruction Code). BASIC-språket konstruerades under sextiotalet vid Dartmouth College, med avsikt att möjliggöra för vem som helst att använda datorer. Till fördelarna med BASIC framför andra datorspråk kan räknas att BASIC är lätt att lära, att programmeringen sker i dialogform och att många färdiga program finns utvecklade. Du har redan använt några enkla BASIC-program, t ex

10 PRINT "PELLE PETTERSSON"

Ett BASIC-program kännetecknas av att instruktioner används för att bilda satser, som skrivs på rader försedda med radnummer.

radnummer	sats
10 PRINT "PELLE PETTERSSON"	
instruktion	data

SYSTEMKOMMANDON

Utöver själva BASIC-programmet har vi behov av kommandon till datorn, för att beskriva hur den ska hantera programmet. Exempel på sådana kommandon som vi använt är RUN, LIST och NEW.

	Kommentar
10 PRINT "PELLE PETTERSSON"	Program
RUN	Kommando
PELLE PETTERSSON	Resultat
ABC80	Kvittens på att programmet är utfört.

SAMMANFATTNING

Vi kommer i fortsättningen att avsluta varje kapitel med en sammanfattning av nya BASIC-instruktioner och kommandon som införts i kapitlet.

Instruktion	Beskrivning	Exempel
PRINT	Skriver på bildskärmen. Text måste omges med citationstecken.	10 PRINT "HEJ"
	Semikolon efter data eliminerar radformatning.	10 PRINT "HEJ";
	Kommatecken efter data ger framflyttning till nästa kolumn.	10 PRINT "HEJ",
GOTO	Får programkörningen att fortsätta på viss rad.	20 GOTO 10

Kommando	Beskrivning	Exempel
RUN	Kör det program, som finns i minnet.	RUN
NEW	Raderar gamla program i minnet.	NEW
LIST	Ger utskrift på bildskärmen av det program som finns i minnet. Följande varianter finns:	
	samtliga rader	LIST
	endast en rad	LIST 100
	allt mellan två radnummer.	LIST 100-200
	allt t o m viss rad	LIST - 200
	allt fr o m viss rad	LIST 100 -
	Avbryter programkörningen.	



ÖVNINGSEXEMPEL

OBS! Använd kommandot NEW mellan varje övningsexempel.

- 2.1 Vilken utskrift ger följande program?

```
10 PRINT "HEJ"  
20 PRINT "PÅ"  
30 PRINT "DEJ"
```

- 2.2 Vilken utskrift ger följande program?

```
10 PRINT "HEJ",  
20 PRINT "PÅ",  
30 PRINT "DEJ"
```

- 2.3 Hur kan programmet i exempel 2.2 förändras om utskriften ska se ut så här?

HEJ PÅ DEJ

Observera att vi vill ha ett mellanslag mellan orden!

- 2.4 På tangentbordet till ABC 80 skrivs följande rader

```
10 PRINT "PELLE PETTERSSON"  
20 PRINT "580 05 LINKÖPING"  
15 PRINT "STORGATAN 10"  
10 PRINT "ANNA PETTERSSON"
```

Vad händer när du kör programmet med RUN?

- 2.5 Var på TV-skärmen hamnar utskriften från nedanstående program?

```
10 PRINT, "LISA PETTERSSON"  
20 PRINT, "STORGATAN 10"  
30 PRINT, "580 05 LINKÖPING"
```

- 2.6 Fyll i de radnummer som saknas efter instruktionerna GOTO i nedanstående program om man önskar utskriften "OK!" på TV-skärmen.

```
10 GOTO .....  
20 PRINT "K";  
30 GOTO .....  
40 PRINT "O";  
50 GOTO .....  
60 PRINT "I"
```

Provkör för kontroll.

- 2.7 I vilken kolumn hamnar utskriften av telefonnumret i nedanstående programexempel.

```
10 PRINT "PELLES TEL.NR. ÄR",  
20 PRINT "123456"
```

KAPITEL 3

Datorn söker i register

ABC80

ETT NAMN BLIR STRÄNGVARIABEL

Om du i ett program ofta använder exempelvis namn, kan du istället för namnet använda en kortare beteckning, som kallas variabel. En variabel, som kan betyda både bokstäver och siffror, kallas strängvariabel, och betecknas med en bokstav och valutatecknet A (utläses vanligen "sol"), t ex $A\text{A}$. Tecknet A skriver du genom att använda tangenterna SHIFT och A . Vill du i ett program ofta använda ditt namn, kan du tilldela en strängvariabel värdet av ditt namn. Detta sker med BASIC-instruktionen LET, som betyder tilldela.

Innan du tittar på hur instruktionen LET används måste datorns minne raderas.

Skriv därför

NEW

(I fortsättningen påminner vi inte om RETURN som krävs i slutet på varje rad).

Följande programrad ger variabeln $A\text{A}$ värdet PELLE PETTERSSON.

10 LET $A\text{A}$ = "PELLE PETTERSSON"

Observera att citationstecknen måste vara med kring namnet. Vi ska nu visa att $A\text{A}$ efter rad 10 betyder "PELLE PETTERSSON".

Tillfoga raden

20 PRINT $A\text{A}$

Kör programmet genom att skriva

RUN

Efter körningen får du en utskrift av namnet. På skärmen ska stå exempelvis

10 LET $A\text{A}$ = "PELLE PETTERSSON"

20 PRINT $A\text{A}$

RUN

PELLE PETTERSSON

ABC80

DATAINMATNING UNDER PROGRAMKÖRNING

Strängvariabeln A\$ kan tilldelas värden på andra sätt än med instruktionen LET. Låt oss se på ett alternativ. Ändra raden 10 genom att skriva en ny rad med radnummer 10.

```
10 INPUT A$
```

När datorn under programkörningen kommer till rad 10, så avbryts körningen tills du via tangentbordet matat in ett värde på strängvariabeln A\$. Kontrollera nu att du skrivit in programmet korrekt.

```
LIST
```

```
10 INPUT A$  
20 PRINT A$
```

ABC80

Kör programmet med RUN på vanligt sätt. Vad händer? Jo, datorn skriver ett frågetecken på skärmen. Det betyder att du nu via tangentbordet ska ange ett värde på A\$. Skriv t ex ditt namn. Kom ihåg att det måste stå inom citationstecken.

När du tryckt på efter ditt namn fortsätter datorn programkörningen på rad 20, som medför att värdet av strängvariabeln A\$, d v s ditt namn, skrivs på TV-skärmen.

FLERA VARIABLER I SAMMA SATS

Om du samtidigt vill tilldela flera strängvariabler värden i en INPUT-sats, behöver de olika variablerna bara åtskiljas med kommatecken.

```
10 INPUT A$,B$  
20 PRINT A$,B$
```

När du kör programmet skrivs liksom tidigare ett frågetecken på skärmen. Mata då in värden på variablerna A\$ och B\$, t ex ditt namn och ditt telefonnummer åtskilda av kommatecken.

```
? "PELLE", "123456"
```

Efter RETURN skriver datorn värdena på A\$ och B\$ enligt programrad 20.

Eftersom det stod kommatecken mellan A $\square$ och B $\square$, skrivs A $\square$ i kolumn 1 och B $\square$ i kolumn 2.

PELLE 123456

ABC80

YTTERLIGARE TILLDELNINGSLTERNATIV

Om du i ett program vill tilldela strängvariabler olika värden utan att programkörningen avbryts, kan du använda instruktionerna READ och DATA.

Låt oss titta på ett exempel.

```
10 READ A  $\square$ , B  $\square$ 
20 DATA "PELLE", "123456"
30 PRINT A  $\square$ , B  $\square$ 
```

När du beordrar körning med RUN händer följande. Rad 10 utförs och datorn söker efter en DATA-sats någonstans i programmet. När den hittar en sådan tilldelas A $\square$ det första värdet i DATA-satsen (PELLE) och B $\square$ det andra värdet (123456). Med hjälp av PRINT-satsen på rad 30 skrivs de tilldelade värdena på A $\square$ och B $\square$ ut.

DATORN SIGNALERAR NÄR DU GÖR FEL

Vad händer om du tillfogar följande rad?

```
40 GOTO 10
```

Jo, datorn går till programrad 10 och försöker läsa in nya värden på A $\square$ och B $\square$. Men DATA-satsen i rad 20 förbrukas första gången rad 10 utförs vid varje körning och ingen ytterligare DATA-sats finns.

Prova genom att skriva RUN. ABC 80 reagerar med en akustisk signal och skriver

PELLE 123456

**ERR 30 LINE 10
ABC80**

Felmeddelande ges alltid när någon ologisk eller otillåten operation anges. Se vidare appendix 2.

För att kontrollera att nya värden tilldelas variablerna A α och B α om ytterligare DATA-satser tillfogas, utökar vi programmet med raderna

```
21 DATA "OLLE", "234567"  
29 DATA "KARIN", "345678"
```

Skriv LIST och förvissa dig om att programmet ser ut som följer.

```
LIST  
10 READ A  $\alpha$ , B  $\alpha$   
20 DATA "PELLE", "123456"  
21 DATA "OLLE", "234567"  
29 DATA "KARIN", "345678"  
30 PRINT A  $\alpha$ , B  $\alpha$   
40 GOTO 10
```

ABC80

När du kör programmet får du resultatet

PELLE	123456
OLLE	234567
KARIN	345678

ERR 30 LINE 10
ABC80

Som du märker, får vi även nu en felutskrift som berättar att datorn försöker läsa fler värden till A α och B α , än vi har angett i DATA-satserna. Detta beror på att datorn går från rad 40 till rad 10 även när alla datasatser är förbrukade. Felet kan korrigeras genom att du tillfogar raderna

```
35 IF A  $\alpha$  = "KARIN" THEN GOTO 100  
100 PRINT "KLART"
```

Den tillförda rad 35 innebär att ABC 80 testar om A α har tilldelats värdet KARIN i den sista DATA-satsen. Om A α inte har fått värdet KARIN, så fortsätter datorn till nästa rad och går tillbaka till rad 10. Om däremot A α har tilldelats värdet KARIN, så går datorn till rad 100. Rad 100 är en PRINT-sats, för utskrift av KLART. Efter rad 100 finns inga ytterligare rader, d v s programmet är slut. Kör programmet och konstatera att du inte får någon felutskrift.

RUN

PELLE	123456
OLLE	234567
KARIN	345678
KLART	

ABC80

För att vi ska kunna repetera vad som händer i programmet skriver du LIST.

LIST

```
10 READ A$, B$
20 DATA "PELLE", "123456"
21 DATA "OLLE", "234567"
29 DATA "KARIN", "345678"
30 PRINT A$, B$
35 IF A$ = "KARIN" THEN GOTO 100
40 GOTO 10
100 PRINT "KLART"
```

ABC80

I rad 10 tilldelas variablerna A\$ och B\$ värden från den första DATA-satsen med hjälp av instruktionen READ. Satserna READ och DATA förekommer alltid tillsammans. DATA-satserna kan placeras var som helst i programmet och vara godtyckligt många. Med READ hämtas värden från DATA-satserna efter stigande radnummer. Rad 30 medför utskrift av det aktuella värdet på variablerna A\$ och B\$. I rad 35 undersöker datorn om A\$ har tilldelats värden från den sista DATA-satsen. Om så är fallet går datorn till rad 100 och skriver KLART, annars fortsätter den till rad 40. I rad 40 går datorn tillbaka till rad 10 för att hämta nya variabelvärden.

EN LITEN TELEFONKATALOG

Det program vi nu har skrivit är ju ett program som skriver ut en liten telefonkatalog. Du kan lätt utöka katalogens omfattning genom att tillfoga ytterligare DATA-satser med flera namn och telefonnummer, t ex

```
22 DATA "KURT", "122113"
23 DATA "GUNNAR", "433415"
24 DATA "LENA", "546326"
```

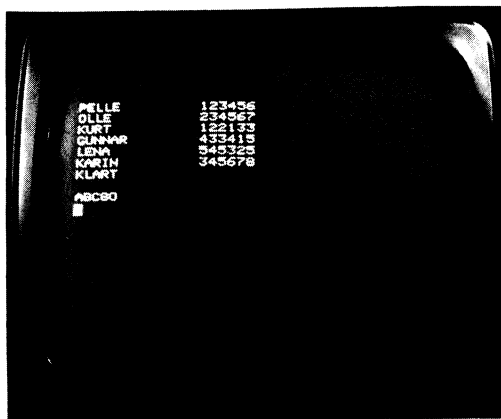
Om du skriver datasatser med högre nummer än 29, måste du ändra rad 35 så att testen avser det sista värdet som AÖ tilldelas.

LUFTIGARE UTSKRIFT

För att göra utskriften på TV-skärmen luftigare kan man tillföra en blankrad före utskriften.

5 PRINT

Kör programmet med RUN och titta på din egen telefonkatalog.



Datorutskriven telefonkatalog

DATORN SÖKER I TELEFONKATALOGEN

Låt oss nu utveckla vårt program så att datorn själv söker reda på telefonnummer i telefonkatalogen.

Tillfoga raderna

```
6 PRINT "TILL VEM ÖNSKAS TELEFONNR"  
7 INPUT NÖ
```

Rad 6 innebär utskrift av förklarande text inför den kommande INPUT-satsen.

Rad 7 medför att ett frågetecken skrivs ut, varefter datorn väntar på att variabeln NÖ ska tilldelas ett värde via tangentbordet. Variabeln NÖ tilldelas värdet lika med namnet på den person, vars telefonnummer du söker.

Skriv om rad 30.

```
30 IF NÖ = AÖ THEN GOTO 50
```

Rad 30 innebär att datorn testar om det inmatade namnet är lika med det namn ur telefonkatalogen, som A $\square$ för tillfället representerar. Om N $\square$ = A $\square$ ska det samhörande värdet på B $\square$ skrivas på TV-skärmen. Detta värde på B $\square$ är det telefonnummer du sökt. Enligt rad 30 ska datorn gå till rad 50, när den funnit telefonnumret. Rad 50 måste alltså bestå av en PRINT-sats som ombesörjer utskrift av rätt telefonnummer.

50 PRINT "NUMRET ÄR", B $\square$

Om inte N $\square$ = A $\square$ i rad 30 så fortsätter datorn till rad 35. Där testas om A $\square$ tilldelats det sista värdet i telefonkatalogen. Om så inte är fallet, går datorn vidare till rad 40, som medför återgång till rad 10. I rad 10 tilldelas A $\square$ och B $\square$ nya värden från nästa DATA-sats.

Ditt färdiga program ska nu se ut så här, frånsett DATA-satserna, där du kanske haft egna förslag.

LIST

```
5 PRINT
6 PRINT "TILL VEM ÖNSKAS TELEFONNR"
7 INPUT N $\square$ 
10 READ A $\square$  , B $\square$ 
20 DATA "PELLE","123456"
21 DATA "OLLE","234567"
22 DATA "KURT","122113"
23 DATA "GUNNAR","433415"
24 DATA "LENA","545325"
29 DATA "KARIN","345678"
30 IF N $\square$  = A $\square$  THEN GOTO 50
35 IF A $\square$  = "KARIN" THEN GOTO 100
40 GOTO 10
50 PRINT "NUMRET ÄR",B $\square$ 
100 PRINT "KLART"
```

ABC 80

Exempel på programkörning.

RUN

```
TILL VEM ÖNSKAS TELEFONNR
?LENA
NUMRET ÄR      545325
KLART
```

ABC80

Kör programmet flera gånger och prova med olika namn, som finns i DATA-satserna.

Vad händer om du svarar med ett namn, som inte finns i någon DATA-sats? Jo, datorn skriver KLART, beroende på att villkoret för att datorn ska gå till rad 100 från rad 35 uppfylldes, när alla datasatser provats.

REGISTERSÖKNING TYPISK DATORUPPGIFT

Ett program, som söker rätt på nummer i en telefonkatalog, behöver tydligen inte bli speciellt omfattande och programmet kan lätt utökas med godtyckligt antal datasatser. Normalt när datorer söker i större register lagras inte data i datasatser utan på andra sätt. Principen för sökning i register är dock densamma, även om lagringsformerna för data kan variera. De exempel som finns i detta kapitel illustrerar därför ett mycket vanligt användningsområde för datorer.

DATALAGEN MÅSTE FÖLJAS

Den nya datatekniken gör att stora mängder uppgifter kan samlas in och föras ihop på ett helt annat sätt än tidigare. När ett register finns på data, går det också mycket snabbare och lättare att hantera de olika uppgifterna. För att skydda den personliga integriteten finns därför olika lagar, som reglerar hur personuppgifter får registreras på data. Till personuppgifter räknas då inte bara namn, adress och personnummer, utan alla uppgifter som rör enskilda människor och deras förhållanden som t ex skattebetalare, anställda, bilägare, bankkunder och telefonabonnenter. Detta innebär att även register av den typ som behandlas i detta kapitel principiellt är personregister i lagens mening. Den statliga myndighet som ser till att lagarna följs är Datainspektionen (DI). Datainspektionen ger även tillstånd för att upprätta dataregister. Kontakta gärna

Datainspektionen
Bergsgatan 51
Box 12050
102 22 STOCKHOLM
Tel 08/22 79 80

för ytterligare upplysningar.

SAMMANFATTNING

Instruktion	Kort beskrivning	Exempel
LET	Tilldelar en variabel ett värde. Man kan i ABC 80 BASIC skriva uttrycket direkt utan LET.	10 LET A\$="KALLE" 10 A\$="KALLE"
INPUT	Väntar på inmatning av data från tangentbordet. Detta data måste omges av citationstecken enbart då mellanslag, komma eller citationstecken ingår i strängen.	10 INPUT A\$
READ	Läser variabelvärde från en DATA-sats.	10 READ A\$
DATA	Lagrar variabelvärden som ska läsas av en READ-sats. När mellanslag, kommatecken eller citationstecken inte ingår i strängen kan citationstecken utelämnas. DATA-satser kan stå var som helst i programmet.	20 DATA "HEJ DU" 30 DATA KALLE
IF...THEN...	Om villkoret är uppfyllt fortsätter programkörningen med instruktionen efter THEN, annars med nästa rad.	20 IF A\$ = B\$ THEN GOTO 100

Ovanstående BASIC-instruktioner kan användas även för att hantera variabler som bara innehåller siffror. Exempel på detta kommer i kapitel 4.

ÖVNINGSEXEMPEL

- 3.1 Programmet på sid 26 ger utskriften KLART även om man söker telefonnumret till en person, som inte finns i registret. Utöka programmet så att utskriften FINNS EJ MED I REGISTRET erhålls i sådana fall.
- 3.2 Ändra programmet i uppgift 3.1 så att sökning kan ske på telefonnummer istället för på namn. Utskrift ska ske av namnet på den, som har det inmatade telefonnumret.
- 3.3 En framsynt beg bilhandlare, som skaffat ABC 80, har skrivit nedanstående program för att underlätta kundernas val i det stora sortimentet.

```
10 PRINT
20 PRINT "VILKET BILMÄRKE ÖNSKAS"
30 INPUT A$
40 PRINT
50 PRINT "MÄRKE", "FÄRG", "MIL"
60 PRINT
70 READ B$, C$, D$
80 IF A$ = B$ THEN GOTO 200
90 IF B$ = "SLUT" THEN GOTO 220
100 GOTO 70
110 DATA "FIAT", "BLÅ", "4000"
120 DATA "SAAB", "GRÖN", "5000"
130 DATA "VOLVO", "GRÖN", "1000"
140 DATA "SAAB", "GUL", "2000"
150 DATA "VOLVO", "BLÅ", "12000"
190 DATA "SLUT", "SLUT", "SLUT"
200 PRINT B$, C$, D$
210 GOTO 70
220 PRINT
```

Avsikten med programmet är att kunden skall kunna få en lista omfattande samtliga bilar av ett visst märke, som finns i lagret. Resultatet av en körning visas här.

```
VILKET BILMÄRKE ÖNSKAS
?VOLVO
```

MÄRKE	FÄRG	MIL
VOLVO	GRÖN	1000
VOLVO	BLÅ	12000

En dag skaffar bilhandlaren följande två bilar, en röd Volvo som gått 3000 mil och en blå Toyota som gått 6000 mil. Skriv in två DATA-satser på lämplig plats i programmet så att registret blir aktuellt och kör programmet på nytt.

Hur ska programmet förändras, när bilhandlaren säljer den gröna Volvon, som gått 1000 mil?

- 3.4 Den ovan omtalade beg bilhandlaren upptäcker så småningom att många kunder är mera intresserade av den tilltänkta bilens färg än av dess fabrikat. Hur ska programmet modifieras, så att listan istället omfattar alla bilar av en viss färg?
- 3.5 Sekreta polisen i Grönköping har upprättat en hemlig lista över presumtiva landsförrädare. Deras personnummer är

181207-9273	210106-9249	250428-9253
261103-8387	270805-9130	281122-8390
350109-6998	350529-6776	361109-7076

Skriv ett BASIC-program som undersöker huruvida en misstänkt häktad person tillhör de presumtiva landsförrädarnas skara. Personnumret matas in via en INPUT-sats. Ett register av denna typ kräver normalt Datainspektionens tillstånd. De i exemplet använda personnumren är hämtade ur en förteckning över personnummer, som enligt Datainspektionen får användas för testkörningar.

KAPITEL 4

Datorn utför beräkningar

ABC80

DATOR ELLER RÄKNEDOSA?

Du har hittills inte använt datorn för några beräkningsuppgifter, men givetvis är datorn lika användbar som en avancerad räknedosa, när det gäller att utföra beräkningar. Den stora fördelen med datorn är emellertid att resultatet kan redovisas på ett överskådligt sätt, vilket hänger samman med att datorn kan presentera både omfångsrik text och många siffror samtidigt på TV-skärmen. En räknedosa visar ju normalt bara ett enda tal på ca 10 siffror åt gången.

KONTOUTDRAG MED DATOR

Det kontoutdrag du får från banken visar transaktionerna på exempelvis ditt checkkonto. Siffrorna på kontoutdraget är normalt resultat av datorberäkningar.

II

Bankens noteringar	Datum	Beloppet avser	Insättning/Uttag	Saldo, kronor	öre
	1230	ING SALDO		****	720811
5202030425	0102	6329526	20000-		520811
6627277	0108	MINUTEN	50000-		470811
6632895	0112	MINUTEN	50000-		420811
6608833	0115	MINUTEN	50000-		370811
8901002169	0116	6329527	30000-		340811
8901006836	0121	MINUTEN	50000-		290811
9570001	0127	LÖN	389831		680642
		43247725			
6610801	0127	MINUTEN	50000-		610642
ÖVERF	0127	39604186	20000-		610642
		UTC SALDO		****	610642

Exempel på kontoutdrag

Ett enkelt BASIC-program, som du själv kan använda för att bokföra förändringar på ett bankkonto, kan se ut så här.

```

10 PRINT
20 PRINT "MATA IN INGÅENDE SALDO";
30 INPUT A
40 PRINT "MATA IN AKTUELL TRANSAKTION";
50 INPUT B
60 PRINT
70 LET A = A + B
80 PRINT "TRANSAKT.", "NYTT SALDO"
90 PRINT B, A
100 PRINT
110 GOTO 40

```

Det är nu dags att ta fram kontoutdraget och prova programmet. En insättning av t ex 80 kr matas in med talet 80. Ett uttag av t ex 100 kr matas in med talet - 100. En tillgång eller en skuld representeras på motsvarande sätt i saldot.

Exempel:	Ingående saldo	1000
	Transaktion	- 1500
	Nytt saldo	- 500

Förklaringen till programmet ges av följande uppställning.

Radnummer	Radens funktion
10, 60, 100	Mellanrum skapas i texten.
20, 40	Ger ledtext som talar om vilka värden som ska anges.
30, 50	Aktuell variabel tilldelas värde via tangentbordet.
70	LET A = A + B innebär att det gamla saldot A ersätts med summan av det gamla saldot A och transaktionen B. A representerar nu värdet av det nya saldot. Det gamla saldot (föregående värdet av A) finns utskrivet på bildskärmen.
80	Text som förtydligar utskriften.
90	Värdet av variablerna B och A, d v s beloppen för transaktion resp saldo, skrivs på TV-skärmen.
110	Efter det att uppgifterna på raderna 10 - 100 utförts, återgår datorn till att utföra instruktionen på rad 40.

Datorn utför programmets instruktioner så länge du fortsätter att mata in nya transaktioner. Du kan avbryta programkörningen genom att inte mata in något tal utan bara trycka på returtangenten. Genom att inte mata in något tal, gör du ur datorns synpunkt ett fel. Datorn avbryter då programkörningen och ger felutskriften

ERR 12 LINE 50
ABC80

Som du själv kan kontrollera i appendix 2 betyder ERR 12 "Felaktigt tal", d v s i detta fall inget tal alls.

TALVARIABLER ANVÄNDS VID BERÄKNINGAR

Som du såg, saknade variabelnamnen α -tecken i föregående program. α -tecknet kallas strängtecken och en variabel som innehåller α -tecknet kallas strängvariabel. Strängvariabler kan representera både siffror, bokstäver och andra tecken. En variabel utan α -tecken kallas talvariabel och kan bara representera tal. Av dessa variabeltyper är det normalt bara talvariabler som används vid beräkningar.

BEKVÄM REDIGERING

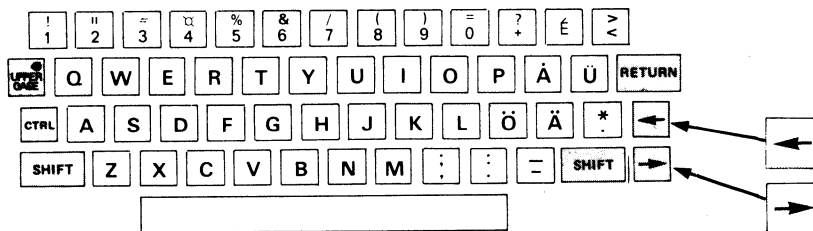
I kapitel 2 har du lärt dig hur man använder radertangenten för att korrigera felslag. Hur utför man en ändring i en redan avslutad programrad? En möjlighet är att du själv skriver om raden i ändrat skick. Kommandot ED (förkortning av eng EDIT) erbjuder emellertid en bekvämare möjlighet. Antag att du vill byta ut ordet AKTUELL mot NY på rad 40. Korrigeringen kan du då göra så här. Skriv

ED 40

Datorn svarar då med

40 PRINT "MATA IN AKTUELL TRANSAKTION"

Håll nu kopieringstangenten  nedtryckt till dess att datorn på nytt skrivit programrad 40 t o m ordet AKTUELL.



Radertangent och kopieringstangent

Avlägsna nu ordet AKTUELL med radertangenten  och tillfoga ordet NY. Återstoden av raden ska bibehållas oförändrad. Detta åstadkommer du genom att hålla kopieringstangenten nedtryckt till dess hela rad 40 är utskriven på TV-skärmen. Som vanligt avslutar du raden genom att trycka på returtangenten. Öva gärna några gånger på att redigera programrader med ED. Kom ihåg att alla tecken som är felaktiga avlägsnas med radertangenten. De tecken som ska bibehållas, tas fram på bildskärmen med kopieringstangenten.

BERÄKNING AV MOMS

De flesta företag är skyldiga att redovisa mervärdeskatt (moms). För dessa företag är det en del av den dagliga rutinen att antingen beräkna momsen, eller att leta i en momstabell efter det sökta beloppet. Om ABC 80 används för beräkningen, kan alla ingående belopp presenteras samtidigt på TV-skärmen. Risken minskar för att ett manuellt misstag ska orsaka fel, eftersom samtliga belopp med förklarande text finns tillgängliga för kontroll. Ett enkelt program för momsberäkningen kan se ut så här.

Kom ihåg att alltid radera gamla program i datorns minne med NEW!

```
10 PRINT
20 PRINT
30 PRINT "PRISET FÖRE MOMS",
40 INPUT P
50 PRINT "PRISET INKL MOMS", P+P*0.2063
60 PRINT "DÄRAV UTGÖR MOMS", P*0.2063
70 GOTO 10
```

Körning av programmet ger t ex följande utskrift.

PRISET FÖRE MOMS	7500
PRISET INKL MOMS	603.15
DÄRAV UTGÖR MOMS	103.15

PRISET FÖRE MOMS	?
------------------	---

ANVÄND RÄTT SYMBOL

I BASIC används decimalpunkt istället för decimalkomma, för att åtskilja heltalsdel och decimaler. Detta beror på att decimalpunkten är den internationellt vedertagna symbolen. Om heltalsdelen är noll, skriver datorn endast ut decimalpunkten och decimalerna. Talet 0.2063 skrivs alltså .2063 av datorn. Du kan även själv använda detta skrivsätt vid inmatning till datorn. För att undvika förväxling av decimalpunkt och multiplikationstecken har man i BASIC valt att använda tecknet * för multiplikation.

HUR STORA TAL KLARAR DATORN?

En körning av momsprogrammet med pris före moms 1 000000 ger resultatet

PRISSET FÖRE MOMS	?1000000
PRISSET INKL MOMS	1.2063E+6
DÄRAV UTGÖR MOMS	2.063E+5

Datorn arbetar internt med sex siffrors noggrannhet. Mycket stora och mycket små tal presenteras därför i tiopotensform på TV-skärmen. Normalt brukar talet 1206300 skrivas med hjälp av tiopotenser på följande sätt.

$$1,2063 \cdot 10^6$$

Sexan i 10^6 kallas exponent. I ABC 80 anges tiopotenser med hjälp av bokstaven E följt av exponenten. Exponenten föregås alltid av tecken. Ovanstående tal kommer då i ABC 80 att anges på följande sätt med tiopotenser.

$$1.2063 \text{ E}+6$$

I detta fall är exponenten positiv. Om exponenten är negativ ska det stå ett minustecken efter E. Vid inmatning kan tal anges antingen på tiopotensform eller på vanligt sätt. Datorn kan hantera svindlande stora tal, övre gränsen går vid $0.999999\text{E}+127$. Motsvarande minsta tal är $0.1\text{E}-127$. Prova potensformen vid inmatning av data i momsprogrammet.

1000	inmatas som $1\text{E}+3$
127300	inmatas som $1.273\text{E}+5$

RESULTAT UTAN DECIMALER

En körning av föregående program med pris före moms 506.50 ger följande utskrift.

PRISSET FÖRE MOMS	?506.50
PRISSET INKL MOMS	610.991
DÄRAV UTGÖR MOMS	104.491

En nackdel med den här presentationen är att datorn skriver för många decimaler. Beloppet 610.991 borde hellre skrivas 610.99. Som en åtgärd väljer vi

att modifiera programmet, så att enbart antalet kronor tas med i utskriften. Det är möjligt att utesluta alla decimaler i ett tal genom att beräkna "heltalsdelen" av talet.

Programraden

50 PRINT "PRISET INKL MOMS", INT (P + P*0.2063)

tolkas av datorn så att enbart heltalsdelen av uttrycket inom parentesen skrivs på TV-skärmen. INT (förkortning av eng INTEGER) betyder heltal. INT är en sk funktion, som påverkar beräkningsuttryck. Funktioner behandlas utförligare i kapitel 9. Tillfoga alltså beteckningen INT framför beräkningsuttrycken inom parentes på raderna 50 och 60. Programmet kommer att se ut så här.

```
10 PRINT
20 PRINT
30 PRINT "PRISET FÖRE MOMS",
40 INPUT P
50 PRINT "PRISET INKL MOMS", INT(P + P * .206
3)
60 PRINT "DÄRAV UTGÖR MOMS", INT(P * .2063)
70 GOTO 10
```

TV-skärmens rader rymmer maximalt 40 tecken, medan en programrad kan innehålla ända upp till 120 tecken. Programraden 50 måste därför utnyttja två rader på TV-skärmen. Programrad 60 innehåller 40 tecken och ryms precis på en enda rad. Markören, som anger var nästa tecken skrivs, flyttas dock automatiskt till nästa rad på TV-skärmen. Avslutning av programraden sker som vanligt genom att du trycker på returtangenten.

Testa programmet på nytt med samma data som förut!

PRISET FÖRE MOMS	?506.50
PRISET INKL MOMS	610
DÄRAV UTGÖR MOMS	104

RÄKNEOPERATIONERNAS RANGORDNING

I BASIC-program markeras division med tecknet /. Talet 5 dividerat med 3 skrivs alltså i BASIC som 5/3. Vid sammansatta uttryck tillämpar datorn de normala prioritetsreglerna, d v s multiplikation och division utförs före addition och subtraktion. Vid lika prioritet utförs operationerna från vänster till höger. Prioritetsordningen kan alltid ändras med parenteser. Parenteser upphäver alltid normal prioritet.

Några exempel visar tekniken.

Skrivet på vanligt sätt

$$5(3+2)$$

$$\frac{5}{3+2}$$

$$\frac{5+7}{3+2}$$

Skrivet i BASIC

$$5*(3+2)$$

$$5/(3+2)$$

$$(5+7)/(3+2)$$

Tänk igenom nedanstående program och kontrollera resultatet på datorn.

```
10 LET A = 2
20 LET B = 6/2 + 5 * A
30 LET A = (A + B)/5
40 PRINT A/3
```

MÅNGA MÖJLIGHETER ÅTERSTÅR

De beräkningar vi utfört i detta kapitel har bara utnyttjat de fyra enkla räknesätten. Givetvis kan datorn användas för mera avancerade beräkningar. Den förmåga till grafisk presentation som ABC 80 har, erbjuder ytterligare möjligheter att presentera resultatet på ett överskådligt sätt. En utförlig genomgång av hur datorn används vid behandling av matematiska problem återfinns i kapitel nio och tio.

VARIABELTYPER

Du har inte stött på några nya instruktioner i detta kapitel. Vi har enbart utvidgat användningsområdet för de instruktioner som introducerats i tidigare kapitel. Instruktionerna kan användas tillsammans med olika variabeltyper. I ABC 80 finns strängvariabler och talvariabler. Strängvariabler används för text och talvariabler huvudsakligen för beräkningar. Talvariablerna kan indelas i flyttalsvariabler och heltalsvariabler. De variabler som använts för beräkningar i detta kapitel är flyttalsvariabler. Heltalsvariabler används för speciella ändamål. De tre variabeltypernas egenskaper framgår av nedanstående sammanställning.

Flyttalsvariabler

Flyttalsvariabler har namn som består av en bokstav eller en bokstav och en siffra. En flyttalsvariabel kan anta värden från $\pm 0.1\text{E}-127$ till $\pm 0.999999\text{E}+127$. En flyttalsvariabel är alltid nollställd första gången den används i ett program. Exempel på flyttalsvariablers namn är A, B3, Ö9.

Heltalsvariabler

Heltalsvariabler har namn som flyttalsvariabler men åtföljt av procenttecken, %. En heltalsvariabel kan endast anta heltalsvärden från - 32768 till + 32767. En heltalsvariabel är alltid nollställd första gången den används i ett program. Exempel på heltalsvariablers namn är A%, B3%, Ö9%.

Strängvariabler

Strängvariabler har namn som flyttalsvariabler men åtföljt av strängtecknet, $\text{\textasciitilde}$. (Motsvarar \$ i amerikansk datorlitteratur.) En strängvariabel i ABC 80 kan innehålla ett godtyckligt antal tecken. Om strängen ska innehålla mera än 80 tecken, måste dock detta anges med en speciell instruktion (DIM), som genomgås i ett senare kapitel. Exempel på strängvariablers namn är A $\text{\textasciitilde}$, B3 $\text{\textasciitilde}$, Ö9 $\text{\textasciitilde}$.

SAMMANFATTNING

Instruktion	Kort beskrivning	Exempel
PRINT	Skriver på TV-skärmen. Kan även innehålla beräkningsuttryck.	10 PRINT "HEJ", A 20 PRINT A, A% 10 PRINT (A + B) * 2
LET	Tilldelar en variabel ett värde. LET behöver ej utskrivas i ABC 80 BASIC.	20 LET A = A * 10 30 LET A% = 5 40 LET A = "HEJ" 40 A = "HEJ"
INPUT	Väntar på inmatning av data från tangentbordet.	20 INPUT A 30 INPUT A% 40 INPUT A
READ	Läser variabelvärden från DATA-satser.	10 READ A, A%, A
DATA	Lagrar variabelvärden som skall läsas av en READ-sats.	20 DATA 5.31, 4, HEJ 30 DATA "BO LÖÖV"
IF...THEN...	Testar om ett logiskt uttryck är sant och utför i så fall satsen efter THEN.	40 IF A = 5 THEN GOTO 10 50 IF A = "HEJ" THEN GOTO 10
Funktion	Kort beskrivning	Exempel
INT(X)	Ger det största heltal som är mindre än eller lika med x.	10 LET A = INT(6.3)
Kommando	Kort beskrivning	Exempel
ED	Ger möjlighet till ändring i en programrad utan att den behöver skrivas om.	ED 40

ÖVNINGSEXEMPEL

4.1 Skriv ett BASIC-program som utför följande beräkningar

- a) $(2+5)^3$
- b) $(2+5)/7$
- c) $((a+b)ab)/a$ efter att a tilldelats värdet 2 och b värdet 3.

4.2 Skriv ett program som med instruktionen READ läser in talen 2, 4, 7, 11, 16 från en DATA-sats och skriver ut talens summa, föregånget av texten "SUMMAN ÄR".

4.3 Vad skrivs ut på TV-skärmen efter körning av nedanstående program?

```
10 LET A = 5
20 LET B = A*2 - 8
30 LET A = B + 2
40 PRINT A, B
```

4.4 Följande program beräknar medelåldern i en familj som består av fyra personer. Familjemedlemmarna är pappa Lars 36 år, mamma Anna 38 år, samt barnen Per 4 år och Eva 6 år.

```
10 READ L, A, P, E
20 PRINT "MEDELÅLDERN="; (L + A + P + E)/4; " ÅR"
30 DATA 36, 38, 4, 6
```

Ändra programmet så att antalet variabler och motsvarande data gäller för dina familjeförhållanden.

4.5 Motortidningen "FART OCH FLÄKT" har anordnat en stor test av bilar, där bland annat bensinförbrukningen skall beräknas. För varje bil finns uppgifter om hur många mil (M) som körts på ett visst antal liter bensin (B). Bensinförbrukningen uttryckt i liter per mil kan då beräknas som B dividerat med M. Skriv ett BASIC-program som via INPUT-satser hämtar data till variablerna M och B och sedan beräknar och skriver ut bensinförbrukningen i liter per mil.

- 4.6 Följande program skriver på TV-skärmen en "multiplikationstabell" för ett godtyckligt tal mellan 1 och 12.

```
10 PRINT "FÖR VILKET TAL";  
20 INPUT X  
30 PRINT  
40 PRINT "MULTIPLIKATIONSPROGRAM"  
50 LET A = A + 1  
60 PRINT X; " *"; A; " = ";  
70 IF A = 12 THEN GO TO 90  
80 GO TO 50  
90 PRINT
```

Komplettera programmet med lämpligt beräkningsuttryck på rad 60.

- 4.7 Modifiera programmet i uppgift 4.4 så att inmatning av familjemedlem-
marnas ålder sker via INPUT-satser på lämpligt sätt.

KAPITEL 5

Datorn fattar beslut

ABC80

SLUMPTAL

I vissa sammanhang är det önskvärt att ha tillgång till ett godtyckligt tal, ett slumptal. Med funktionen RND kan ABC 80 generera ett sådant tal. Kör följande program så förstår du hur RND fungerar.

```
10 FOR X = 1 TO 60
20 PRINT RND,
30 NEXT X
```

Utskriften blir

```
.079112      .106671      .452764
.352572      .832773      .299984
o s v
```

Gemensamt för alla slumpstal är tydligen att de har värden mellan noll och ett. Önskar man ett slumpstal mellan noll och hundra, ändras programrad 20 till

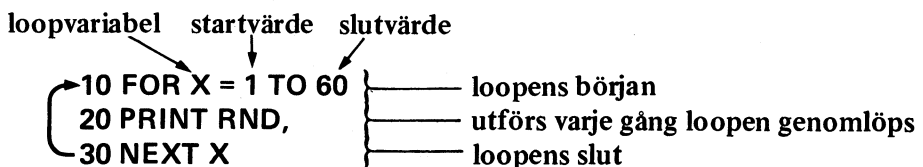
```
20 PRINT RND * 100,
```

Trots att instruktionen RND förekommer bara en gång i programmet, skrivs 60 slumpstal på TV-skärmen. Det beror på att programraderna 10 och 30 ger datorn instruktion om att mellanliggande programavsnitt ska upprepas 60 gånger.

PROGRAMLOOPAR

Ett programavsnitt som förekommer en gång i programmet, men utförs flera gånger, kallas loop eller slinga. Programraden 10 FOR X = 1 TO 60 anger loopens början. Variabeln X används här som s k loopvariabel. Efter varje varv i loopen ökas loopvariabelns värde med ett. Loopen genomlöps på detta sätt tills loopvariabelns slutvärde uppnåtts.

Loopens slut markeras på raden 30 med instruktionen NEXT X. Den sista instruktionen i loopen innehåller alltid loopvariabeln.



ALLTID SAMMA SLUMPTAL?

För att program som innehåller funktionen RND ska kunna kontrolleras, producerar datorn normalt alltid samma uppsättning slumpstal. Studera denna effekt genom att köra följande program några gånger.

```
10 PRINT RND, RND, RND
```

När man testat sitt program vill man oftast att olika slumpstal ska genereras vid varje körning. Detta åstadkommes med instruktionen RANDOMIZE. Tillfoga raden.

```
5 RANDOMIZE
```

Uppenbarligen bildas nu olika slumpstal när programmet körs flera gånger.

DATORN TÄNKER PÅ ETT TAL

Vi ska nu utnyttja datorns förmåga att åstadkomma slumpstal i en enkel gissningslek. Datorn programmeras att generera ett slumpstal, ett heltal mellan 0 och 9. Detta tal är tills vidare en hemlighet, väl bevarad i datorns minne ända till dess att du slutligen gissar det rätta talet. Här följer program för detta.

```
10 PRINT  
20 PRINT "GISSA ETT TAL MELLAN 0 OCH 9"  
30 RANDOMIZE  
40 LET S = INT (RND * 10)  
50 PRINT "DIN GISSNING";  
60 INPUT G  
70 IF G = S THEN GOTO 110  
80 PRINT "DET VAR INTE DET RÄTTA TALET"  
90 PRINT  
100 GOTO 50  
110 PRINT "RÄTT GISSAT"
```

Även det här programmet innehåller en loop. Programraderna 50 t o m 100 utförs här till dess att gissningen överensstämmer med värdet på variabelns tal. Observera att metoden med loopvariabel inte kan användas här. Det är ju omöjligt att förutse hur många gissningar som kan komma att behövas för att hitta rätt tal. När din gissning överensstämmer med datorns slumpstal lämnas programloopen. Detta sker i rad 70.

```
70 IF G = S THEN GOTO 110
```

Datorn fattar här ett beslut som grundas på om gissningen G är lika med slump-talet S. Om $G = S$ går datorn till programrad 110 och skriver RÄTT GISSAT. Om G och S är olika är villkoret inte uppfyllt. Datorn ska då fortsätta att utföra programraderna i stigande radnummerrföljd som vanligt. Raderna 80, 90 och 100 utförs då, varefter datorn går tillbaka till loopens början på rad 50.

FLERA VALMÖJLIGHETER

Eftersom du i föregående program bara har 10 möjliga tal att välja mellan, tar det inte så lång tid att gissa rätt.

Om däremot valet gäller alla heltal mellan 0 och 99, behöver du lite hjälp med gissandet. Låt t ex datorn ge besked om huruvida det föreslagna talet är för stort eller för litet. Med denna ledtråd är det alltid möjligt för den skicklige gissaren att klara uppgiften på mindre än 8 försök.

Ett förslag till program för detta är

```
10 PRINT "GESSA ETT SLUMPTAL MELLAN 0 OCH 99"
20 INPUT G
30 PRINT "DITT FÖRSLAG: "
40 PRINT "S = "
50 IF G > S THEN PRINT "FÖR STORT"
60 IF G < S THEN PRINT "FÖR LITET"
70 IF G = S THEN PRINT "RÄTT GISSAT"
80 GOTO 10
90 GOTO 10
100 GOTO 10
```

Prova programmet och försök hitta en metod att snabbt finna det rätta talet!

I detta program har datorn att välja mellan tre olika alternativ. Gissningen kan vara ett tal som är för stort, för litet eller lika med datorns slumptal. Alternativen är

1. $G = S$
2. $G > S$, som betyder G större än S
3. $G < S$, som betyder G mindre än S

Datorn känner igen alternativ 1 i programrad 80 och alternativ 2 i programrad 90. Eftersom det bara finns 3 möjliga alternativ, är det tillräckligt att datorn sorterar ut alternativen 1 och 2.

Låt oss undersöka vilka programrader som utförs i de olika alternativen.

Alternativ	Programrader som utförs
1. $G = S$	10 - 80 samt 160
2. $G > S$	10 - 90, 130 - 150, 60 o s v
3. $G < S$	10 - 120, 60 o s v

För att underlätta din analys av programmet, kommer här kommentarer till de olika programraderna.

Radnummer	Radens funktion
10, 30	Ger tomma rader på TV-skärmen.
20	Ger förklarande utskrift.
40	RANDOMIZE medför att olika slumpstal erhålls varje gång programmet körs.
50	Tilldelar variabeln S ett heltalsvärde mellan 0 och 99.
60	Ger förklarande utskrift.
70	Variabeln G tilldelas värde lika med din gissning.
80	Medför hopp till rad 160 om du gissat rätt.
90	Medför hopp till rad 130 om du gissat på ett för stort tal.
100	Om du varken gissat på rätt eller på för stort tal utförs denna rad, som ger utskrift "FÖR LITET".
110, 140	Ger tomma rader på TV-skärmen.
120, 150	Återhopp till rad 60 så att gissningen kan upprepas.
130	Ger utskrift om villkoret i rad 90 var uppfyllt.
160	Ger utskrift om villkoret i rad 80 var uppfyllt.

DATORN SORTERAR TAL

Att sortera tal efter en given beskrivning är en mycket vanlig och dessutom enkel uppgift för en dator. Man utnyttjar då att datorn kan avgöra vilket tal

som är minst av två givna tal. Datorn kan också enkelt låta talen byta plats. Låt oss först titta närmare på hur datorn kan byta plats på två variabelvärden. Antag att variabeln A tilldelats värdet 7 och variabeln B värdet 5. Hur ska ett program se ut, som byter plats på variabelvärdena? Skulle dessa programrader fungera?

```
10 LET A = B
20 LET B = A
```

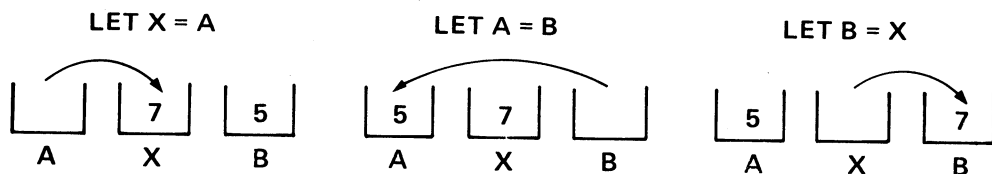
Varför inte? Jo, när datorn tilldelar A det nya värdet 5, så går det tidigare värdet på A förlorat och kan inte användas i nästa rad. I rad 20 tilldelas således B det nya värdet på A, d v s både A och B har efter körning av programmet värdet 5. En lösning av problemet är att temporärt lagra värdet på variabeln A i form av en hjälpvariabel X. Detta sker genom tillägg av programrad 5.

```
5 LET X = A
10 LET A = B
20 LET B = X
```

Prova genom att köra nedanstående program.

```
1 LET A = 7
2 LET B = 5
5 LET X = A
10 LET A = B
20 LET B = X
30 PRINT "A="; A, "B="; B
```

Det här programmet fungerar! Efter körning blir A = 5 och B = 7, d v s variabelvärdena har bytt plats.



Användning av hjälpvariabel

En rutin som sorterar tal i stigande ordning kan nu formuleras. Man utnyttjar datorns egenskap att kunna avgöra vilket av två tal som är störst och beroende på resultatet av jämförelsen låter man, om så erfordras, datorn byta plats på talen. Vi demonstrerar tekniken med ett exempel.

Låt datorn via en INPUT-sats hämta in värden på variablerna A, B och C. Dessa värden skall sedan sorteras och skrivas ut i stigande följd. Ett program som utför detta kan se ut så här.

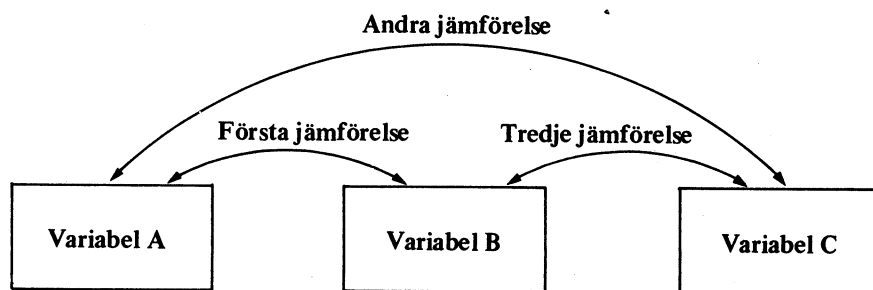
```
10 PRINT "MATA IN TRE TAL"
20 INPUT A, B, C
30 IF A<B THEN 70
40 LET X = A
50 LET A = B
60 LET B = X
70 IF A<C THEN 110
80 LET X = A
90 LET A = C
100 LET C = X
110 IF B<C THEN 150
120 LET X = B
130 LET B = C
140 LET C = X
150 PRINT "STIGANDE ORDNING"
160 PRINT A, B, C
```

Prova programmet och försök att fundera ut principen.

Som du säkert ser använder vi oss av den tidigare genomgångna metoden för att byta värden mellan två variabler. Genom upprepade jämförelser och byten ser vi till att variabeln A tilldelas det lägsta av de inmatade värdena. När därefter värdena på B och C jämförts och eventuellt bytt plats, kan vi vara säkra på att A tilldelats det lägsta och C det högsta värdet.

Radnummer	Radens funktion
10	Förklarande text skrivs.
20	Inmatning av värden på variablerna A, B och C.
30	Testar om värdet på variabeln A redan är mindre än värdet på variabeln B. I så fall står redan A och B i rätt ordning och inget byte skall ske. Instruktionerna på raderna 40, 50 och 60 överhoppas därför.
40, 50, 60	Byter inbördes värden mellan variablerna A och B.
70	Motsvarar rad 30 men jämför A och C.
80, 90, 100	Byter inbördes värden mellan variablerna A och C.

- 110 Motsvarar rad 30 men jämför B och C.
- 120, 130, 140 Byter inbördes värden mellan variablerna B och C.
- 150 Förklarande text skrivs.
- 160 De sorterade talen skrivs på TV-skärmen i stigande ordning.



Variabelsortering

För att förstå hur byten av variabler sker i ovanstående sorteringsprogram, kan du komplettera nedanstående tabell med de aktuella variabelvärden som gäller efterhand som raderna exekveras.

Radnummer	Värde på variabel efter att aktuell rad utförts			
	A	B	C	X
30	13	7	4	0
40	13	7	4	13
50	7	7	4	13
60	7	13	4	13
70				
80				
90				
100				
110				
120				
130				
140	4	7	13	

INDEXERADE VARIABLER

De flyttalsvariabler vi hittills har använt har betecknats med en bokstav, alternativt med en bokstav och en siffra. Heltalsvariabler och strängvariabler betecknades på motsvarande sätt med tillägg av procenttecken (%) resp. strängtecken (X).

Ibland har man behov av flera kombinationer än vad som kan åstadkommas med normala variabelnamn. Därför kan alla variabler förses med index, så att ett godtyckligt antal nya variabler skapas. Index utgörs av ett tal, skrivet inom parentes efter variabeln. Variabelnamnet A kan således användas för att skapa t ex fem nya variabler genom att index tillfogas:

A(1) , A(2) , A(3) , A(4) , A(5)

En grupp sådana indexerade variabler kallas lista eller vektor.

En annan fördel med användandet av indexerade variabler är att inmatningen av variabelvärden lätt kan göras i en loop. Dessutom underlättas sortering, som vi ska se på nästa sida. Låt oss som exempel se hur vår lista A kan tilldelas värden.

```
10 FOR N = 1 TO 5
20 INPUT A(N)
30 NEXT N
40 FOR N = 1 TO 5
50 PRINT A(N)
60 NEXT N
```

Detta program är en loop som genomlöps 5 gånger och tilldelar variablerna A(1), A(2), A(3), A(4) och A(5) värden. Därefter skrivs, med hjälp av en ny loop, värdena ut på TV-skärmen. Prova!

Du kan utan vidare använda index från 0 till 10, men vid större index än 10 måste ABC 80 först instrueras att reservera plats i minnet för alla dessa variabler. Detta sker med BASIC-instruktionen DIM (DIM betyder "dimensionera"), t ex

10 DIM A(100)

20 DIM A%(50)

Funktion

Reserverar plats för 101 element (flyttal) i en lista, d v s index t o m 100 får användas.

Reserverar plats för 51 element (heltal) med index t o m 50.

30 DIM A(25)

Reserverar plats för 26 element
(strängar) med index t o m 25.

LISTSORTERING

Vi skall nu se hur indexerade variabler används för sortering av godtyckligt stora listor. Du kommer väl ihåg tekniken vi använde för att sortera tre variabelvärden i rätt ordning med hjälp av parvisa jämförelser och ev byten?

Om vi istället låter fyra variabler bilda en lista med elementen A(1), A(2), A(3), A(4) kan vi tillämpa samma princip vid sorteringen men göra programmet kortare genom att utnyttja tekniken med loopar. Vi låter variablerna i fältet A ha värden som framgår av nedanstående lista.

Variabel	A(1)	A(2)	A(3)	A(4)
Värde	13	11	16	7

Liksom tidigare ska variabelvärdena sorteras i stigande storleksordning. Detta åstadkommer vi genom att successivt jämföra variablerna parvis och vid behov låta dem byta plats, så att det största av värdena kommer till höger i listan. Jämförelserna sker som tidigare i följande ordning:

Jämförelse mellan

Efter jämförelser och eventuellt byte

A(1) och A(2)

A(1)	A(2)	A(3)	A(4)
11	13	16	7

↙ byte ↘

A(1) och A(3)

A(1)	A(2)	A(3)	A(4)
11	13	16	7

↙ ej byte ↘

A(1) och A(4)

A(1)	A(2)	A(3)	A(4)
7	13	16	11

↙ byte ↘

A(2) och A(3)

A(1)	A(2)	A(3)	A(4)
7	13	16	11

↙ ej byte ↘

A(2) och A(4)

A(1)	A(2)	A(3)	A(4)
7	11	16	13

↙ byte ↘

A(3) och A(4)

A(1)	A(2)	A(3)	A(4)
7	11	13	16

↙ byte ↘

Detta kan allmänt skrivas som att A(I) jämförs med A(J), där I antar värden från 1 till 3. För varje värde på I antar J alla värden mellan I + 1 och 4.

Ett BASIC-program, som utför denna listsortering, kan se ut så här.

```
40 FOR I = 1 TO 3
50 FOR J = I + 1 TO 4
60 IF A(I) < A(J) THEN 100
70 LET X = A(I)
80 LET A(I) = A(J)
90 LET A(J) = X
100 NEXT J
110 NEXT I
```

I rad 60 testas om variabelvärdena behöver bytas. Om så inte är fallet går programmet till rad 100 som medför att två nya variabler jämförs. Skall däremot byte av variabelvärdena ske, så utförs detta med raderna 70, 80, och 90. Du kommer väl ihåg varför vi måste utnyttja hjälpvariabeln X vid bytet?

Genom att programmet består av två loopar så varierar I från 1 till 3 och för varje värde på I varierar J från I + 1 till 4.

För att kunna provköra programmet behöver du bara tillfoga rader för inmatning av värden till variablerna, samt rader för utskrift av den sorterade listan.

```

10 FOR I = 1 TO 4
20 INPUT A(I)
30 NEXT I
120 PRINT "SORTERAD LISTA"
130 FOR I = 1 TO 4
140 PRINT A(I);
150 NEXT I

```

En körning av programmet med värden enligt listan på sid 52 ger följande utskrift.

RUN

?13

?11

?16

?7

SORTERAD LISTA

7 11 13 16

ABC80

Hur skall programmet ändras för att datorn ska kunna sortera t ex 7 godtyckliga tal istället för 4? Enkelt, du ändrar bara slutvärdet för loopvariablerna! På rad 10, 50 och 130 ändras från 4 till 7 och på rad 40 till 6.

Detta kan generaliseras. Vill du sortera N stycken tal, ändrar du på motsvarande sätt slutvärdena på raderna 10, 50 och 130 till N och på rad 40 till N - 1.

Kom bara ihåg, att om du vill använda listor med index större än 10, måste DIM-satser tillfogas. Om t ex raden

5 DIM A(100)

tillfogas i sorteringsprogrammet, kan index upp t o m 100 användas.

DATORN SORTERAR TÄVLINGSRESULTAT

En arrangör av alpina skidtävlingar vill ha ett datorprogram, som direkt efter varje åkning sorterar in resultatet i resultatlistan och skriver ut denna på TV-

skärmen. 10 stycken utförsåkare deltar i tävlingen. Programmet, som utnyttjar de principer för sortering av tal som vi just har tittat på, kan se ut så här.

```

10 LET N = N + 1
20 INPUT A(N), A$(N)
30 FOR I = 1 TO N - 1
40 FOR J = I + 1 TO N
50 IF A(I) < A(J) THEN 120
60 X = A(I)
70 A(I) = A(J)
80 A(J) = X
90 X$ = A$(I)
100 A$(I) = A$(J)
110 A$(J) = X$
120 NEXT J
130 NEXT I
140 PRINT CHR$(12)
150 PRINT "RESULTATLISTA T O M ÅKARE"; N
160 PRINT
170 FOR I = 1 TO N
180 PRINT A(I), A(I)
190 NEXT I
200 IF N < 10 THEN GOTO 10
210 PRINT "ÅKET KLART"

```

Radnummer	Radens funktion
10	Variabeln N ökas med ett för varje sortering, d v s håller reda på hur många åkare som kört.
20	Variabeln A(N) tilldelas den aktuella åkarens tid och strängvariabeln A\$(N) åkarens namn.
40 - 130	Dubbelloop för sortering. Som synes måste vi här byta värden både i tabellen A och i tabellen A\$. I övrigt samma sorteringsteknik som tidigare. Som du ser har vi i raderna 60 till 110 utelämnat instruktionen LET. Det är alltid tillåtet att utelämna LET i tilldelningssatser hos ABC 80.
140	CHR\$(12), som kommer efter PRINT, betyder att ABC 80 raderar skärmen och börjar nästkommande utskrift längst upp till vänster på TV-skärmen.

150	Ger förklarande text på TV-skärmen.
160	Ger ett mellanrum på skärmen.
170 - 180	Loop för utskrift av sorterad resultatlista.

Prova programmet genom att successivt mata in följande tider och namn. Mata först in tiden, följd av kommatecken, sedan namnet inom citationstecken.

Tid	Namn
100.34	STENSTARK
109.41	TÖNTI
100.92	GRÖSS
102.31	HAJDHÄGGER
100.25	PIL NER
102.31	PIL MER
101.19	NÖJRÖJTER
106.37	HANTARSEJNER
102.35	BAKOBSSON
105.83	BJÄLLER

SAMMANFATTNING

En indexerad variabel med ett enda index kallas lista eller vektor. En variabel kan också ha två index och kallas då tabell eller matris. Ett gemensamt namn för indexerade variabler är fält. Vi har hittills endast behandlat fält med ett index och dessa kallas ibland endimensionella fält. Tvådimensionella fält, d v s matriser, behandlas i kapitel 14. Som index i indexerade variabler får användas talen 0 t o m 10, om du inte reserverar större plats med en DIM-sats.

I IF-satser används s k logiska uttryck. Ett logiskt uttryck kan vara sant eller falskt. Om det logiska uttrycket är sant, så utförs den instruktion som kommer efter THEN. Direkt efter THEN får en godtycklig BASIC-sats skrivas. Om BASIC-satsen är en GOTO-sats kan instruktionen GOTO utelämnas och enbart radnummer anges. Om uttrycket är falskt utförs efterföljande programrad.

Logisk operator	Kort beskrivning	Exempel
>	Större än. Minnesregel: stor öppning mot stort tal.	10 IF A > B THEN GOTO 50
<	Mindre än.	20 IF A < B THEN 50
=	Lika med.	30 IF A = B THEN PRINT A
>=	Större än eller lika med.	40 IF A >= B THEN 70
<=	Mindre än eller lika med.	50 IF A <= B THEN 10
<>	Ej lika med. (Skilt från.)	60 IF A <> B THEN GOTO 10

Instruktion	Kort beskrivning	Exempel
RANDOMIZE	Ger RND-funktionen ett slumpmässigt startvärde.	10 RANDOMIZE
FOR...TO...	Utför ett programavsnitt ett antal gånger som bestäms av loopvariabelns start- och slutvärde.	20 FOR X = 1 TO 5
NEXT	Avslutar ett FOR-avsnitt (loop).	50 NEXT X
LET	Tilldelar en variabel ett värde. Uttrycket kan även skrivas direkt utan LET.	30 LET A = 5 30 A = 5 40 LET A * 2 = B * 2 40 A * 2 = B * 2
DIM	Reserverar utrymme för en indexerad variabel (lista).	10 DIM A(50) 20 DIM A%(75) 30 DIM A * (30)

Funktion	Kort beskrivning	Exempel
RND	Ger ett slumptal från och med 0 till och med 0.999999.	10 LET A = RND 20 LET B = RND * 10

Specialtecken**CHR ♂ (12)****Kort beskrivning**

Tömmer bildskärmen och flyttar markören till övre vänstra hörnet.

Exempel**10 PRINT CHR ♂ (12)****ÖVNINGSEXEMPEL**

- 5.1 För en skolelev i första klassen skulle uppgiften att skriva följande mening 20 gånger vara nästan omöjlig att utföra.

MOR ROR OCH FAR ÄR EN ORM

Skriv ett datorprogram som utför denna uppgift.

- 5.2 Vid julfesten hos idrottsföreningen IFM hade sålts 100 lotter med nummer 0 till och med 99. 10 vinster skulle lottas ut. Skriv ett program som verkställer dragningen och skriver en vinstlista.

VINSTLISTA**VINST 1****NUMMER...****VINST 2****NUMMER...****O S V**

- 5.3 Ändra programmet i uppgift 5.2 så att det kan användas på nyårsfesten, då man sålde 200 lotter med numren 100 - 299 och ville lotta ut 15 vinster.
- 5.4 Skriv ett program, där datorn simulerar funktionen hos en vanlig tärning och skriver ut

TÄRNINGEN STANNAR PÅ...

- 5.5 Skriv ett program som tränar förmågan att i huvudet beräkna summan av två godtyckliga tal mindre än 100. På TV-skärmen skall skrivas två slumpmässiga tal mellan 0 och 99. Svaret matas in via en INPUT-sats och kontrolleras i datorn. Datorn skriver sedan på TV-skärmen om svaret var rätt eller fel. Om svaret var fel, ska samma tal presenteras igen för ett nytt försök.
- 5.6 10 matematikprofessorer fick köra programmet i uppgift 5.5. Var och en av professorerna körde programmet 50 gånger. I nedanstående lista framgår i hur många fall var och en av dem svarade rätt vid första försöket.

Professor nr	Antal rätt vid första försöket
1	26
2	41
3	29
4	12
5	33
6	48
7	42
8	19
9	13
10	38

Mata in ovanstående resultat i ett endimensionellt fält och beräkna medelvärdet av de tio deltagarnas antal rätt vid första försöket.

Medelvärdet får du genom att summera de olika deltagarnas resultat och dividera med antalet. Låt datorn skriva ut en lista med deltagarnas resultat och det beräknade medelvärdet.

- 5.7 Modifiera gissningsprogrammet på sid 46 så att utskrift erhålls på TV-skärmen av hur många försök som erfordrats för rätt gissning.

- 5.8 Nedanstående program skriver ut en stryktipsrad.

```
10 PRINT CHR$(12)
20 RANDOMIZE
30 PRINT "TIPSFÖRSLAG"
40 PRINT
50 FOR X = 1 TO 13
60 PRINT "MATCH NR"; X,
70 T = RND
80 IF T < .6 THEN 130
90 IF T < .8 THEN 150
100 PRINT "      2"
110 NEXT X
120 GOTO 170
130 PRINT "1"
140 GOTO 110
150 PRINT "  X"
160 GOTO 110
170 PRINT
180 PRINT "LYCKA TILL"
```

Vilken är sannolikheten för ettor, kryss respektive tvåor? Ändra programmet så att sannolikheten för etta blir 50%, för kryss 25% och för tvåa 25%.

- 5.9 Vid ett uttagnings-test för blivande agenter vid Grönköpings Agenskola (GASK) erhöles följande värden på de sökandes intelligenskvot (IQ).

Sökande nr	IQ
001	36
002	41
003	59
004	110
005	74
006	107
007	130

Skriv ett program som sorterar de sökande så att den med högst IQ placeras överst på listan och utnämns till agent. Resten sorteras efter stigande IQ.

KAPITEL 6

**Lagra program
på kassett**

ABC80

SNABB INMATNING AV PROGRAM

ABC 80 är en mycket lättskött dator, men det är trots allt ganska tidsödande att skriva in långa BASIC-program via tangentbordet. Med hjälp av en kassettbandspelare kan dock programmeringen rationaliseras högst avsevärt. När ett program är klart, kan det sparas på kassettband genom att en till datorn ansluten kassettbandspelare startas, varefter du ger datorn kommando att lagra programmet på bandet. Om du vid ett senare tillfälle vill använda samma program är det bara att placera kassetten i bandspelaren, starta den och ge datorn kommando att hämta in önskat program.

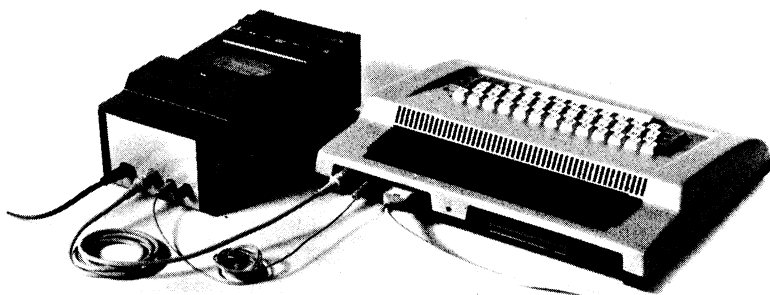
Programmet överförs mycket snabbt mellan dator och kassett, hastigheten är ca 70 tecken per sekund. En snabb maskinskrivare klarar bara en tiondel av det tempot och en genomsnittlig ABC 80-användare är förmodligen ännu långsammare när det gäller att skriva in program manuellt via tangentbordet. Tidsvinsten med program på kassett är alltså stor. Vi ska nu se hur man praktiskt går till väga.

INKOPPLING AV BANDSPELAREN

Till ABC 80-systemet hör en kassettbandspelare, som är speciellt utrustad för att kunna startas och stoppas med signaler från datorn. Oberoende av datorns styrsignaler kan bandet snabbspolas på vanligt sätt.

När du söker efter ett program på kassetten är räkneverket bästa hjälpmedlet. Du kan också mycket lätt hitta ett program genom att lyssna på inspelningen. Snabbspolning tillåts under pågående avspelning utan att bandet först stoppas.

Eftersom motorstyrningen tillkommer jämfört med en vanlig bandspelare måste två kablar användas för att koppla samman tangentbordet med bandspelaren. Sammankopplingen utförs så som bilden visar.



Sammankoppling av bandspelare och tangentbord

BAND

De kassetter som ska användas för programlagring måste vara av mycket god kvalitet för att förhindra s k "drop out". "Drop out" innebär att tecken tappas bort p g a ojämnheter i bandets oxidskikt. Detta kan givetvis få förödande konsekvenser och man bör därför välja bra kassetter av kända fabrikat. Var försiktig med dina programkassetter så att de inte utsätts för smuts, höga temperaturer eller rent mekaniska skador.

LAGRING AV PROGRAM

Förbered ett band genom att mata fram bandet till dess att hela den inledande startremsan passerat förbi öppningen i kassetten. Det kan du göra genom att vrida en blyertspenna runt i centrum av spolen. Eftersom ABC 80 påbörjar inspelning efter en fördröjning på ungefär 5 sekunder, kan det i annat fall uppstå ett fel om bandet har en lång startremsa.

Anslut bandspelaren till tangentbordet och till vägguttaget, placera bandet på dess plats i bandspelaren och nollställ räkneverket. Vi ska nu visa hur du kan använda bandspelaren. Som exempel väljer vi några korta program på bara en rad vardera.

Mata in första programmet i datorn via tangentbordet.

```
10 PRINT "UPPG. ENL. PROGRAM1 ÄR UTFÖRD"
```

Förbered bandspelaren för inspelning genom att trycka ned tangenterna REC och PLAY. Lagra programmet på kassett med kommandot SAVE.

Skriv t ex

```
SAVE PROGRAM1
```

Glöm inte att avsluta kommandot genom att trycka på tangenten RETURN. Datorn startar nu bandspelaren. När TV-skärmen efter ungefär 13 sekunder visar **ABC 80** är programmet inspelat.

Radera rad 10 och mata in nästa program i datorn via tangentbordet.

```
20 PRINT "JAG HAR NU KÖRT PROGRAM2"
```

Lagra programmet på kassett genom att skriva

```
SAVE ÖVNING 2
```

De namn som vi valt att ge programmen är PROGRAM1 resp ÖVNING 2. Det största antal tecken som får ingå i ett programnamn är 12. Om programnamnet är längre än 8 tecken, måste det nionde tecknet vara en punkt.

Spola nu tillbaka bandet till början och tryck ned tangenten PLAY på bandspelaren.

Kör det första programmet med kommandot

RUN PROGRAM1

Efter en halv minut har datorn hämtat och kört programmet. Övertyga dig om att programmet finns i datorns minne genom att använda LIST på vanligt sätt.

Nästa program, ÖVNING 2, kan nu köras utan föregående returspolning. Skriv

RUN ÖVNING 2

När detta program överfördes från bandet till datorns minne raderades samtidigt PROGRAM1. Du kan lätt se att så är fallet genom att använda kommandot LIST.

Det är inte nödvändigt att känna till programnamnet för att kunna överföra programmet från band till datorns minne.

Spola tillbaka bandet till början igen och prova kommandot

RUN CAS:

Datorn startar då bandspelaren, hämtar programmet som ligger närmast på kassetten och skriver ut

FOUND PROGRAM1.BAC

Sedan kör datorn programmet.

Förutom de 8 positionerna i namnet PROGRAM1 finns enligt ovan ytterligare 4 tillgängliga positioner för namngivning av program. Om dessa inte utnyttjas tillfogar datorn automatiskt .BAC till det givna programnamnet.

Om du inte önskar köra programmet direkt utan bara vill överföra det till datorn, kan RUN bytas ut mot LOAD i exemplen ovan. Spola tillbaka bandet och skriv

LOAD CAS:

Du kan då med LIST undersöka programmet innan du kör. Programkörning startas sedan som vanligt med RUN.

Ibland vill man tillfoga en programdel från kassettband utan att radera det program som man har i datorns minne. Kommandot för detta, om det program som ligger i tur på kassetten ska tillfogas, är

MERGE CAS:

Om datorn själv ska söka upp ett speciellt programnamn och tillfoga det till det program, som redan ligger i datorns minne, använder du kommandot MERGE, följt av det aktuella programnamnet.

Med de metoder som hittills har beskrivits, har du själv med ett kommando beordrat datorn att hämta ett program från kassettbandet. Man kan också låta datorn sköta programbyte helt på egen hand, genom att i programmet skriva en instruktion för detta.

Lagra följande två program på kassettband. Kalla det första PROG1 och det andra PROG2.

```
10 PRINT "PROGRAM 1 ÄR UTFÖRT"  
20 CHAIN "PROG2"
```

Lagra programmet med SAVE PROG1.

```
10 PRINT "PROGRAM 2 ÄR UTFÖRT"
```

Lagra programmet med SAVE PROG2. Spola sedan tillbaka bandet och skriv

RUN CAS:

Båda programmen kommer då att köras i tur och ordning. När instruktionen CHAIN "PROG2" utförs i det första programmet, startar datorn bandspelaren och överför programmet PROG2 från kassett till datorns minne. På detta sätt kan datorn beordras att köra oerhört långa program, som annars inte får plats i den.

FLEXSKIVA SOM LAGRINGSMEDIUM

För att lagra program kan du till ABC 80 också använda en separat flexskiveenhet. Fördelen med flexskiva blir mest märkbar när stora program eller stora datamängder utnyttjas. Ett mycket stort program, som upptar hela minnet i ABC 80, tar c:a 10 minuter att lagra på kassett. Motsvarande tid för överföring av programmet till flexskiva är bara några sekunder.

SAMMANFATTNING

Instruktion	Kort beskrivning	Exempel
CHAIN "..."	Hämtar angivet program från kassett.	100 CHAIN "PROG2"
Kommando	Kort beskrivning	Exempel
SAVE	Lagrar ett namngivet program på kassett.	SAVE PROG1
RUN ...	Hämtar program med angivet namn från kassetten och kör programmet.	RUN PROG1
RUN CAS:	Hämtar närmaste program från kassetten och kör programmet.	RUN CAS:
LOAD ...	Överför program med angivet namn från kassett till minnet.	LOAD PROG2
LOAD CAS:	Överför närmaste program från kassetten till minnet.	LOAD CAS:
MERGE ...	Hämtar angivet program från kassett utan att radera det program som finns i minnet. Om samma radnummer finns både på kassett och i minnet kommer raden från kassetten att användas.	MERGE PROGRAM3
MERGE CAS:	Hämtar närmaste program från kassett utan att radera det program som finns i minnet. Om samma radnummer finns både på kassett och i minnet kommer raden från kassetten att användas.	MERGE CAS:

KAPITEL 7

**Flödesschema
underlättar
programmeringen**

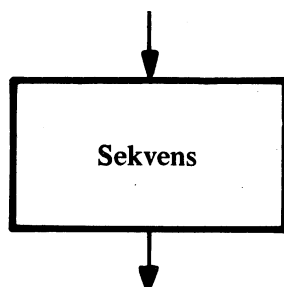
ABC80

FLÖDESSCHEMA SOM HJÄLPMEDEL

Det har varit relativt lätt att skriva program för de uppgifter som datorn hittills haft att utföra. Ett kort program är lätt att överblicka, men när programmets omfattning blir större än 20 - 30 rader kan det bli svårt att konstruera ett korrekt program. Hjälpmiddel för att lösa programmeringsuppgifter saknas inte. En av de bästa därvidlag är flödesschemat.

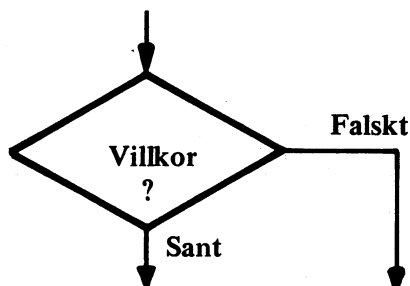
FLÖDESSCHEMATS GRUNDELEMENT

För konstruktion av ett flödesschema fordras bara två grundelement, sekvens-elementet och villkorselementet. Sekvenselementet åskådliggör att en sekvens ska utföras.



Sekvens-element.

Villkorselementet innebär, att datorn ska undersöka om det villkor som ställts är uppfyllt eller ej.



Villkorselement.

Villkorselementet har två utgångar. Datorn kommer att fortsätta programkörningen på olika programrader, beroende på om villkoret uppfyllts eller ej.

KONSTRUERA ETT FLÖDESSCHEMA

Låt oss nu beskriva en datoruppgift med flödesschema. Problemet består i att beräkna hur storleken av ett kapital ändras med ränta år för år.

Först skriver vi ned vad som ska utföras i rätt tidsföljd, punkt för punkt.

1. Kapitalets storlek, K , bestäms.
2. Räntesatsen, R , fixeras.
3. Antal år, $\bar{A}$, som beräkningen gäller anges.
4. Kapitalets ändring för ett år beräknas: $K = K + K \cdot R/100$.
5. Text för utskrift anges: KAP, RÄNTESATS, ANTAL ÅR.
6. De beräknade och givna talvärdena presenteras: K , R , $\bar{A}$.
7. Åter till punkt 4 om beräkningarna inte är slutförda.
8. Stopp.

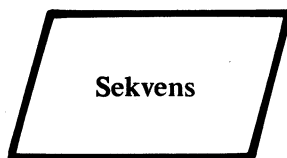
Uppgiften kommer att kunna lösas på detta sätt. Tyvärr kommer texten i tabellhuvudet, KAP RÄNTESATS ANTAL ÅR, att skrivas ut för varje nytt år som kapitalet beräknas. Punkterna 4 t o m 7 ska utföras många gånger och detta blir därför en loop i det färdiga programmet.

För att undvika att texten återkommer för varje ny beräkning ställer vi som punkt 4a ett villkor.

- 4a Om kapitalet för första året är beräknat ska punkt 5 utelämnas, d v s hoppas över.

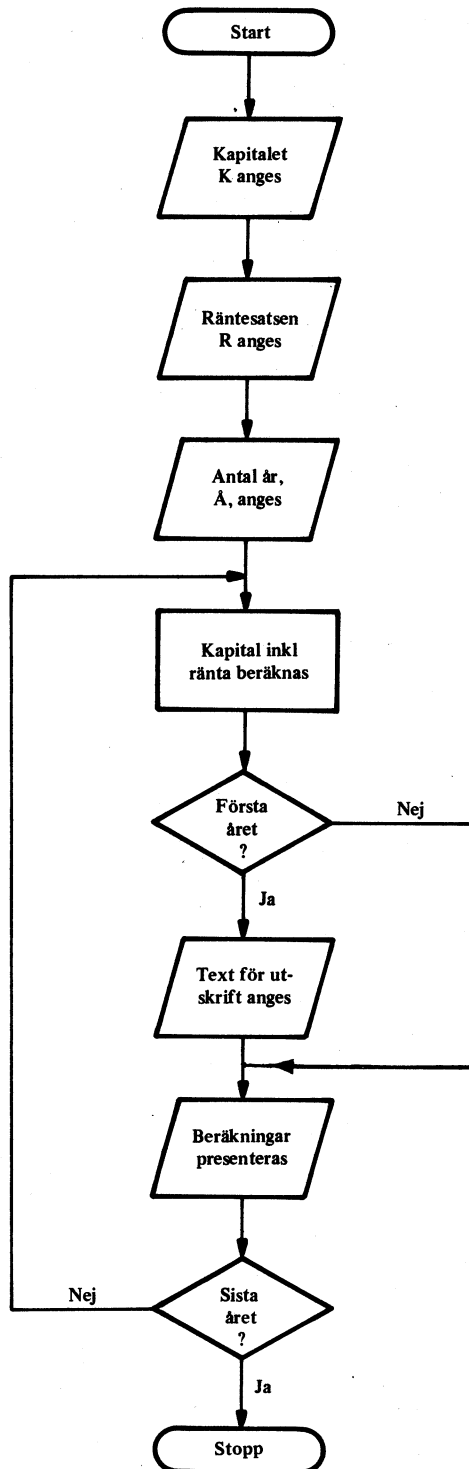
När de nio punkterna ovan sammanställts i ett flödesschema, får vi det resultat som visas på nästa sida. Symbolerna i flödesschemat ska sedan jämföras med motsvarande programrader i det färdiga programmet.

I detta schema förekommer bl a nedanstående symbol.



In- eller utmatning

Den används för en avgränsad uppgift där det förekommer en utmatning till TV-skärmen eller inmatning från tangentbordet. En sammanställning av flödesschemasympboler ges i slutet av kapitlet.



Varje symbol i flödesschemat ska nu översättas till ett programavsnitt. Steg för steg får vi

1. **10 PRINT "ANGE KAPITALET";**
20 INPUT K
2. **30 PRINT "ANGE RÄNTESATSEN";**
40 INPUT R
3. **50 PRINT "ANGE ANTAL ÅR BERÄKN. AVSER";**
60 INPUT Å
70 FOR T = 1 TO Å
4. **80 LET K = K + K * R / 100**
- 4a. **90 IF T > 1 THEN**
5. **100 PRINT**
110 PRINT "KAP", "RÄNTESATS", "ANTAL ÅR"
120 PRINT
6. **130 PRINT K, R, T**
7. **140 NEXT T**
8. **150 END**

↑
Här saknas ett radnummer. När hela flödesschemat är genomgången, fyller du i det rätta radnumret.

Det saknade radnumret på rad 90 är tydligen 130. Programrad 90 blir således

90 IF T > 1 THEN 130

Instruktionen END i rad 150 anger att programmet avslutas här. END har hittills inte använts och behöver oftast inte ingå i programmet. Vi återkommer till END senare i kapitlet.

Kör nu programmet med t ex kapitalet 900 kr och räntesatsen 7% för tiden 18 år. Resultatet blir 3041,97 kr.

PROGRAMKÖRNINGEN REGLERAS FRÅN TANGENTBORDET

När man vill beräkna kapitalets förändring med ränta på ränta för t ex 50 år, är programmet inte så användbart i sitt nuvarande skick. På grund av maskinens snabbhet presenteras alla 50 beräkningarna i rask följd, och kvar på skärmen blir bara de 21 senaste åren.

Om programmet i stället skrivs så att beräkningarna utförs i en oändlig loop, med ett tillfälligt stopp i loopen, kan presentationen av resultattabellen stegas fram år för år. Programkörningen kan hejdas med en INPUT-sats, men mera praktiskt är att i stället använda en GET-sats.

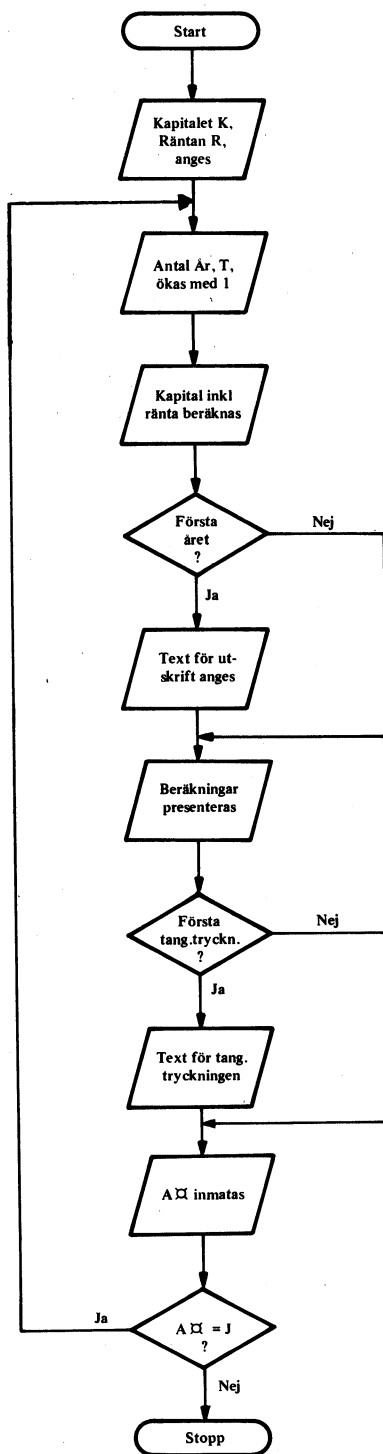
Raden

160 GET A

innebär att A tilldelas värdet av ett enda tecken. Detta tecken skrivs inte på TV-skärmen. Dessutom är GET-satsen speciell på så sätt att man inte behöver trycka på tangenten RETURN för att avsluta inmatningen. Datorn fortsätter ändå programkörningen omedelbart efter det att en tangent blivit nedtryckt.

Flödesschemat för det nya programmet visas på nästa sida. I det nya programmet kan de flesta programraderna från det föregående programmet bibehållas oförändrade.

```
10 PRINT "ANGE KAPITALET";
20 INPUT K
30 PRINT "ANGE RÄNTESATSEN";
40 INPUT R
50 } Obs! Raderna 50 och 60 ska avlägsnas.
60 }
70 LET T = T + 1
80 LET K = K + K * R/100
90 IF T > 1 THEN 130
100 PRINT
110 PRINT "KAP", "RÄNTESATS", "ANTAL ÅR"
120 PRINT
130 PRINT K, R, T
140 IF T > 1 THEN 160
150 PRINT "FÖR VARJE TRYCKNING PÅ TANG.J
FORTSÄTTER BERÄKNINGARNA FÖR YTTERLIGARE
ETT ÅR"
160 GET A
170 IF A = "J" THEN 70
```



Prova programmet genom att låta datorn beräkna vad banksaldot hade varit idag (1979) om en krona insattes år 1900 och räntan under alla åren hade varit 7%. Resultatet blir 209,57 kr (datorn ger 209,565 kr).

Om du håller tangenten J nedtryckt hela tiden går programkörningen något fortare. Som tidigare nämnts är ju alla tangenter repeterande.

FÖRDRÖJNING I PROGRAMMET

När datorn ska presentera en text som är större än vad TV-skärmen rymmer, kan man som i föregående exempel låta åskådaren trycka på en godtycklig tangent, när datorn ska presentera fortsättningen av texten. Ibland är det lämpligt att som alternativ låta datorn vänta en viss tid och sedan fortsätta programkörningen. Man kan lätt göra en sådan fördröjning i programmet genom att konstruera en loop, som genomlöps ett lämpligt antal gånger. Prova detta program.

```
10 FOR F = 1 TO 2000
20 NEXT F
30 PRINT "FÖRDRÖJNING"
```

Denna fördröjning tar ca 2 sekunder. Fördubbla tiden för fördröjningen genom att byta ut rad 10 enligt nedan och kör programmet igen.

```
10 FOR F = 1 TO 4000
```

FLERA SATSER PÅ SAMMA PROGRAMRAD

I ABC 80-BASIC finns möjlighet att skriva flera satser på samma rad, åtskilda av kolontecken. Programmet ovan får då efter omskrivning utseendet

```
10 FOR F = 1 TO 4000 : NEXT F
30 PRINT "FÖRDRÖJNING"
```

KORTARE PROGRAM MED SUBROUTINER

Låt oss nu skriva ett program som visar principen för hur man stegvis kan presentera text på TV-skärmen. Datorn får i uppgift att annonsera de filmer som ska visas på biograferna under veckan. Data om varje film ska visas i tre sekunder, därefter tillkommer nästa film o s v, ända till dess att informationen är

fullständig. Ett flödesschema som visar vilken text som ska visas på TV-skärmen återfinns på nästa sida. Sekvensen VÄNTA 3 s förekommer 4 ggr i flödesschemat. Trots det behöver denna sekvens bara skrivas en enda gång i programmet. Vid varje tillfälle som instruktionen GOSUB (eng go to subroutine) utförs, kommer datorn att utföra sekvensen VÄNTA 3 s, som kallas subrutin. Subrutinen, som återfinns på programraderna 170 - 190, innehåller förutom fördröjningen på 3 sekunder också PRINT-satser, som förtydligar utskriften. Raden

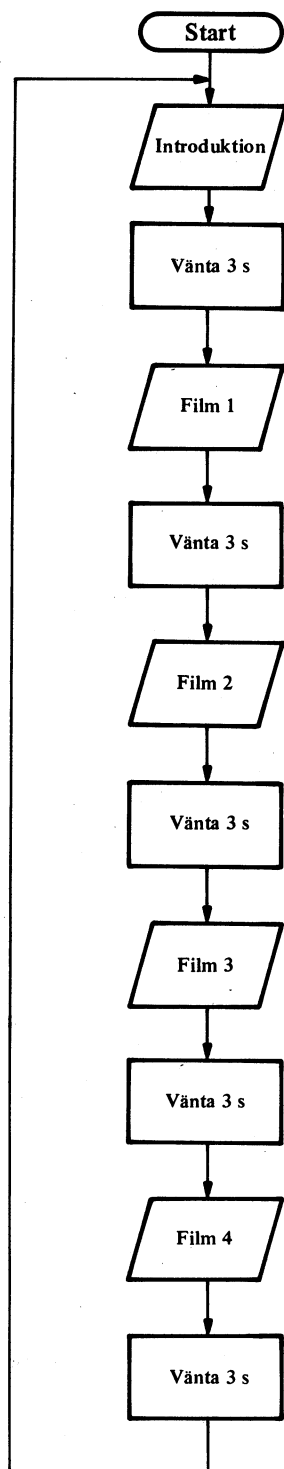
30 GOSUB 170

innebär att datorn ska utföra programavsnittet som börjar med rad 170. För att datorn ska få besked om var slutet på subrutinen finns, avslutas subrutinen med instruktionen RETURN.

Programmet blir

```
10 PRINT CHR$(12)
20 PRINT "FILMER DENNA VECKA:"
30 GOSUB 170
40 PRINT "HAR KNATTE OCH TJATTE FÅTT FNA
TT?"
50 PRINT "SÖNDAG 13 - 15, 15 - 17"
60 GOSUB 170
70 PRINT "KENT 007 SKJUTER ALLT FRAMFÖR
SIG"
80 PRINT "MÅNDAG 19 - 21, 21 - 23"
90 GOSUB 170
100 PRINT "ROCKING VRÅL MED PÅL"
110 PRINT "ONSDAG 19 - 21, 21 - 23"
120 GOSUB 170
130 PRINT "ROBBING HOOD BEATEN IN THE FO
REST"
140 PRINT "LÖRDAG 13 - 15"
150 GOSUB 170
160 GOTO 10
170 PRINT "=====
===="
180 PRINT
190 FOR F = 1 TO 3000 : NEXT F : RETURN
```

Kör programmet!



PROGRAMMET UTÖKAS

Eftersom biografägaren nu vill informera om att film nr 2 och film nr 3 är barnförbjudna tillfogas ytterligare en subrutin

200 PRINT "BARNFÖRBJUDEN" : RETURN

Subrutinen ska anropas från två platser i programmet. Tillfoga t ex

```
85 GOSUB 200
115 GOSUB 200
```

Provkör programmet och kontrollera att ordet "Barnförbjuden" kommer in på rätt plats.

EGNA KOMMENTARER I PROGRAMMET

Långa program kan ofta vara svåra att förstå i efterhand. Den som skriver egna program kan underlätta sitt arbete genom att föra in kommentarer i programmet med REM (eng remark). Raderna

```
165 REM SUBROUTIN FÖRDRÖJNING
195 REM SUBROUTIN BARNFÖRBJUDEN
```

kan tillfogas i programmet. Datorn utför ingenting på dessa programrader. Där-
emot finns raderna i datorns minne och skrivs ut som vanligt när kommandot
LIST används. Programmet är nu så stort, att det inte ryms på TV-skärmen.
Använd därför kommandot LIST - 100 för att titta på första delen av program-
met och LIST 100 - för andra.

SUBROUTINER PLACERAS UTANFÖR HUVUDPROGRAMMET

Den del av programmet, som inte innehåller några subrutiner, kallar vi huvud-
program. Huvudprogrammet omfattar i det här exemplet raderna 10 - 160 och
subrutinerna återfinns på raderna 170 - 200. Man brukar alltid, för att prog-
rammen ska bli lättöverskådliga, placera eventuella subrutiner sist. Om du vill
att programmet ska utföras bara en gång, skriver du en END-sats mellan huvud-
program och subrutiner, t ex

```
160 END
```

I ett program som inte innehåller någon subrutin, behövs ingen END-sats.
ABC 80 får ändå information om vilken instruktion som är den sista i prog-
rammet.

NUMRERA OM PROGRAMRADERNA

I det ursprungliga programmet fanns enbart radnummer med hela tiotal, 10, 20, 30 o s v. Om du önskar att också det färdiga programmet, efter många ändringar, ska ha radnummer med hela tiotal använder du kommandot REN (eng renumber).

Skriv

REN

Kontrollera med kommandona LIST -100 resp LIST 100 - vilken förändring som har skett med programmets radnummer. Observera att radnummer efter GOSUB också ändras, så att subrutinernas nya radnummer placeras efter varje GOSUB. Detta gäller på samma sätt satser som innehåller GOTO.

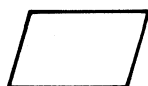
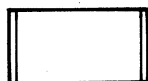
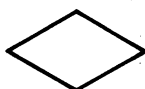
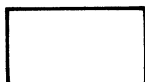
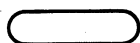
SAMMANFATTNING

Instruktion	Kort beskrivning	Exempel
GET	Läser ett tecken från tangentbordet till en strängvariabel utan att skriva tecknet på skärmen.	90 GET AØ
GOSUB	Hoppa till subrutin.	30 GOSUB 90
RETURN	Avslutar subrutin och medför återhopp till huvudprogrammet till raden efter motsvarande GOSUB.	120 RETURN
END	Avslutar programkörningen.	200 END
REM	Kommentar kan skrivas i programmet.	30 REM ** SUBR 1 **

Kommando	Kort beskrivning	Exempel
REN	Numrerar om alla raderna i programmet och ändrar satser med GOTO, GOSUB o s v. Följande varianter finns: Numrerar raderna med början på 10 och med intervall 10. Numrerar raderna med början på ett visst radnummer, t ex 25, och med intervall 25. Numrerar raderna med början på visst radnummer, t ex 500, och med ett önskat intervall, t ex 20.	 REN REN 25 REN 500, 20

FLÖDESSCHEMATS SYMBOLER

SYMBOL



BETYDELSE

Start eller stopp i program. Alternativ anges i symbolen.

Avgränsad uppgift som kort beskrivs i symbolen.

Exekveringen kan fortsätta på olika rader, beroende på utfallet av ett test som görs här. Testet beskrivs i symbolen.

Avgränsad mängd av uppgifter som ej specificeras i detta schema. Lösningar av uppgifterna beskrivs i separat flödesschema.

Avgränsad uppgift där TV-skärm, tangentbord eller andra in- /ut-enheter är involverade. Uppgiften anges i symbolen.

Numrerad utgång från flödesschemat. Flödesschemat fortsätter på en annan plats, där den här symbolen återfinns med samma nummer.

ÖVNINGSEXEMPEL

7.1 Programmet

10 PRINT CHR\$(7)

medför att datorn ger en ljudsignal ifrån sig. Det är samma signal som ljuder då datorn lämnar ett felmeddelande.

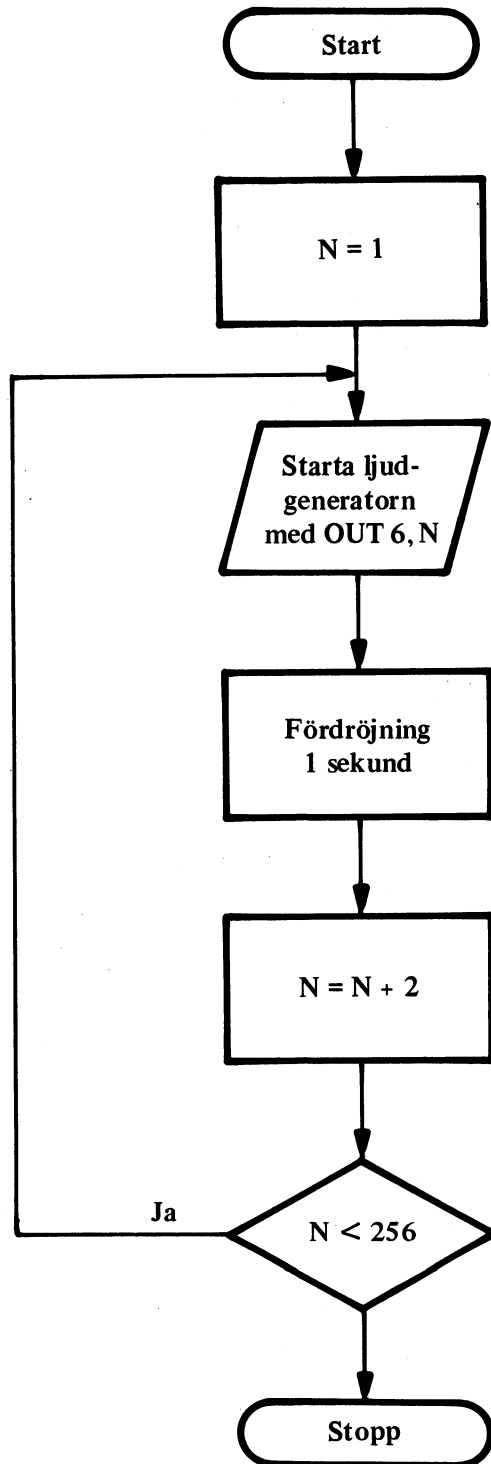
Gör ett flödesschema och ett program som kan användas vid familjens äggkokning. Datorn ska fråga efter koktid och därefter ge en signal efter den angivna tiden. Signalen ska inte upphöra förrän den ansvarige äggkokaren kvitterar signalen genom att trycka på tangenterna CTRL och C.

- 7.2 Använd det första programmet i kapitel 7 för att bestämma hur många år det dröjer innan en bil, som i dag kostar 50 000 kr, prissätts med 1 miljon kr om inflationen är 9%.
- 7.3 I ABC 80 finns en inbyggd ljudgenerator. Ljud från denna kan åstadkommas med BASIC-instruktionen OUT 6, N där N är ett udda tal mellan 1 och 255. Olika värden på N ger olika ljudeffekter. OUT 6, 0 stänger av ljudgeneratoren. Nedanstående program demonstrerar hur ljudgeneratoren låter för N = 7.

```
10 OUT 6, 7 : REM ** START AV LJUD **  
20 FOR F = 0 TO 5000 : NEXT F  
30 OUT 6, 0 : REM ** STOPP AV LJUD **
```

Rita ett flödesschema för ovanstående program. Prova gärna med andra värden på N.

- 7.4 Skriv ett program med utgångspunkt från flödesschemat på nästa sida.



- 7.5 Följande program genererar tre olika ljud när tangenterna A, B resp C nedtrycks.

```
10 GET A↵
20 IF A↵ = "A" THEN GOSUB 100
30 IF A↵ = "B" THEN GOSUB 200
40 IF A↵ = "C" THEN GOSUB 300
50 GO TO 10
100 OUT 6, 5 : RETURN
200 OUT 6, 3 : RETURN
300 OUT 6, 251 : RETURN
```

Utöka programmet så att ytterligare tre ljud skapas när tangenterna D, E och F nedtrycks.

KAPITEL 8

Datorn hanterar bokstavs- kombinationer

ABC80

DATORN SKRIVER BREV

Att skriva "personliga" brev som reklam är ett vanligt område där datorer används. Låt oss titta på ett program där datorn utnyttjas för att skriva ett brev, som ges personlig karaktär genom att en lämpligt vald textsträng upprepas på några ställen. När man skriver brev, kommer möjligheten att använda små bokstäver väl till pass. Tryck därför på tangenten  så att lysdioden i knappen slocknar. Tangentbordet fungerar nu som på en vanlig skrivmaskin, där ju normalt små bokstäver erhålls, när bokstavstangenterna trycks ner. Stora bokstäver erhålls med hjälp av skifttangenten. Du kan skriva BASIC-instruktioner och kommandon med små bokstäver om du vill. Återgå till "UPPER CASE"-läge när du inte längre har behov av små bokstäver.

```
10 PRINT "TILL VEM";  
20 INPUT A;  
30 PRINT CHR$(12)  
40 PRINT "Hej "; A;  
50 PRINT "Du "; A; " är välkommen på"  
60 PRINT "kalas på torsdag kl. 14.00"  
70 PRINT "Hej då "; A; ". Vi ses!"  
80 PRINT  
90 PRINT "Kram Lena"
```

Variera PRINT-satserna, så du får ett brev som passar dig.

STRÄNGELEMENT

En följd av tecken, omgivna av citationstecken, kallas sträng. Strängar med maximalt 80 tecken kan representeras i form av strängvariabler utan dimensionering. Efter dimensionering (med DIM) kan strängar innehålla ett godtyckligt antal tecken. Om vi vill behandla de olika tecknen i en sträng, måste vi införa följande regel för hur de olika elementen i en sträng numreras.

Teckenposition nummer	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25		
Sträng	"	A	B	C		O	M		B	A	S	I	C	P	R	O	G	R	A	M	M	E	R	I	N	G	"

A är tecken nr 1 och G är tecken nr 25 i ovanstående exempel.

STRÄNGFUNKTIONERNA LEFT α , RIGHT α och MID α

När vi nu vet hur de olika tecknen i en sträng anges, kan vi börja använda de speciella strängfunktioner som finns i ABC 80.

Vi tilldelar strängvariabeln A α värde på följande sätt:

```
10 A $\alpha$  = "ABC OM BASICPROGRAMMERING"
```

Den första strängfunktion som vi ska använda är LEFT α (A α , J). Denna funktion ger som resultat de J första tecknen i strängen A α .

Raden

```
20 B $\alpha$  = LEFT  $\alpha$  (A $\alpha$ , 3)
```

tilldelar strängvariabeln B α värdet "ABC".

Nästa strängfunktion är RIGHT α (A α , J), som ger alla tecken från och med tecken J i strängen A α . Raden

```
30 C $\alpha$  = RIGHT  $\alpha$  (A $\alpha$ , 8)
```

tilldelar strängvariabeln C α värdet "BASICPROGRAMMERING".

För att kunna tilldela strängvariabeln D α värdet "OM", använder vi strängfunktionen MID α (A α , I, L) som ger L tecken från och med tecken I i strängen A α .

```
40 D $\alpha$  = MID  $\alpha$  (A $\alpha$ , 5, 2)
```

Avsluta programmet med lämpliga PRINT-satser, så att följande utskrift erhålles:

```
ABC  
OM  
BASICPROGRAMMERING
```

Hur skall programmet förändras, för att följande utskrift skall erhållas?

```
ABC                OM  
BASICPROGRAMMERING
```

Du ser att vi med dessa strängfunktioner kan bearbeta innehållet i en sträng på olika sätt. Fundera på hur nedanstående program förändrar utskriften av strängen A α .

```
10 A $\alpha$  = "ABC OM BASICPROGRAMMERING"  
20 FOR I = 1 TO 25  
30 IF MID $\alpha$  (A $\alpha$ , I, 1) = " " THEN PRINT " ";  
40 PRINT MID $\alpha$  (A $\alpha$ , I, 1);  
50 NEXT I
```

När du kör programmet, märker du att vi får tre mellanslag mellan varje ord i stället för ett, som vi fick tidigare. Detta beror på att vi i rad 30 testat tecken för tecken i strängen A α . När datorn finner att tecknet är ett mellanslag, så skrivs två extra mellanslag på TV-skärmen.

Tillämpningar där datorn skriver brev och hanterar ordkombinationer är idag mycket vanliga. Många sådana system är skräddarsydda för att enbart klara ordbehandlingsfunktioner, men även generella datorer som ABC 80 lämpar sig utmärkt för sådana uppgifter.

BOOKSTÄVER HAR VÄRDE

Internt i en dator kan enbart siffror hanteras. Bokstäver och andra tecken motsvaras därför internt i datorn av en sifferkombination. Ett sätt att numrera bokstäver och tecken är med hjälp av ASCII-kod (American Standard Code for Information Interchange). ASCII uttalas "aski". Tabellen på nästa uppslag ger ASCII-koden för de tecken som används i ABC 80.

Flera strängfunktioner utnyttjar egenskapen att varje tecken motsvaras av en sifferkombination.

Strängfunktionen ASC(A α) ger ASCII-värdet på första tecknet i A α . Kör nedanstående program och kontrollera resultaten med ASCII-tabellen.

```
10 INPUT A $\alpha$   
20 PRINT ASC(A $\alpha$ )  
30 IF A $\alpha$  = "SLUT" THEN 50  
40 GOTO 10  
50 END
```

När du vill sluta tilldelar du A α värdet "SLUT".

Strängfunktionen CHR\$(I) ger som resultat det tecken, som har ASCII-koden I.

Raden

```
10 PRINT CHR$(43)
```

ger utskrift av ett " + " -tecken på TV-skärmen. Strängfunktionen CHR\$(I) kan innehålla maximalt fyra ASCII-koder.

Vilken utskrift ger följande rad?

```
10 PRINT CHR$(82,123,116,116)
```

Låt oss som ytterligare tillämpning titta på hur datorn kan fås att skriva ett antal slumpmässiga ord med tre bokstäver.

```
10 REM TEXTGENERERING
20 RANDOMIZE
30 FOR X = 1 TO 60
40 FOR I = 1 TO 3
50 A(I) = INT(65 + 29 * RND)
60 NEXT I
70 PRINT CHR$(A(1),A(2),A(3)),
80 NEXT X
```

Rad nr	Radens funktion
10	Kommentar
20	Ger slumpmässigt startvärde på RND.
30, 40, 60, 80	Skapar två loopar.
50	Tilldelar fältet A slumpstal mellan 65 och 93.
70	Skriver ut tre stora bokstäver. Kontrollera i ASCII-tabellen att de stora bokstäverna motsvarar intervallet 65 till 93.

ASCII-TABELL

ASCII	Tecken	ASCII	Tecken	ASCII	Tecken
7	Bell	59	;	93	À
10	Radfram- matning (LF)	60	<	94	Û
		61	=	95	—
		62	>	96	é
12	Blank skärm (FF)	63	?	97	a
		64	É	98	b
13	Vagnretur (CR)	65	A	99	c
32	Blank	66	B	100	d
33	!	67	C	101	e
34	“	68	D	102	f
35	#	69	E	103	g
36	¤	70	F	104	h
37	%	71	G	105	i
38	&	72	H	106	j
39	'	73	I	107	k
40	(	74	J	108	l
41	)	75	K	109	m
42	*	76	L	110	n
43	+	77	M	111	o
44	,	78	N	112	p
45	-	79	O	113	q
46	.	80	P	114	r
47	/	81	Q	115	s
48	0	82	R	116	t
49	1	83	S	117	u
50	2	84	T	118	v
51	3	85	U	119	w
52	4	86	V	120	x
53	5	87	W	121	y
54	6	88	X	122	z
55	7	89	Y	123	ä
56	8	90	Z	124	ö
57	9	91	Ä	125	å
58	:	92	Ö	126	ü
				127	■

DATORN JÄMFÖR STRÄNGAR

När du nu har sett att datorn kan tolka värdet av en bokstav som motsvarande ASCII-kod, är det lättare att förstå att vi kan jämföra strängar med varandra och t ex sortera i bokstavsordning. Följande regler gäller:

När två lika långa strängar jämförs, jämförs ASCII-koden för de första tecknen i respektive sträng och avgör vilken sträng som är störst. Om de första tecknen är lika, avgör de bägge andra tecknen vilken sträng som är störst, o s v.

När två olika långa strängar jämförs, så jämförs den kortare strängen (med längden n) med de n första tecknen i den längre strängen. Om den kortare strängen är lika med början av den längre, så är den längre strängen större än den kortare.

Alla tidigare använda relationsoperatorer kan användas tillsammans med strängar.

Om t ex $A \text{ } \text{<} \text{ } = \text{"AB"} \text{ } , B \text{ } \text{<} \text{ } = \text{"ABC"} \text{ } , \text{ och } C \text{ } \text{<} \text{ } = \text{"BC"}$

gäller att

$$A \text{ } \text{<} \text{ } < C \text{ } \text{<} \text{ }$$
$$A \text{ } \text{<} \text{ } < B \text{ } \text{<} \text{ }$$
$$B \text{ } \text{<} \text{ } < C \text{ } \text{<} \text{ }$$

SORTERA NAMN

Att sortera många namn i bokstavsordning kan vara mödosamt att göra för hand. Genom att utnyttja datorns förmåga att jämföra strängar kan vi låta datorn utföra sorteringen. Vi kan använda samma rutin för sortering som använts tidigare. För att förenkla inmatningen använder vi instruktionen INPUTLINE. När en strängvariabel tilldelas värde med INPUTLINE, inmatas en komplett rad med både blanktecken och övriga tecken, utan att man inramar texten med citationstecken. Även tecknen som motsvarar radframmatning och vagnretur ingår i strängen, vilket medför en extra radframmatning när en sådan sträng skrivs på TV-skärmen.

```

10 PRINT "HUR MÅNGA NAMN";
20 INPUT N
30 FOR X = 1 TO N
40 INPUT LINE A$(X)
50 PRINT
60 NEXT X
70 FOR I = 1 TO N - 1
80 FOR J = I + 1 TO N
90 IF A$(I) < A$(J) THEN 110
100 X$ = A$(I) : A$(I) = A$(J) : A$(J) = X$
110 NEXT J
120 NEXT I
130 PRINT CHR$(12)
140 FOR X = 1 TO N
150 PRINT A$(X)
160 NEXT X

```

Mata in hur många namn du vill sortera och därefter de aktuella namnen. Efter att det sista namnet matats in, presenteras direkt den sorterade listan. Om du vill mata in flera än 10 namn, måste det tillfogas en rad som dimensionerar fältet A\$.

```
5 DIM A$(20) = 30
```

Tillfogas denna rad kan 20 namn sorteras. Vi inför samtidigt begränsningen att varje sträng får innehålla max 30 tecken. Denna restriktion införs för att inte i onödan ta upp plats i datorns minne.

Finns det några risker för fel i ovanstående sorteringsprogram? Ja, kontrollera ordningsföljden för bokstäverna Å, Ä och Ö i ASCII-tabellen! För att bokstäverna Å, Ä och Ö ska sorteras i rätt ordning, måste ett tillägg göras till programmet. Ett förslag ges i övningsexempel 8.5.

SORTERA EFTER LÄNGD

Strängfunktionen LEN(A\$) ger som resultat längden av strängen A\$.

```

10 A$ = "ABC OM"
20 PRINT LEN(A$)

```

Ovanstående program ger som resultat utskrift av siffran 6 på skärmen, d v s det aktuella antalet element i strängen A\$. Strängfunktionen LEN(A\$) kan an-

vändas för att sortera strängar efter stigande längd. Antag t ex att en affärsinnehavare på sin ABC 80 vill presentera dagens "extrapriser". Fundera på hur sorteringsprogrammet i föregående avsnitt kan modifieras, så att inmatade strängar sorteras efter stigande längd innan de presenteras!

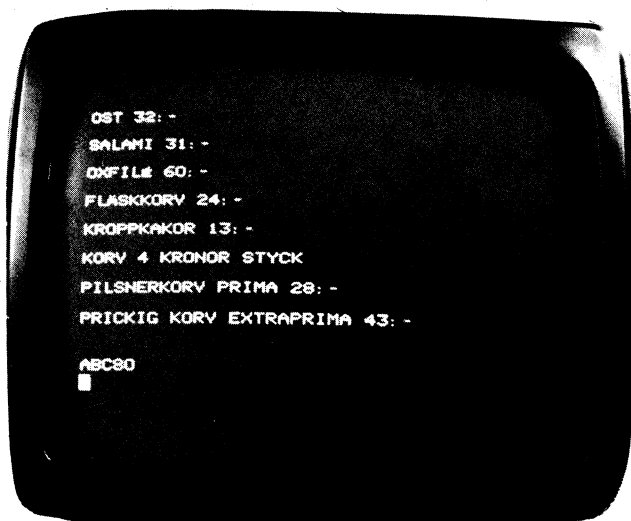
Förslag till modifierat program.

```
10 PRINT "HUR MÅNGA ARTIKLAR";
20 INPUT N
30 FOR X = 1 TO N
40 INPUT LINE A$(X)
50 PRINT
60 NEXT X
70 FOR I = 1 TO N-1
80 FOR J = I + 1 TO N
90 IF LEN(A$(I)) < LEN(A$(J)) THEN 110
100 X$ = A$(I) : A$(I) = A$(J) : A$(J) = X$
110 NEXT J
120 NEXT I
130 PRINT CHR$(12)
140 FOR X = 1 TO N
150 PRINT A$(X)
160 NEXT X
```

När ovanstående program kördes hos kött- och fläskhandlaren Frick användes följande 8 indata.

```
HUR MÅNGA ARTIKLAR? 8
KORV 4 KRÖNOR STYCK
OXFILÉ 60:-
PILSNERKORV PRIMA 28:-
OST 32:-
PRICKIG KORV EXTRAPRIMA 43:-
KROPPKAKOR 13:-
FLÄSKKORV 24:-
SALAMI 31:-
```

Resultatet efter körning framgår av nedanstående figur.



Sortering efter stränglängd

SAMMANFOGNING AV STRÄNGAR

Den enda matematiska operation som kan utföras med strängar är addition. Följande exempel visar funktionen.

```
10 PRINT "HEJ" + "SAN"
```

Ett exempel som mera fullständigt visar hur sammansättning av strängar kan utnyttjas praktiskt är följande program som skulle kunna användas av Kungl. namnregistreringsverket för att konstruera nya namnförslag.

```
10 FOR I = 1 TO 8
20 INPUT A$(I), B$(I)
30 NEXT I
40 FOR I = 1 TO 8
50 FOR J = 1 TO 8
60 PRINT A$(I) + B$(J),
70 NEXT J
80 NEXT I
```

Programmet består i raderna 10, 20 och 30 av en loop som läser in 8 strängar till fältet A\$ med lämpliga inledningar till efternamn och 8 strängar till fältet B\$ med lämpliga avslutningar till efternamn. Provkör programmet med t ex följande ordkombinationer.

SKOGS, ÄNG
 LIND, ROT
 PALM, BLAD
 LÖNN, ROS
 VINTER, HED
 SOMMAR, LÖV
 LAND, GREN
 LUND, KVIST

Resultatet blir 64 kombinationer, som nedanstående bild visar.



Sammansättning av strängar

STRÄNGAR BLIR TAL

Ibland har man behov av att kunna utföra beräkningar på siffror som ingår som tecken i strängar. Därför finns funktionen $\text{VAL}(A\alpha)$ som omvandlar siffer-tecken i strängar till tal.

```

10 A $\alpha$  = "5"
20 X = VAL(A $\alpha$ )
30 PRINT X * X * 2

```

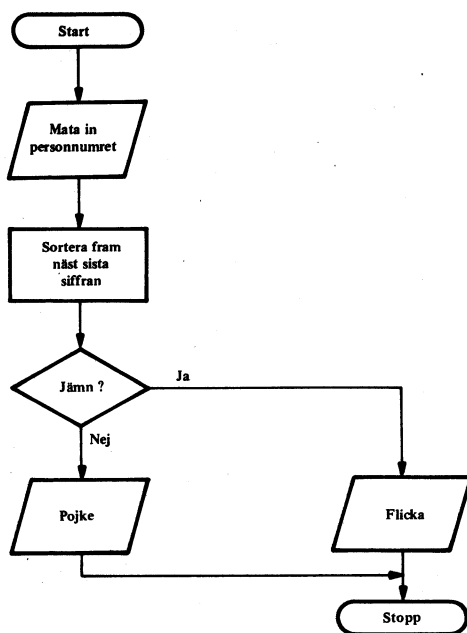
I ovanstående program tilldelas först strängvariabeln $A\alpha$ innehållet siffran 5. Därefter ges flyttalsvariabeln X värdet 5 med hjälp av strängfunktionen $\text{VAL}(A\alpha)$. Flyttalsvariabler kan ingå i aritmetiska uttryck och utskriften efter programkörningen blir 50.

Nedanstående exempel visar en mera realistisk användning av strängfunktionen VAL(A α).

Som du säkert känner till avslöjar näst sista siffran i ett personnummer om personnumret tillhör en person av manligt eller kvinnligt kön. Vi ska nu se på ett program som testar näst sista siffran i ett personnummer. Om siffran är udda skriver datorn "DU ÄR EN POJKE" och om siffran är jämn skriver den "DU ÄR EN FLICKA". Programmet utnyttjar bl a strängfunktionen VAL(B α).

```
10 PRINT "MATA IN DITT PERSONNUMMER"  
20 PRINT  
30 INPUT A $\alpha$   
40 B $\alpha$  = MID $\alpha$ (A $\alpha$ ,10,1)  
50 B = VAL(B $\alpha$ )  
60 IF B/2 = INT(B/2) THEN 100  
70 PRINT  
80 PRINT "DU ÄR EN POJKE"  
90 GOTO 120  
100 PRINT  
110 PRINT "DU ÄR EN FLICKA"  
120 PRINT
```

Programmets funktion beskrivs i följande flödesschema.



TAL BLIR STRÄNGAR

Strängfunktionen $\text{NUM}\varnothing(A)$ omvandlar tal till strängar, d v s den är motsatsen till funktionen $\text{VAL}(A\varnothing)$. Strängen som erhålls med $\text{NUM}\varnothing(A)$ ger en sträng med de tecken som man skulle fått, om man skrivit ut A med en PRINT-sats.

```
10 A = 1234
20 A $\varnothing$  = NUM $\varnothing$ (A)
30 FOR X = 1 TO 5
40 PRINT MID $\varnothing$ (A $\varnothing$ , X, 1); " ";
50 NEXT X
```

När positiva tal skrivs ut i PRINT-satser erhålls ett mellanslag i stället för plus-tecken. Detta mellanslag återfinns därför i strängen $\text{NUM}\varnothing(A)$. I exemplet ovan blir alltså $\text{NUM}\varnothing(A) = " 1234"$, d v s strängen innehåller 5 element.

Genom att omvandla flyttalsvariabeln A till en strängvariabel $A\varnothing$ kan vi påverka utskriften av sifferkombinationerna med hjälp av tidigare använda strängfunktioner. I detta fall skrivs siffrorna ut med två mellanslag mellan varje.

AUTOMATISK UNDERSTRYKNING

Strängfunktionen $\text{STRING}\varnothing(L, A)$ ger en sträng innehållande L st av det ASCII-tecken som motsvarar ASCII-värdet A. Tex ger $\text{STRING}\varnothing(5, 42)$ strängen "*****". Jämför ASCII-tabellen på sid 88. Strängfunktionen $\text{STRING}\varnothing(L, A)$ kan användas för automatisk understrykning av strängar, som visas i nedanstående program.

```
10 INPUT A $\varnothing$ 
20 PRINT CHR $\varnothing$ (12)
30 PRINT A $\varnothing$ 
40 L = LEN(A $\varnothing$ )
50 PRINT STRING $\varnothing$ (L, 61)
```

Vilket ASCII-tecken används för understrykning? Ändra gärna rad 50 så att andra ASCII-tecken används för "understrykning".

DATORN SÖKER EFTER ORD

Det går lätt att undersöka om en viss teckenkombination ingår i en sträng med hjälp av strängfunktionen $\text{INSTR}(I, A\varnothing, B\varnothing)$.

INSTR(I,A α ,B α) söker efter strängen B α i strängen A α från och med tecken I. Resultatet av funktionen blir läget av första förekomsten av B α i A α eller, om B α inte finns i A α , värdet 0.

Om A α = "ABABCAB" och B α = "AB" så kommer strängfunktionen INSTR(2,A α ,B α) att ge resultatet 3.

Innan vi studerar ett enkelt exempel ska vi först införa strängfunktionen SPACE α (I), som ger en sträng bestående av I stycken mellanslag (eng space).

I nedanstående exempel testas om texten BASIC finns med i en inmatad sträng. Om så är fallet skrivs strängen ut och texten BASIC stryks under, annars erhålls utskrift som talar om att texten BASIC inte ingår i strängen.

```
10 INPUT A $\alpha$ 
20 B $\alpha$  = "BASIC"
30 X = INSTR (1, A $\alpha$ , B $\alpha$ )
40 IF X = 0 THEN 50
50 PRINT CHR$(12)
60 PRINT A $\alpha$ 
70 PRINT SPACE$(X-1); "-----"
80 GOTO 100
90 PRINT "TEXTEN - BASIC - INGÅR EJ"
100 PRINT
```

SAMMANFATTNING

Instruktion	Kort beskrivning	Exempel
INPUTLINE	Läser in en rad från tangentbordet komplett med blanktecken och citationstecken. Även tecken för vagnretur och radframmatning ingår på slutet.	50 INPUTLINE A α
DIM	Reserverar utrymme. Följande varianter finns.	
	Dimensionering av sträng.	10 DIM A α = 200
	Dimensionering av variabelfält.	30 DIM A(20) 40 DIM A%(50)
	Dimensionering av strängfält.	50 DIM A α (20) = 30

Strängfunktion	Kort beskrivning	Exempel
LEFT $\varnothing$ (A $\varnothing$,J)	Ger de J första tecknen i strängen A $\varnothing$.	10 C $\varnothing$ = LEFT $\varnothing$ (A $\varnothing$,J)
MID $\varnothing$ (A $\varnothing$,I,J)	Ger J tecken från och med teckenposition I.	20 D $\varnothing$ = MID $\varnothing$ (A $\varnothing$,I,J)
RIGHT $\varnothing$ (A $\varnothing$,I)	Ger alla tecken i A $\varnothing$ från och med tecken I.	30 D $\varnothing$ = RIGHT $\varnothing$ (A $\varnothing$,I)
LEN (A $\varnothing$)	Ger aktuella längden av strängen A $\varnothing$.	40 L = LEN (A $\varnothing$)
ASC (A $\varnothing$)	Ger ASCII-värdet på första tecknet i A $\varnothing$.	50 V = ASC (A $\varnothing$)
CHR $\varnothing$ (I)	Ger ett tecken med ASCII-värdet I. Kan innehålla upp till fyra ASCII-tecken.	10 PRINT CHR $\varnothing$ (7) 5 PRINT CHR $\varnothing$ (65,66,67)
INSTR (I,A $\varnothing$,B $\varnothing$)	Söker efter strängen B $\varnothing$ i strängen A $\varnothing$ med början i pos. I. Funktionen ger läget för första förekomsten av B $\varnothing$ i A $\varnothing$, eller, om B $\varnothing$ ej finns i A $\varnothing$, värdet 0.	60 P = INSTR (I,A $\varnothing$,B $\varnothing$)
SPACE $\varnothing$ (I)	Ger en sträng av I st mellan- slag (space).	50 PRINT SPACE $\varnothing$ (I)
STRING $\varnothing$ (I,C)	Ger en sträng av I stycken CHR $\varnothing$ (C).	60 PRINT STRING $\varnothing$ (I,C)
NUM $\varnothing$ (A)	Ger en sträng med de tecken man skulle fått om man skrivit ut A med en PRINT-sats.	70 A $\varnothing$ = NUM $\varnothing$ (A)
VAL (A $\varnothing$)	Ger värdet av A $\varnothing$ tolkat som ett tal.	80 A = VAL (A $\varnothing$)
+	Sammanfogar strängar.	10 A $\varnothing$ = B $\varnothing$ + C $\varnothing$ + "!"

ÖVNINGSEXEMPEL

- 8.1 Skriv ett program som ger utskriften

VAD HETER DU MIN VÄN?

Därefter ska A☒ via en INPUT-sats tilldelas ett namn som värde, t ex BJÖRN, och datorn skall därefter ge utskriften

HEJ BJÖRN

BJÖRN ÄR ETT VACKERT NAMN

- 8.2 Skriv ett program som tilldelar A☒ värdet "ABC OM PROGRAMMERING". Använd sedan lämpliga strängfunktioner så att programmet ger utskriften

ABC - OM - PROGRAMMERING

- 8.3 Skriv ett program som via en INPUT-sats tilldelar A☒ värdet av ett personnummer. Därefter skall följande utskrift presenteras, där årtalet bestäms hjälp av personnumrets två första siffror.

DU ÄR FÖDD ÅR 1944

- 8.4 Skriv ett program som skriver 60 bokstavskombinationer, omfattande tre bokstäver, där den första alltid är S. Programmet kan lämpligen användas vid tillsättning av statliga utredningar för att hitta förslag till lämpliga förkortningar.

- 8.5 Nedanstående program sorterar namn i rätt ordning även med avseende på bokstäverna Å, Ä och Ö. Vissa begränsningar finns emellertid. Kör programmet och försök visa vilka begränsningar som finns.

```
10 PRINT "HUR MÅNGA NAMN";
20 INPUT N
30 FOR X = 1 TO N
40 INPUT LINE A$(X)
50 PRINT
60 NEXT X
70 FOR I = 1 TO N - 1
80 FOR J = I + 1 TO N
90 IF A$(I) < A$(J) THEN 110
100 X$ = A$(I) : A$(I) = A$(J) : A$(J) = X$
110 NEXT J
120 NEXT I
130 PRINT CHR$(12)
140 FOR X = 1 TO N
150 IF ASC(A$(X)) = 91 THEN 180
160 IF ASC(A$(X)) = 92 THEN 180
170 PRINT A$(X)
180 NEXT X
190 FOR X = 1 TO N
200 IF ASC(A$(X)) = 91 THEN PRINT A$(X)
210 IF ASC(A$(X)) = 92 THEN PRINT A$(X)
220 NEXT X
```

- 8.6 Skriv ett program som läser in en sträng, tex ett namn, och ger utskrift av strängen inramad på nedanstående sätt.

```
*****
* GRAND HOTEL *
*****
```

- 8.7 Efter genomförd 3-dagarskurs vid Grönköpings luftvärnsskår utnämns alla deltagande furirer till överfurirer. Skriv ett program som läser in nedanstående deltagarlista.

FURIR EMIL STORM
FURIR ADAM RAPP
FURIR NILS BOM
FURIR HUGO KRUT

Programmet skall därefter skriva ut samma lista men med alla deltagarna utnämnda till överfurirer.

- 8.8 En svensk chefsbyråkrat hade svårigheter med att uttala bokstaven S. Han uppdrog därför åt sin chefsekreterare att testa alla utskrivna tal, rad för rad, och tillse att antalet S i raderna aldrig översteg 5% av alla tecken. Skriv ett program som testar förekomsten av S i en rad och slår larm om frekvensen av S överstiger 5%.
- 8.9 I ett kungarike i norra Europa utbröt revolution och gamle Kung Sune störtades. Den nye diktatorn King Kong införde ett påbud att i alla gamla skrifter skulle texten KUNG SUNE utbytas mot KING KONG. Skriv ett program där meningar på max 120 tecken kan inmatas och som automatiskt ändrar texten KUNG SUNE till KING KONG. Det förutsätts att texten KUNG SUNE förekommer bara en gång per mening.

KAPITEL 9

Datorn löser matematiska problem

ABC80

YTTERLIGARE BERÄKNINGSMÖJLIGHETER

ABC 80 kan inte bara räkna med de fyra räknesätten, den kan också direkt räkna ut värdet på de vanligaste matematiska funktionerna. De funktioner som ABC 80 kan beräkna är potensfunktioner, trigonometriska funktioner, logaritmfunktioner, exponentialfunktioner samt kvadratrötter. Dessutom innehåller ABC 80 några specialfunktioner såsom heltalsdel, absolutvärde, slumpal m m.

Alla funktioner kan användas precis på samma sätt som tal och variabler används i BASIC-satserna.

POTENSUTTRYCK

Du har i kap 4 lärt dig hur tiopotenser kan användas för att representera tal. ABC 80 kan även användas för att beräkna godtyckliga potensuttryck. Ett exempel på potensuttryck är 2^8 . Tvåan kallas bas och åttaan exponent.

I den version av BASIC som används av ABC 80 skrivs potensfunktioner med hjälp av symbolerna "Ü" (observera att du måste använda den stora bokstaven Ü) eller "***".

Exempel	2^8
skrivs som	$2 \text{ Ü } 8\%$
alternativt	$2 *** 8\%$

Bas och exponent får vara konstanter eller variabler i flyttalsrepresentation eller heltalsrepresentation. Om exponenten är ett heltal förbättras dock noggrannheten om den definieras som ett heltal, d v s med %-tecken. Negativa tal får enbart upphöjas till heltalsexponenter.

Följande program beräknar värdet av uttrycket

$$y = x + 2x^2 + 3x^3$$

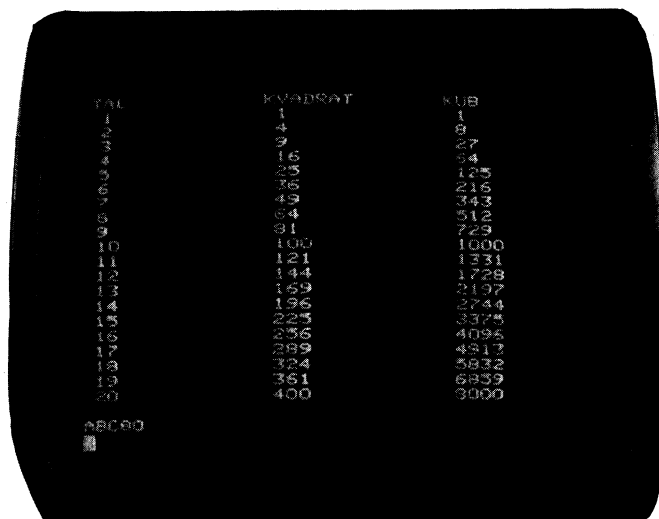
för olika värden på x.

```
10 PRINT CHR 12 (12)
20 PRINT
30 PRINT
40 PRINT "X=";
50 INPUT X
60 LET Y = X + 2 * X ** 2% + 3 * X ** 3%
70 PRINT "Y="; Y
80 GOTO 20
```


Eftersom exponenterna i detta exempel är heltal, anges de med %-tecken. Mata in några olika värden på x och studera resultatet.

Nästa program ger utskrift i tabellform av samhörande värden på x, x^2 och x^3 .

```
10 PRINT CHR$(12)
20 PRINT "TAL","KVADRAT","KUB"
30 FOR X = 1 TO 20
40 PRINT X, X**2%, X**3%
50 NEXT X
```



TAL	KVADRAT	KUB
1	1	1
2	4	8
3	9	27
4	16	64
5	25	125
6	36	216
7	49	343
8	64	512
9	81	729
10	100	1000
11	121	1331
12	144	1728
13	169	2197
14	196	2744
15	225	3375
16	256	4096
17	289	4913
18	324	5832
19	361	6859
20	400	8000

ABC 800

Tabell som visar potenser av x

FUNKTIONER

Som nämndes i inledningen till detta kapitel, finns i ABC 80-BASIC ett antal matematiska standardfunktioner, som underlättar programmeringen i samband med matematiska problem.

Följande matematiska funktioner förekommer.

Funktion	Betyder	Anmärkning
SIN(X)	$\sin x$	x i radianer
COS(X)	$\cos x$	
TAN(X)	$\tan x$	
ATN(X)	$\arctan x$	
LOG(X)	$\ln x$	naturliga logaritmen av x
LOG 10(X)	$\lg x$	10-logaritmen av x
EXP(X)	e^x	
SQR(X)	$\sqrt{x}$	kvadratroten ur x

Följande specialfunktioner förekommer.

Funktion	Beskrivning
INT(X)	Det största heltal som är mindre än eller lika med x.
ABS(X)	Absolutbeloppet av x, $ x $.
SGN(X)	-1 om $x < 0$ 0 om $x = 0$ 1 om $x > 0$
FIX(X)	Heltalsdelen av x, $[x]$.
PI	π , d v s talet 3,14159.
RND	Slumptal mellan 0 och 0,999999 inklusive dessa.

En del av dessa har du redan träffat på, nämligen INT(X) samt RND som bl a användes i kap 5.

De flesta funktioner följs av en parentes. I parentesen ska det s k argumentet skrivas. Alla funktioner kan ingå i aritmetiska uttryck. Argumenten kan också utgöras av aritmetiska uttryck.

ARITMETISKA UTTRYCK

Ett aritmetiskt uttryck kan bestå av konstanter, variabler, funktioner, parenteser och aritmetiska operationer. Uppbyggnaden av ett aritmetiskt uttryck följer de vanliga matematiska reglerna.

Några exempel på aritmetiska uttryck.

Normalt skrivsätt

Skrivet i BASIC

$$18 \cos x + x^2$$

$$18 * \text{COS}(X) + X ** 2$$

$$\ln(\arctan(\frac{X}{20})) + y(x + y)$$

$$\text{LOG}((\text{ATN}(X/20))) + Y * (X + Y)$$

$$\pi(A + \lfloor B \rfloor)$$

$$\text{PI} * (A + \text{INT}(B))$$

DEFINITION AV EGNA FUNKTIONER

Med hjälp av DEF FN ... -satsen kan du definiera egna funktioner. Exempel.

```
10 DEF FNY(X) = SIN(X) * EXP(X + 1)
```

```
100 DEF FNA(X) = TAN(X * SQR(X ** 3))
```

Tilldelningen sker alltså enligt DEF FNX(variabel) = aritmetiskt uttryck, där X är en godtycklig bokstav.

Det är en stor fördel att kunna definiera egna funktioner om funktionen i fråga ska anropas flera gånger i ett program, t ex för att beräkna funktionsvärdet för olika värden på en variabel. Funktionen behöver då enbart skrivas på ett ställe och programmet blir därför kortare.

Exempel.

Beräkna $f(1) - f(0,5) + f(2)$

$$\text{för } f(x) = e^{-x^2} + 1$$

Ett BASIC-program, som utför beräkningarna kan se ut på följande sätt.

```
10 DEF FNA(X) = EXP(-X ** 2) + 1
20 PRINT FNA(1) - FNA(.5) + FNA(2)
```

TABELLERING AV FUNKTIONSVÄRDEN

Du har redan i avsnittet om potensuttryck träffat på ett exempel, där funktionsvärden skrivs ut i tabellform.

Följande program åstadkommer på samma sätt tabellering av samhörande värden på x , $\sin x$ och $\cos x$.

```
10 PRINT "X","SIN(X)","COS(X)"
20 FOR X = 0 TO 6.4 STEP .4
30 PRINT X, SIN(X), COS(X)
40 NEXT X
```

På rad 20 i programmet finns en FOR ... TO ... -instruktion. När du använt denna instruktion tidigare har steglängden i en loop alltid varit 1, d v s loopvariabeln har automatiskt ökat med 1 för varje gång som loopen genomlöpts. Genom att utöka instruktionen med STEP, kan steglängden ändras till ett godtyckligt värde. I detta program ökas loopvariabeln X med 0,4 varje gång som loopen på raderna 20, 30 och 40 genomlöps. Körning av programmet ger följande utskrift.



X	SIN(X)	COS(X)
0	.000000	1.000000
.4	.389421	.921060
.8	.717354	.696327
1.2	.932044	.360366
1.6	.999574	-.009187
2.0	.909333	-.416149
2.4	.675460	-.737479
2.8	.334980	-.942226
3.2	-.058369	-.998351
3.6	-.442318	-.895756
4.0	-.756301	-.653644
4.4	-.931605	-.337232
4.8	-.961616	-.047507
5.2	-.837460	.163513
5.6	-.571227	.275369
6.0	-.279415	.260165
6.4	.116555	.393180

Funktionsvärden i tabell

Resultatet vid beräkningen av $\sin x$ och $\cos x$ ges alltid med 6 siffror. Eventuella nollor som är utan betydelse skrivs inte ut. I många fall behöver man inte presentera resultatet med 6 siffror. Med hjälp av INT-funktionen kan resultatet av en beräkning avrundas till önskat antal siffror.

KORREKT AVRUNDNING

INT(X) ger det största heltal som är mindre än eller lika med X.

Du har redan i kap 4 använt INT-funktionen för att avrunda beräkningsresultat till hela kronor.

Exempel

$$\text{INT}(610.991) = 610$$

$$\text{INT}(18) = 18$$

$$\text{INT}(-7.8) = -8 \quad (-8 \text{ är mindre än } -7,8)$$

En nackdel är att avrundningen alltid sker till närmast mindre heltal. Normalt önskar man att avrundning sker till närmaste heltal.

Exempel

$$18.588 \text{ avrundas till } 19$$

$$18.493 \text{ avrundas till } 18$$

För att göra en sådan avrundning adderas först 0,5 till talet, varefter heltalsdelen tas av summan.

Exempel

$$\text{INT}(18.588 + .5) = 19$$

$$\text{INT}(18.493 + .5) = 18$$

Om du istället vill avrunda till en decimal, multiplicerar du först talet med 10 och dividerar därefter hela INT-funktionen med 10.

Exempel

$$\text{INT}(10 * 18.588 + .5)/10 = 18.6$$

$$\text{INT}(10 * 18.493 + .5)/10 = 18.5$$

Vid avrundning till två decimaler multipliceras talet med 100 och hela INT-funktionen divideras med 100.

Exempel

$$\text{INT}(100 * 18.588 + .5)/100 = 18.59$$

$$\text{INT}(100 * 18.493 + .5)/100 = 18.49$$

Följande generella formel för avrundning kan således ställas upp.

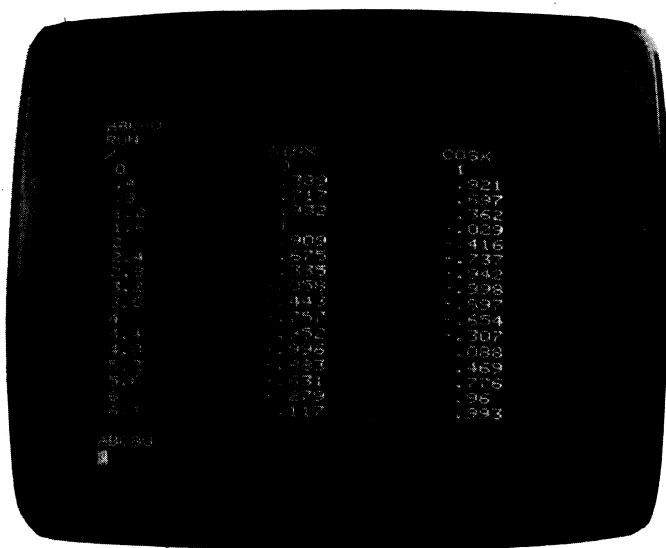
$$\text{INT}(10^{**D\%} * X + .5) / 10^{**D\%}$$

där X avrundas till D korrekta decimaler.

Vi ska nu använda INT-funktionen för att modifiera det tidigare programmet för tabellering av samhörande värden på x, sin x och cos x, så att funktionsvärdena avrundas till tre decimaler.

```
10 PRINT "X","SIN(X)","COS(X)"
20 FOR X = 0 TO 6.4 STEP .4
30 PRINT X, INT(1000 * SIN(X) + .5)/1000,
  INT(1000 * COS(X) + .5)/1000
40 NEXT X
```

Körning av programmet ger följande utskrift.



X	SIN(X)	COS(X)
0	0	1
.4	.393	.921
.8	.717	.697
1.2	.932	.362
1.6	1	-.009
2.0	.909	-.416
2.4	.675	-.737
2.8	.335	-.942
3.2	-.038	-.998
3.6	-.442	-.897
4.0	-.757	-.654
4.4	-.950	-.307
4.8	-.995	-.088
5.2	-.999	.169
5.6	-.972	.316
6.0	-.817	.58
6.4	-.517	.863

Funktionsvärden med önskat antal decimaler

PLOTTA FUNKTIONER MED PRINT TAB

Instruktionen TAB(K) kan användas tillsammans med PRINT, för att dirigera utskriften till önskad position på raden. Heltalsdelen av K, där K kan vara ett aritmetiskt uttryck, anger den position på raden där utskriften ska börja. Heltalsdelen av K får anta värden mellan 0 och 39.

Exempel

```
10 PRINT TAB(12); "MATEMATIK"
```

ger följande utskrift

```
      MATEMATIK
      ↑
Position 12
```

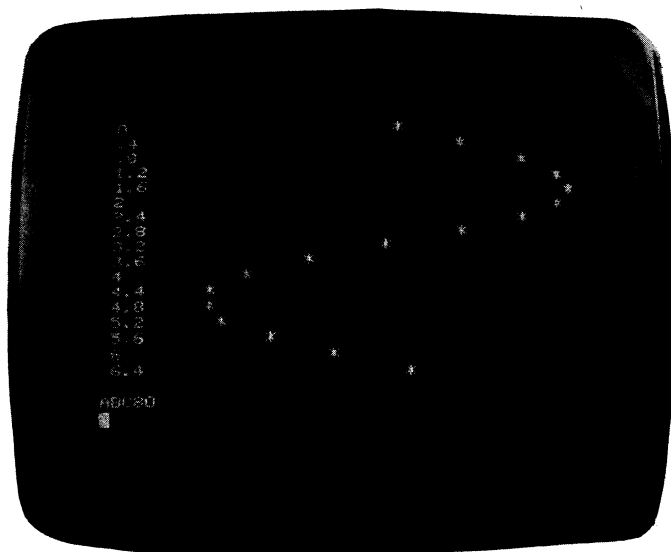
Med hjälp av TAB-instruktionen kan även kurvor plottas på TV-skärmen.

Vi väljer som exempel att plotta funktionen $y = \sin x$. I programmet ska ingå en PRINT-sats, som utformas så att x-värdet först skrivs längst till vänster på TV-skärmen, varefter kurvan plottas till höger. TAB-instruktionen ska användas för att skriva ett tecken, t ex "*", på en position av skärmen som bestäms av y-värdet. Dessutom måste funktionsvärdena skalas, d v s multipliceras med en faktor och eventuellt adderas till en konstant term, så att den tillgängliga ytan på skärmen utnyttjas på bästa sätt. Genom att välja $K = 24 + 15 * Y$ i TAB-instruktionen, kommer utskriften att ske mellan positionerna 9 och 39, eftersom $\sin x$ varierar mellan -1 och +1. I de första nio positionerna på TV-skärmen finns då plats kvar för att tabellera x-värdena.

Ett program, som plottar funktionen $y = \sin x$, kan se ut på följande sätt.

```
10 FOR X = 0 TO 6.4 STEP .4
20 LET Y = SIN(X)
30 PRINT X; TAB(24 + 15 * Y); "*"
40 NEXT X
```

Körning av programmet ger följande resultat.



Sinuskurva med hjälp av TAB-funktionen

Observera att de tänkta y-axeln pekar åt höger på TV-skärmen, medan den tänkta x-axeln pekar rakt nedåt.

PLOTTA FUNKTIONER MED PRINT CUR

Tyvärr kommer koordinataxlarna vid plottning med hjälp av TAB-instruktionen inte att få den riktning, som du förmodligen är van vid.

Med hjälp av instruktionen CUR(R, K) kan utskrift påbörjas vid önskad rad R och önskad kolumn K på skärmen. R och K får vara aritmetiska uttryck. Heltalsdelen för R får anta värden mellan 0 och 23, medan heltalsdelen för K får anta värden mellan 0 och 39.

Exempel.

```
10 PRINT CUR(11, 12); "MATEMATIK"
```

ger följande utskrift.

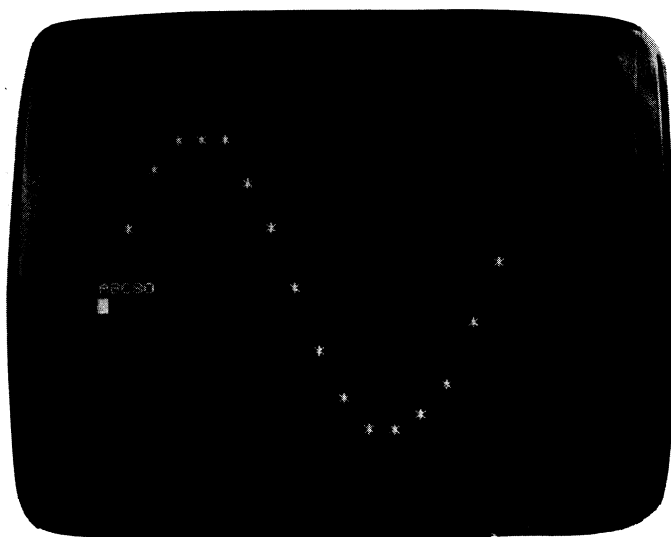
```
Rad 11 —————> MATEMATIK
                   ↑
                Kolumn 12
```


CUR-instruktionen kan användas för att plotta ett tecken, t ex "*", på en position av skärmen där R och K bestäms av y respektive x. Skalning måste givetvis ske, så att skärmen utnyttjas på bästa sätt.

Ett program som plottar funktionen $y = \sin x$ kan se ut så här.

```
10 PRINT CHR$(12)
20 FOR X = 0 TO 6.4 STEP .4
30 LET Y = SIN(X)
40 PRINT CUR(12 - 10 * Y, 5 * X); "*"
50 NEXT X
```

Körning av programmet ger följande resultat.

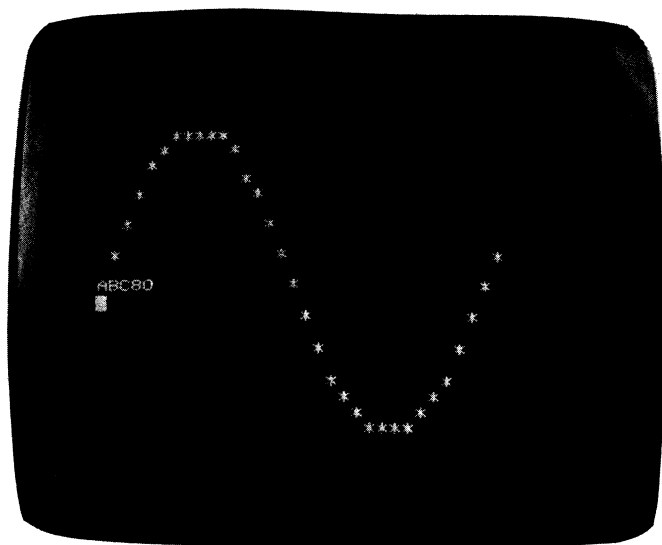


Sinuskurva med hjälp av CUR-funktionen

Ändra steglängden i programmet ovan från 0,4 till 0,2 d v s ändra rad 20 till

```
20 FOR X = 0 TO 6.4 STEP .2
```

En körning av programmet ger då följande resultat.



Steglängden påverkar kurvans utseende

Som framgår av resultatet ovan, blir sinuskurvan tillplattad i topparna. Detta beror på att upplösningen inte är tillräcklig. Studera även övningsexempel 9.7, där det visar sig att upplösningen heller inte räcker till för att ge en jämn och snygg kurva.

För att förbättra upplösningen vid plottning av kurvor, figurer etc, har ABC 80 möjlighet till en sk grafisk mod. Varje teckenposition delas då upp i 6 grafiska punkter. I kap 10 ska vi närmare studera den grafiska moden och plotta samma funktion som ovan, för att se hur upplösningen då förbättras.

SAMMANFATTNING

Instruktion	Kort beskrivning	Exempel
FOR...TO...STEP...	Ger möjlighet att definiera stegvärdet i en FOR...TO...-sats	10 FOR X=1 TO 10 STEP 2
DEF FN...(...)	Definition av egna funktioner	10 DEF FNA(X,Y) = X * X + Y

Funktion	Kort beskrivning	Exempel
SIN(X)	$\sin x$, x i radianer	10 Y = SIN(X)
COS(X)	$\cos x$, x i radianer	20 Y = COS(5 * X)
TAN(X)	$\tan x$, x i radianer	30 Y = TAN(X + 2)
ATN(X)	$\arctan x$ i radianer	40 Y = ATN(.5)
LOG(X)	e-logaritmen av x	50 Y = LOG(756)
LOG10(X)	10-logaritmen av x	60 Y = LOG 10(1000%)
EXP(X)	e^x	70 Y = EXP(2)
SQR(X)	Kvadratroten ur x , $\sqrt{x}$	80 Y = SQR(A ** 2 + B ** 2)
INT(X)	Största heltalet mindre än eller lika med x	90 Y = INT(Z)
FIX(X)	Heltalsdelen av x , $[x]$	100 Y = FIX(Z)
ABS(X)	Absolutbeloppet av x , $ x $	110 Y = ABS(-6)
SGN(X)	-1 om $x < 0$ 0 om $x = 0$ 1 om $x > 0$	120 Y = SGN(X)
PI	π , d v s talet 3,14159	130 X = PI * R ** 2
TAB(K)	Flyttar markören till position K på raden. (Förekommer endast i PRINT-satser)	140 PRINT TAB(10); "KOL 10"
CUR(R,K)	Flyttar markören till rad R och kolumn K på skärmen. (Förekommer endast i PRINT-satser)	150 PRINT CUR(11,12); "MITT PÅ SKÄRMEN"

ÖVNINGSEXEMPEL

9.1 Skriv ett program som beräknar värdet av uttrycket $x^5 + x^2 + 2$. Använd en INPUT-sats för att definiera olika x -värden.

9.2 Skriv ett program som ger utskrift i tabellform av samhörande värden på x , $\ln x$ och e^x där x antar heltalsvärden från 10 till och med 19.

- 9.3 Ändra steglängden för x från 1 till 0,5 i föregående övningsexempel.
- 9.4 Skriv ett program som med hjälp av en INPUT-sats definierar A och B samt beräknar $C = \sqrt{A^2 + B^2}$.
- 9.5 Skriv ett program som med hjälp av DEF FN-instruktionen definierar funktionen $f(x) = \sqrt{x} \cdot e^{-x^2}$ och därefter beräknar $f(x)$ för $x = 1.7, 2.25$ och 2.5
- 9.6 Skriv ett program som med hjälp av en INPUT-sats beräknar $\sqrt{x}$. Resultatet ska avrundas till två decimaler.
- 9.7 Skriv ett program som med hjälp av PRINT CUR plottar funktionen $y = \arctan x$ där x ska variera mellan -10 och $+10$. Försök att utnyttja skärmen så bra som möjligt.
- 9.8 Funktionen ABS(X) ger absolutvärdet av X . Byt ut Y på rad 40 i programmet på sidan 111 mot ABS(Y). Vad blir resultatet vid körning av programmet? Förklara!

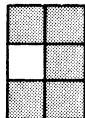
KAPITEL 10

Grafisk presentation på bildskärmen

ABC80

GRAFISK MOD

Med ABC 80 kan du på TV-skärmen rita figurer, grafiska bilder och diagram genom att utnyttja möjligheten att de flesta ASCII-tecken har en grafisk motsvarighet. De grafiska tecknen är uppbyggda av ett antal vita fält ordnade enligt nedanstående princip.



Grafisk motsvarighet till tecknet "ä" (skuggat fält motsvarar vita fält på TV-skärmen).

ASCII-TABELL MED GRAFISKA TECKEN

Tabellen på nästa sida visar den grafiska motsvarigheten till olika ASCII-tecken. Som du märker finns inga grafiska motsvarigheter till stora bokstäver. Detta innebär att stora bokstäver inte påverkas i grafisk mod utan skrivs oförändrade, trots att en rad är försatt i grafisk mod.

SPECIALTECKEN FÖR GRAFISK MOD

För att bestämma om ett tecken skall visas på normalt sätt eller med sin grafiska motsvarighet, finns två specialtecken som styr den grafiska moden.

CHR ␣ (151) ger tecknet för "START GRAFIK"

CHR ␣ (135) ger tecknet för "SLUT GRAFIK"

I vanliga fall skrivs förstås tecknen på normalt sätt, men genom att man skriver det "osynliga" tecknet CHR ␣ (151) på en rad, förändras all utskrift efter CHR ␣ (151) till resp teckens grafiska motsvarigheter. Följande exempel förklarar principen.

```
10 PRINT "123ABC"; CHR ␣ (151); "123ABC"
```

Utskriften efter köring

123ABC  ABC

För att återställa en rad som redan försatts i grafisk mod, används tecknet för "SLUT GRAFIK", CHR ␣ (135).

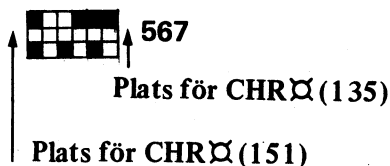
Kod	T	G	Kod	T	G	Kod	T	G	Kod	T	G
32	Blank		56	8		80	P	P	104	h	
33	!		57	9		81	Q	Q	105	i	
34	"		58	:		82	R	R	106	j	
35	#		59	;		83	S	S	107	k	
36	¤		60	<		84	T	T	108	l	
37	%		61	=		85	U	U	109	m	
38	&		62	>		86	V	V	110	n	
39	'		63	?		87	W	W	111	o	
40	(		64	É	É	88	X	X	112	p	
41	)		65	A	A	89	Y	Y	113	q	
42	*		66	B	B	90	Z	Z	114	r	
43	+		67	C	C	91	Ä	Ä	115	s	
44	,		68	D	D	92	Ö	Ö	116	t	
45	-		69	E	E	93	À	À	117	u	
46	.		70	F	F	94	Ü	Ü	118	v	
47	/		71	G	G	95	-	-	119	w	
48	0		72	H	H	96	é		120	x	
49	1		73	I	I	97	a		121	y	
50	2		74	J	J	98	b		122	z	
51	3		75	K	K	99	c		123	ä	
52	4		76	L	L	100	d		124	ö	
53	5		77	M	M	101	e		125	å	
54	6		78	N	N	102	f		126	ü	
55	7		79	O	O	103	g		127	■	

ASCII-tecken tolkade i teckenmod (T) och grafmod (G)

10 PRINT CHR\$(151); "123"; CHR\$(135); "567"

Ovanstående program börjar med att skriva tecknet för "START GRAFIK" på raden och försätter därmed hela raden i grafisk mod. Strängen "123" skrivs därför ut med sin grafiska motsvarighet. Därefter skrivs tecknet för "SLUT GRAFIK" i nästa position och återställer resten av raden till normal utskrift.

Utskrift efter körning.



HELA SKÄRMEN I GRAFISK MOD

Ett program som försätter hela skärmen i grafisk mod kan se ut på följande sätt.

```
10 PRINT CHR$(12)
20 FOR I = 0 TO 23
30 PRINT CUR(I,0); CHR$(151);
40 NEXT I
```

Detta program medför att datorn skriver tecknet "START GRAFIK" som första tecken på skärmens 24 rader.

ENDAST FANTASIN SÄTTER GRÄNS

När du med ett program enligt ovan försatt hela skärmen i grafisk mod, kan du med hjälp av funktionen CUR(R,K) flytta markören till lämplig plats och där skriva önskade grafiska tecken. Tillfoga följande rader, så erhåller du ett program som ritar 6 rutor på TV-skärmen.

```
50 FOR A = 5 TO 30 STEP 5
60 PRINT CUR (10,A); "wä"
70 NEXT A
```


På den demonstrationskassett som följer med ABC 80, finns flera program-exempel som visar olika möjligheter till grafisk presentation.

Om man sätter hela skärmen i grafisk mod får man 72 grafiska rader (R), numrerade 0 - 71, och 78 grafiska positioner (K), numrerade 2 - 79, på varje grafisk rad. Med instruktionen SETDOT R,K kan du tända en godtycklig grafisk punkt. Kontrollera att nedanstående program tänder fyra punkter på TV-skärmen.

```
10 PRINT CHR$(12)
20 FOR I = 0 TO 23
30 PRINT CUR(I,0); CHR$(151);
40 NEXT I
50 SETDOT 30, 30
60 SETDOT 30, 40
70 SETDOT 40, 30
80 SETDOT 40, 40
```

Tecknet "START GRAFIK", CHR\$(151), står fortfarande som första tecken på varje rad. Detta tecken är, som vi påpekat tidigare, osynligt på TV-skärmen men får ej skrivas över. Du kan därför bara använda de grafiska positionerna 2 till 79 på varje rad.

Ytterligare en instruktion finns, som enbart kan användas i grafisk mod, nämligen CLRDOT R,K som släcker en grafisk punkt. Det finns även en villkorsfunktion, DOT(R,K), som testar om en viss grafisk punkt är släckt eller ej. Nedanstående programrad släcker punkten R,K om den är tänd och tänder den om den är släckt.

```
110 IF DOT (R,K) THEN CLRDOT R, K ELSE SET
DOT R, K
```

IF... THEN... ELSE... innebär att om villkoret efter IF är uppfyllt, så utförs instruktionen efter THEN. Om villkoret däremot inte är uppfyllt, utförs istället instruktionen efter ELSE.

Tillfoga raderna

```
90 FOR R = 20 TO 50
100 FOR K = 20 TO 50
120 NEXT K
130 NEXT R
```

Hela programmet ser nu ut så här.

```
10 PRINT CHR$(12)
20 FOR I = 0 TO 23
30 PRINT CUR(I,0); CHR$(151);
40 NEXT I
50 SETDOT 30, 30
60 SETDOT 30, 40
70 SETDOT 40, 30
80 SETDOT 40, 40
90 FOR R = 20 TO 50
100 FOR K = 20 TO 50
110 IF DOT (R, K) THEN CLRDOT R,K ELSE SET
DOT R, K
120 NEXT K
130 NEXT R
```

Kör programmet och förklara funktionen!

KURVRITNING MED BÄTTRE UPPLÖSNING

Vi har i kapitel 9 "ritat" kurvor med funktionen CUR(R,K). Maximal upplösning blev då 24 positioner i Y-led och 40 positioner i X-led. Genom att istället använda funktionen SETDOT R,K kan vi få betydligt bättre upplösning, nämligen 72 positioner i Y-led och 78 i X-led.

När instruktionen SETDOT R,K används, måste naturligtvis skalning ske på motsvarande sätt som när funktionen CUR(R,K) användes för att rita kurvor.

SINUSKURVA I GRAFISK MOD

Nedanstående program visar hur en sinuskurva kan ritas på TV-skärmen.

```
10 PRINT CHR$(12)
20 FOR I = 0 TO 23
30 PRINT CUR(I,0); CHR$(151);
40 NEXT I
50 FOR X = 0 TO 6.4 STEP .1
60 R = 35 - 30 * SIN(X)
70 K = 2 + 10 * X
80 SETDOT R,K
90 NEXT X
100 GOTO 100
```

Kör programmet. Programmet innehåller en oändlig loop och måste alltså avbrytas med tangenterna **CTRL** och **C** .

Radnummer	Radens funktion
10	Raderar TV-skärmen.
20 - 40	Hela skärmen försätts i grafisk mod.
50, 90	Definierar en loop, där loopvariabeln x varierar från 0 till 6.4 med steget 0.1.
60	Skalar sin x på lämpligt sätt i y-led.
70	Skalar x på lämpligt sätt i x-led.
80	Ger en grafisk punkt.
100	Medför hopp tillbaka till rad 100, d v s ger en oändlig loop som eliminerar utskrift av texten ABC 80.

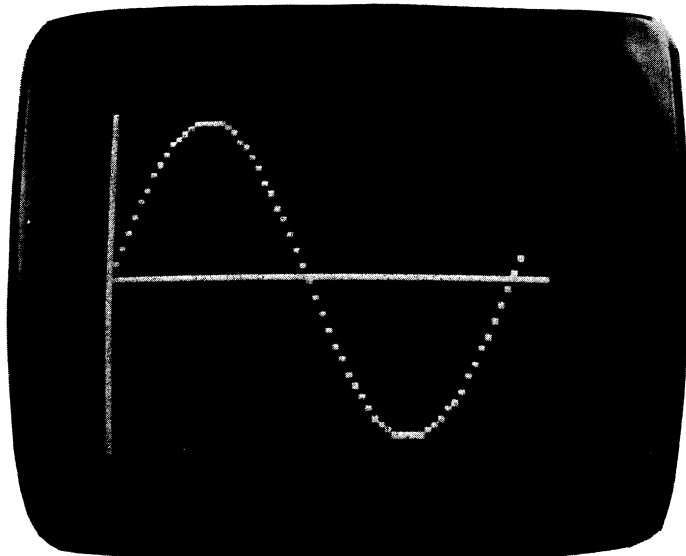
SINUSKURVA MED AXLAR

Vill man förse en kurva med axlar är det lätt att utöka programmet med två loopar som ritar x- resp y-axel.

```
41 FOR R = 3 TO 67
42 SETDOT R,2
43 NEXT R
44 FOR K = 2 TO 70
45 SETDOT 35,K
46 NEXT K
```

Kom ihåg att man inte får skriva i de första två grafiska positionerna på varje rad. Där står ju tecknet för "START GRAFIK".

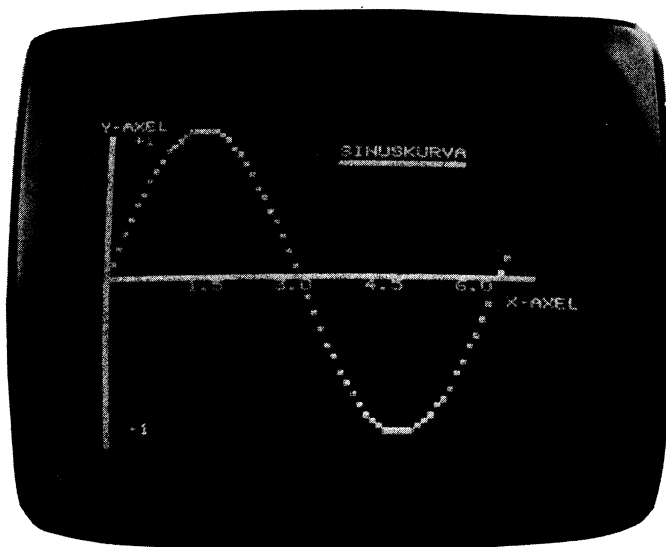
Efter körning av programmet erhålls följande bild på TV-skärmen.



Sinuskurva med axlar

SINUSKURVA MED GRADERADE AXLAR

Om man så önskar, kan man med hjälp av lämpliga utskrifter dels gradera koordinataxlarna, dels förse grafen med förklarande text. Ett exempel visas i nedanstående bild.



Sinuskurva med graderade axlar

För att erhålla ovanstående utskrift har följande programrader tillfogats.

```
100 REM ** UTSKRIFT **
110 PRINT CUR (1, 0); "Y-AXEL"; CHR 151
120 PRINT CUR (13,32); CHR 151; "X-AXEL"
130 PRINT CUR (3,20); "SINUSKURVA"
140 PRINT CUR (4,20); "#####"
150 PRINT CUR (12,7); CHR 151; "1.5 3.
0 4.5 6.0"; CHR 151
160 PRINT CUR (2,2); CHR 151; " + 1 "; CHR 151 (1
51)
170 PRINT CUR (21,2); CHR 151; " - 1 "; CHR 151 (
151)
180 GOTO 180
```

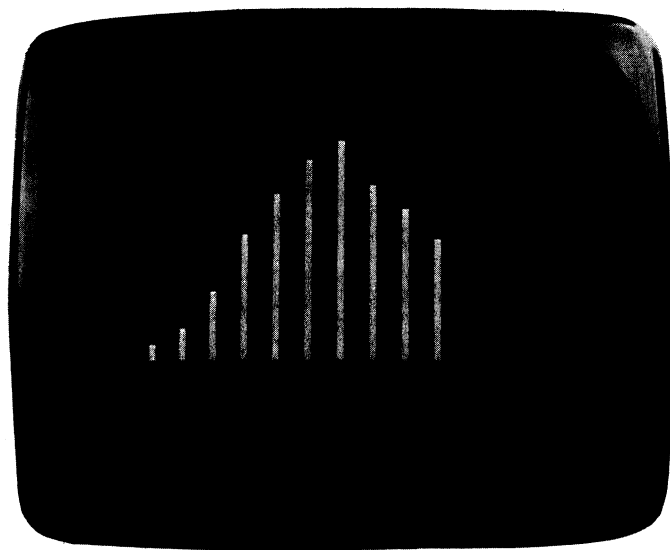
Fundera igenom dessa programrader noga. Vid blandning av grafisk och normal utskrift har man stor nytta av den "videografikkarta", som finns i slutet på kapitlet.

STAPELDIAGRAM

Nedanstående program visar hur man kan rita ett stapeldiagram med hjälp av instruktionen SETDOT R,K. Förutsättningen är att 10 värden mellan 0 och 50 skall representeras i ett stapeldiagram.

```
10 PRINT "MATA IN TIO VÄRDEN MELLAN"
20 PRINT "0 och 50 ÅTSKILDA AV RETURN"
30 FOR N = 1 TO 10
40 INPUT A(N)
50 NEXT N
60 PRINT CHR 12
70 FOR I = 0 TO 23
80 PRINT CUR(I,0); CHR 151;
90 NEXT I
100 FOR K = 10 TO 55 STEP 5
110 X = X + 1
120 FOR R = 50 TO 50 - A(X) STEP -1
130 SETDOT R,K
140 NEXT R
150 NEXT K
160 GOTO 160
```

Resultatet av en körning kan se ut så här.



Stapeldiagram

Förse programmet med textkommentarer genom att tillfoga lämpliga PRINT-satser.

SAMMANFATTNING

Instruktion	Kort beskrivning	Exempel
IF...THEN...ELSE...	Om uttrycket efter IF är sant så utförs det som står mellan THEN och ELSE, annars utförs det som står efter ELSE. Om det bara står ett radnummer efter THEN eller efter ELSE utförs hopp till detta radnummer. ELSE kan utelämnas.	50 IF A = B THEN PRINT A ELSE GOTO 300 60 IF A = B THEN 90 ELSE 200 70 IF A = B THEN 50

SETDOT R,K	Tänder en grafisk punkt på skärmen. (Raden måste vara satt i grafisk mod.) $0 \leq R \leq 72$ $2 \leq K \leq 79$	30 SETDOT 30,40
CLRDOT R,K	Släcker en grafisk punkt på skärmen.	40 CLRDOT 30,40
Villkorsuttryck DOT(R,K)	Kort beskrivning Ger värdet SANT om den grafiska punkten R,K är tänd, annars värdet FALSKT.	Exempel 30 IF DOT(30,40) THEN CLRDOT 30,40 ELSE SETDOT 30,40
Specialtecken CHR⌘(151)	Kort beskrivning "START GRAFIK" startar grafisk mod på en rad.	Exempel 10 PRINT CHR⌘(151); "1234"
CHR⌘(135)	Kort beskrivning "SLUT GRAFIK" avslutar grafisk mod på raden.	Exempel 10 PRINT CHR⌘(151); "1234"; CHR⌘(135); "1"

ABQ

VIDEOGRAFIK

PROGRAM

lin	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23
dot	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23
0	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23
1	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23
2	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23
3	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23
4	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23
5	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23
6	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23
7	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23
8	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23
9	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23
10	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23
11	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23
12	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23
13	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23
14	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23
15	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23
16	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23
17	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23
18	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23
19	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23
20	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23
21	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23
22	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23
23	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23

= plats för CHR = (151) = "START GRAFIK"

ÖVNINGSEXEMPEL

- 10.1 Skriv ett program som sätter hela TV-skärmen i grafisk mod, och därefter ritar ett cirka 6 cm långt minustecken mitt på TV-skärmen. Förändra därefter programmet så att ett plustecken av motsvarande storlek presenteras.
- 10.2 Nedanstående program presenterar ASCII-tecknen från nummer 32 till nummer 127. Komplettera programmet så att även det aktuella ASCII-tecknets grafiska motsvarighet presenteras på samma rad.

```
10 FOR A = 32 TO 127
20 PRINT
30 PRINT CHR$(A)
40 FOR X = 1 TO 1000 : NEXT X
50 NEXT A
```

- 10.3. Modifiera programmet på sidan 123 som ritar stapeldiagram, så att staplarna presenteras svarta på vit botten.
- 10.4 Skriv ett program som ritar en ram på TV-skärmen genom tända lämpliga grafiska positioner med SETDOT R,K. Försök att få så stor ram som möjligt.
- 10.5 En utökning av programmet från exempel 10.4 presenteras nedan. Försök att analysera programmet sedan du provkört det.

```
130 X1 = 1 : Y1 = 1 : X = 2 : Y = 0
140 FOR A = 1 TO 5500
150 X = X + X1 : Y = Y + Y1 : SETDOT Y,X
160 IF X = 79 THEN X1 = -1
170 IF X = 2 THEN X1 = 1
180 IF Y = 71 THEN Y1 = -1
190 IF Y = 0 THEN Y1 = 1
200 NEXT A
210 FOR A = 1 TO 5500
220 X = X + X1 : Y = Y + Y1 : CLRDOT Y,X
230 IF X = 79 THEN X1 = -1
240 IF X = 2 THEN X1 = 1
250 IF Y = 71 THEN Y1 = -1
260 IF Y = 0 THEN Y1 = 1
270 NEXT A : GOTO 10
```

- 10.6 Nedanstående program visar en möjlighet att åstadkomma enkel rörelse på TV-skärmen.

```
10 PRINT CHR$(12)
20 FOR I = 0 TO 23
30 PRINT CUR(I, 0); CHR$(151);
40 NEXT I
50 FOR X = 20 TO 40
60 SETDOT X, 29
70 SETDOT X, 61
80 NEXT X
90 K = 30 : A = 1
100 R = 30 : K = K + A
110 SETDOT R, K
120 FOR X = 1 TO 20 : NEXT X
130 CLRDOT R, K
140 FOR X = 1 TO 20 : NEXT X
150 IF K = 60 THEN A = -1
160 IF K = 30 THEN A = 1
170 GOTO 100
```

Hur kan du förändra hastigheten hos den rörliga punkten?

- 10.7 Försök skriva ett program som kan användas som inbrottslarm hos kött och fläskhandlaren Frick. Programmet skall ge följande utskrift på TV-skärmen.

**Tryck på tangenten Ö
så öppnas kassaskåpet.**

När så den hemska inbrottstjuven under natten närmar sig ABC 80, kan han givetvis inte låta bli att trycka på tangenten Ö. Programmet skall då aktivera ljudgeneratoren och skriva ut lämplig brottsförebyggande text på TV-skärmen.

KAPITEL 11

Lagra data på kassett

ABC80

DATA I DATASATSER

Data som behövs i ett program lagras ofta i DATA-satser. Dessa data läses med instruktionen READ. Om programmet ska användas i skilda sammanhang, med olika data vid varje körning, kan man lagra programmet i olika versioner på kassettband.

Är programmet omfattande är det opraktiskt att lagra programmet i sin helhet i olika upplagor på band. Det räcker ju att lagra de programrader som byts ut mellan varje körning. Låt oss se på ett programexempel

```
10 PRINT CHR$(12)
20 READ A$
30 PRINT "PRISLISTA", A$
40 PRINT
50 FOR K = 1 TO 3 : READ P(K) : NEXT K
60 PRINT "SPIK", P(1)
70 PRINT "SPADE", P(2)
80 PRINT "TRAKTOR", P(3)
90 REM *****
100 DATA BRUTTO
110 DATA 20, 150, 93 000
```

Detta program skriver ut en prislista för ett mycket begränsat sortiment. Provkör programmet!

Förbered bandspelaren för inspelning och lagra programmet på kassett med

SAVE PRISLIST

När man vill skriva ut en nettoprislista för samma sortiment, behöver enbart DATA-satserna på raderna 100 och 110 ändras. Radera föregående program med NEW och mata in de nya DATA-satserna via tangentbordet. De nya DATA-satserna är

```
100 DATA NETTO
110 DATA 18, 140, 90 000
```

Lagra dessa DATA-satser på kassettbandet efter programmet PRISLIST med

SAVE NETTO

Skriv ytterligare en uppsättning DATA-satser

100 DATA ÅTERFÖRSÄLJARPRIS
110 DATA 17, 135, 89 000

Lagra dessa rader på kassett med

SAVE ÅTERFÖRS

Nu ska de tre olika prislistora skrivas ut. Spola tillbaka bandet, ställ in bandspelaren för avspelning och skriv ut den första prislistan med

RUN PRISLIST



Bruttoprislista

När nettoprislistan ska presenteras byts först DATA-satserna i programmet. De nya DATA-satserna överförs från band med kommandot

MERGE NETTO

Kör som vanligt med RUN.



Nettoprislista

Överför sedan återförsäljarpriserna från bandet med

MERGE ÅTERFÖRS

Kör sedan programmet på vanligt sätt.



Återförsäljarprislista

Utöka gärna programmet, så att priserna för ett större sortiment kan lagras. Kom ihåg att reservera plats med en DIM-sats, om antalet artiklar överstiger 10.

DATAFILER SPARAR UTRYMME

Ett alternativ till att lagra data i form av DATA-satser är att använda sig av s k datafiler, som ibland också kallas textfiler. Man skapar då ett utrymme på kassetten där man kan skriva data nästan som på TV-skärmen. Dessa data kan sedan hämtas från bandet ungefär som om de hämtades från tangentbordet.

Instruktionen PREPARE ... ASFILE ... används för att skapa ett sådant utrymme på bandet. Det namn, man vill ge sin datafil, placeras mellan PREPARE och ASFILE. Namnet ska omges av citationstecken. Man ska också ge sin datafil ett nummer. Detta skrivs efter ASFILE i instruktionen.

Exempel.

10 PREPARE "DATA" ASFILE 1

När raden 10 utförs skapas på kassetten en fil med namnet DATA och numret 1. Datorn startar då bandspelaren och skriver filens namn på bandet. För att skriva data på filen använder du instruktionen PRINT med tillägg av siffer-tecknet (#) samt filens nummer och ett kommatecken, t ex PRINT # 1, . Kom ihåg att tecknet # anger att skrivningen sker på en datafil.

Exempel.

```
20 PRINT # 1, "BRUTTO"
```

Vill du skriva flera data på filen fyller du på med ytterligare satser, t ex

```
30 PRINT # 1, "20"
```

```
40 PRINT # 1, "150"
```

```
50 PRINT # 1, "93 000"
```

När filen inte ska rymma flera data måste filens slut markeras och bandspelaren stoppas. Det gör du med instruktionen CLOSE som åtföljs av filens nummer, t ex

```
60 CLOSE 1
```

Ett komplett program som skapar en datafil och skriver några data på filen kan alltså se ut så här.

```
10 PREPARE "DATA" ASFILE 1
```

```
20 PRINT # 1, "BRUTTO"
```

```
30 PRINT # 1, "20"
```

```
40 PRINT # 1, "150"
```

```
50 PRINT # 1, "93 000"
```

```
60 CLOSE 1
```

DATA ÖVERFÖRS TILL BAND

Ställ in bandspelarens kontroller för inspelning, nollställ räkneverket och kör programmet. När datorn har skapat filen och skrivit aktuella data, samt avslutat filen, skrivs som vanligt ABC 80 på skärmen.

Den instruktion som förbereder läsning från datafil är OPEN ... ASFILE ... Mellan OPEN och ASFILE anges inom citationstecken namnet på den datafil som man vill läsa ifrån. Efter ASFILE anges ett filnummer.

Exempel.

```
15 OPEN "DATA" ASFILE 1
```

När rad 15 utförs startas bandspelaren automatiskt och datorn börjar söka efter en datafil med namnet DATA. För att läsa data från filen används instruktionen INPUT med tillägg av siffertecken, filnummer och kommatecken, t ex INPUT # 1, .

Två exempel.

```
21 INPUT # 1, A$  
25 INPUT # 1, P(1), P(2), P(3)
```

För att avsluta läsningen på filen och stoppa bandspelaren använder du instruktionen CLOSE, åtföljd av filnummer, t ex

```
27 CLOSE 1
```

För att kontrollera att avsedda data överförs från bandet tillfogar du raden

```
28 PRINT A$, P(1), P(2), P(3)
```

Det fullständiga programmet för att läsa data från en fil är således

```
15 OPEN "DATA" ASFILE 1  
21 INPUT # 1, A$  
25 INPUT # 1, P(1), P(2), P(3)  
27 CLOSE 1  
28 PRINT A$, P(1), P(2), P(3)
```

Provkör programmet. Glöm inte att bandet dessförinnan måste vara återspolat och bandspelaren vara inställd för avspelning.

Kombinera detta program med programmet PRISLIST i kapitlets början genom att spola tillbaka bandet och ge kommandot

```
MERGE PRISLIST
```

Avlägsna dataraderna 90, 100 och 110 samt raderna 20, 28 och 50. När du använder LIST ska det resulterande programmet bli

```
10 PRINT CHR$ (12)  
15 OPEN "DATA" ASFILE 1  
21 INPUT # 1, A$  
25 INPUT # 1, P(1), P(2), P(3)  
27 CLOSE 1  
30 PRINT "PRISLISTA", A$  
40 PRINT  
60 PRINT "SPIK", P(1)  
70 PRINT "SPADE", P(2)  
80 PRINT "TRAKTOR", P(3)
```

När du kör programmet söker datorn upp datafilen DATA och skriver sedan ut bruttoprislistan precis som i kapitlets första exempel.

ETT ANVÄNDARVÄNLIGT PROGRAM

I detta exempel, som inte är alltför allvarligt menat, ska regementsläkaren registrera sina mätningar av de värnpliktigas midjevidd och kroppslängd. Dessa värdefulla mätresultat kan givetvis också användas av andra intressenter t ex förrådsmanen som ska se till att de värnpliktiga får användbara kläder. I köket kan man förstås med osviklig precision beräkna matförbrukningen med ledning av dessa uppgifter, o s v.

Studera flödesschemat och försök gärna konstruera de programavsnitt som ingår. I det färdiga programmet är de olika avsnitten åtskilda med ett mellanrum för att underlätta jämförelsen mellan flödesschema och färdigt program. Intill flödesschemats symboler anges numren på de programrader som utgör motsvarande programavsnitt.

```
10 DIM L(100) : DIM M(100)

20 PRINT CHR$(12)

30 PRINT "ANGE INMATNING (I) ELLER SAMMA
NSTÄLLNING (S)";

40 GET A$: IF A$="S" THEN GOTO 330

50 PRINT
60 PRINT "OBS! BANDSP. INST. FÖR INSP. (J)
?";

70 GET A$
80 IF A$="J" THEN GOTO 90 ELSE GOTO 60

90 PRINT CHR$(12)

100 PRINT : PRINT
110 I = I + 1
120 PRINT "ANGE UPPMÄTT LÄNGD I CM";
130 INPUT L(I)
140 PRINT "UPPMÄTT MIDJEVIDD I CM";
150 INPUT M(I)
160 PRINT : PRINT

170 PRINT "YTTERLIGARE DATA (J EL N)"

180 GET A$
190 IF A$="J" THEN GOTO 90
```

200 PRINT CHR\$(12)

210 PRINT "DATA ÖVERFÖRS TILL BANDET"

220 PREPARE "FA" ASFILE 1

230 FOR X = 1 TO I

240 PRINT #1, L(X)

250 PRINT #1, M(X)

260 NEXT X

270 CLOSE 1

280 PRINT : PRINT "ÖVERFÖRINGEN ÄR KLAR"

290 PRINT "ÖNSKAS EN SAMMANSTÄLLN. (J EL
N)";

300 GET A\$

310 PRINT

320 IF A\$="J" THEN GOTO 350 ELSE END

330 PRINT "HUR MÅNGA PERSONER";

340 INPUT I

350 PRINT "BANDET ÅTERSPOLAT (J)";

360 GET A\$

370 PRINT

380 IF A\$="J" THEN GOTO 390 ELSE GOTO 350

390 PRINT

400 PRINT "BANDSP. PÅ AVSP. (J)";

410 GET A\$

420 PRINT

430 IF A\$="J" THEN GOTO 440 ELSE GOTO 400

440 PRINT

450 PRINT "LÅNGD", "MIDJEVIDD"

460 PRINT

470 OPEN "FA" ASFILE 1

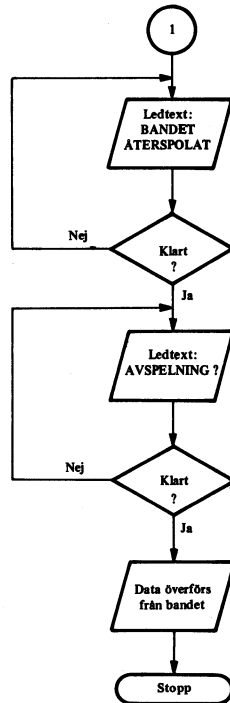
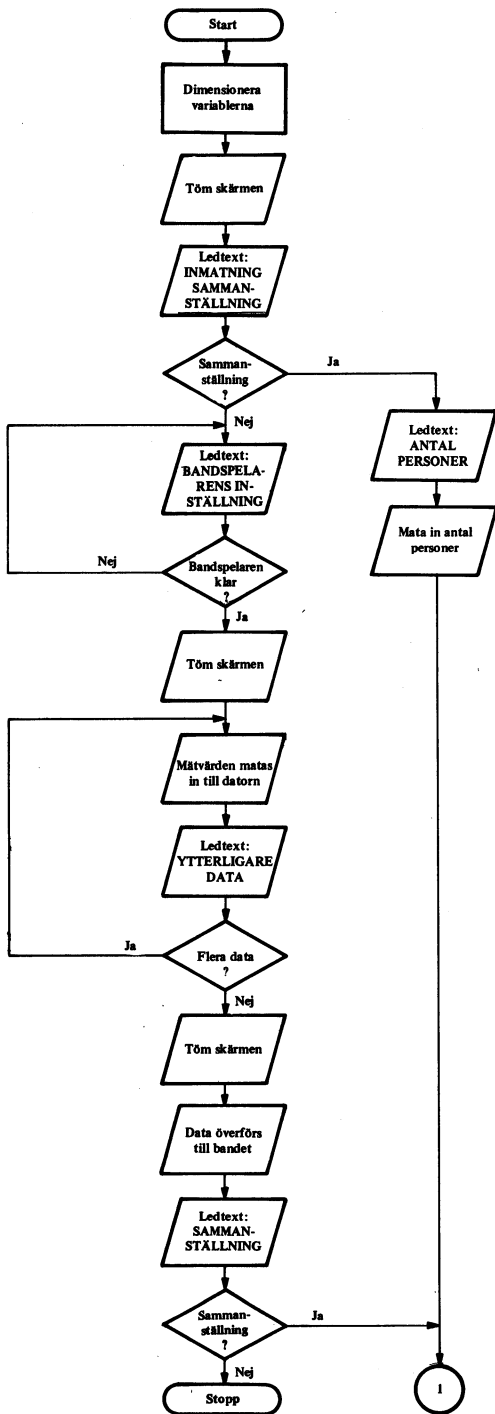
480 FOR T = 1 TO I : INPUT #1, L(T)

490 INPUT #1, M(T)

500 PRINT L(T), M(T)

510 NEXT T : CLOSE 1

520 END



SAMMANFATTNING

Instruktion	Kort beskrivning	Exempel
PREPARE ... ASFILE ...	Skapar en ny fil med visst nummer från 1 till 255. Bandspelare startas.	10 PREPARE "BREV 1. TXT" ASFILE 2
OPEN ... ASFILE ...	Tilldelar en befintlig fil visst nummer från 1 till 255. Bandspelare startas.	20 OPEN "BREV 1. TXT" ASFILE 2
CLOSE	Stänger en fil, d v s avslutar läsning eller skrivning mot filen Vid skrivning skrivs filslut. Bandspelare stoppas.	50 CLOSE 2
PRINT # ... ,	Skriver till en fil på kassett.	30 PRINT # 1,X
INPUT # ... ,	Matar in data från fil på kassett.	40 INPUT # 1,Y
INPUTLINE # ... ,	Läser in en rad från en fil på kassett.	90 INPUTLINE # 1,A

Filnamn kan vara upp till 8 tecken långa, eventuellt med ett tillägg, t ex .TXT, som används för datafiler.

ÖVNINGSEXEMPEL

- 11.1 Skriv ett program som läser 20 tal från tangentbordet och sedan skriver dessa värden på en datafil.
- 11.2 Skriv ett program som läser de 20 talen från den datafil som skapades i föregående exempel och skriver talen på TV-skärmen.

KAPITEL 12

ASCII-aritmetik

ABC80

HELTALSVARIABLER

Vi har hittills inte närmare undersökt skillnaden mellan att representera ett tal i form av en flyttalsvariabel och i form av en heltalsvariabel. Låt oss titta på ett exempel där skillnaden framgår tydligt.

```
10 FOR X = 1 TO 10000 : NEXT X  
20 PRINT "KLAR"
```

Kör ovanstående program och mät den tid som åtgår för programkörningen. Tiden blir ca 10 sek. Ändra därefter programmet så att loopvariabeln blir en heltalsvariabel och mät motsvarande tid på nytt.

```
10 FOR X% = 1% TO 10000% : NEXT X%  
20 PRINT "KLAR"
```

Tiden blir i detta fall ca 2 sek. Programkörningen gick således mycket snabbare när vi använde en heltalsvariabel som loopvariabel. Utan att närmare gå in på rent tekniska förklaringar kan vi konstatera att program utförs snabbare när heltalsvariabler används. När det användas talområdet (-32768 till 32767) så medger, bör därför heltalsvariabler användas i stället för flyttalsvariabler. I rad 10 ingår även heltalskonstanter. Eftersom heltalskonstanter endast utnyttjar hälften så stort minnesutrymme som flyttalskonstanter, bör heltalskonstanter användas i möjligaste mån.

FLYTTALSVARIABLER

Flyttalsvariablernas talområde från $\pm 0.1\text{E}-127$ till $\pm 0.999999\text{E}+127$ är tillräckligt stort för de flesta tillämpningar. Däremot kan, som vi sett, stora eller små tal bara anges med max. sex siffrors noggrannhet. Detta är vid många beräkningar inte tillräckligt. I vissa tillämpningar, t ex momsberäkningarna i kap. 4, var det å andra sidan besvärande med för många decimaler i resultatutskriften. I ABC 80 kan vi komma tillrätta med dessa brister hos flyttalsvariablerna genom att använda numeriska strängar för beräkningar.

NUMERISKA STRÄNGAR

ABC 80 kan utföra beräkningar med stor noggrannhet (upp till max. 29 siffror) med hjälp av den så kallade ASCII-aritmetiken. ASCII-aritmetiken utnyttjar strängar som kan tolkas som siffror. Dessa kallas numeriska strängar. En numerisk sträng får innehålla tecknen + eller -, siffror samt ev decimalpunkt. En numerisk sträng får däremot inte innehålla exponentsymbolen E. Exponentnotation är alltså inte tillåten.

Exempel på numeriska strängar.

-123456789
2003.456
-435.39421076

Numeriska strängvariabler har namn som vanliga strängvariabler, och behandlas på samma sätt. Eftersom det enbart är siffror du ska mata in, behöver strängarna inte omges med citationstecken.

ASCII-ARITMETIK

De instruktioner som kan användas tillsammans med de numeriska strängarna, är addition, subtraktion, multiplikation och division, samt numerisk jämförelse av två numeriska strängar.

Instruktionerna **ADD(A, B, N)**, **SUB(A, B, N)**, **MUL(A, B, N)** och **DIV(A, B, N)** adderar, subtraherar, multiplicerar respektive dividerar de numeriska strängarna **A** och **B**, tolkade som tal. Svaret avrundas till **N** decimaler.

Instruktionen **COMP(A, B)** jämför de numeriska strängarna **A** och **B**, tolkade som tal. Resultatet blir -1 om talet i **A** är mindre än talet i **B**, 0 om talet i **A** är lika med talet i **B**, 1 om talet i **A** är större än talet i **B**.

Följande program kan användas för beräkningar med de fyra räknesätten i ASCII-aritmetik. Vi använder ; som förkortning för PRINT i följande program. Detta är praktiskt i program med många PRINT-satser och är en speciell finess i ABC 80.

```
10 ;CHR(12)
20 ;"ADDITION (1)"
30 ;"SUBTRAKTION (2)"
40 ;"MULTIPLIKATION (3)"
50 ;"DIVISION (4)"
60 ;
70 ;"VILKET RÄKNESÄTT ÖNSKAS";
80 INPUT I
90 ;
100 ;"VILKA TAL A OCH B";
110 INPUT A, B
120 ;
```

```

130 ; "HUR MÅNGA DECIMALER I SVARET"
140 INPUT N
150 ;
160 ON I GOSUB 180, 190, 200, 210
170 GOTO 60
180 ; "SUMMAN =" ; ADD(A, B, N) : RETURN
190 ; "SKILLNADEN =" ; SUB(A, B, N) : RETURN
200 ; "PRODUKTEN =" ; MUL(A, B, N) : RETURN
210 ; "KVOTEN =" ; DIV(A, B, N) : RETURN

```

Vilket räknesätt som ska utföras på de numeriska strängarna väljs med en INPUT-sats (radnummer 80) där 1 = addition, 2 = subtraktion, 3 = multiplikation och 4 = division.

De numeriska strängarna A och B tilldelas värden med nästa INPUT-sats. På radnummer 160 finns en ny instruktion, som du inte träffat på tidigare. Instruktionen ON I GOSUB 200, 300, 400, 500 åstadkommer ett hopp till subrutiner för respektive räknesätt på radnummer 200, 300, 400 eller 500. Den rad, till vilken hoppet sker, beror av värdet på I.

Resultatet av en beräkningsoperation som utförts med ASCII-aritmetik får aldrig överstiga 29 siffror. Noggrannheten vid beräkningar med ASCII-aritmetik varierar, men är normalt större än tjugo siffror.

ON-SATSER

Satsen ON... GOSUB... används när man vill kunna hoppa till olika subrutiner i programmet, beroende på värdet av en variabel eller ett uttryck.

ON I GOSUB 200, 300, 400 åstadkommer hopp till radnummer 200 om I = 1, till radnummer 300 om I = 2 och till radnummer 400 om I = 3. Alla andra värden på I ger felutskrift.

Övriga ON-satser som finns i ABC 80-BASIC är ON... GOTO..., ON... RESTORE... och ON ERROR GOTO...

ON... GOTO används när man vill kunna göra hopp till flera olika ställen i programmet. Den fungerar på samma sätt som ON... GOSUB.

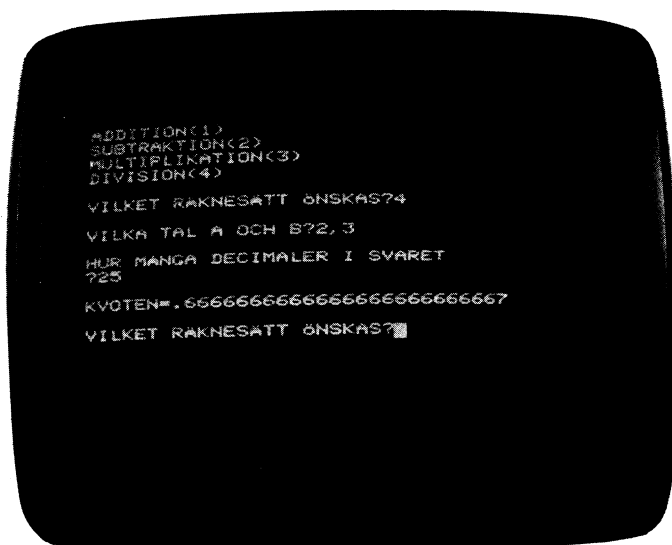
RESTORE innebär att nästa READ-sats återgår till första DATA-satsen för att hämta DATA. ON... RESTORE... används när man vill kunna hämta data från och med bestämda DATA-satser.

ON I RESTORE 200, 300, 400 innebär att datorn börjar hämta data från de DATA-satser som den återfinner från och med radnummer 200 om $I = 1$, från och med radnummer 300 om $I = 2$ o s v.

ON ERROR GOTO... ger ett hopp till angivet radnummer om fel skulle uppstå under programkörningen. Du kommer att lära dig mer om användningen av denna instruktion i kap. 13.

KORREKT AVRUNDNING

När du använder dig av ASCII-aritmetik, kommer svaret alltid att avrundas korrekt. Skriv in programmet ovan och använd det för att beräkna $2/3$ med t ex 25 decimaler. En körning av programmet ser då ut på följande sätt.



Beräkning av $2/3$ med 25 decimaler

MYCKET STORA TAL

Om talområdet vid heltalsvariabler är för litet, eller om sex siffrors noggrannhet vid flyttalsvariabler inte är tillräckligt, är det lämpligt att använda ASCII-aritmetiken. Låt oss använda detta för att beräkna x^N , speciellt för stora värden på N .

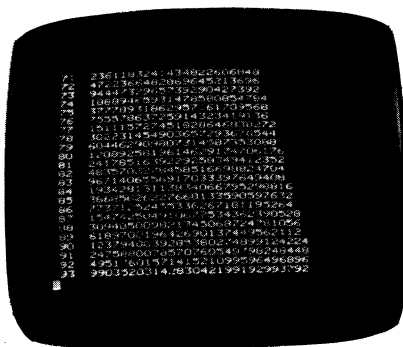
Följande program beräknar x^N där x är ett heltal mellan 1 och 9.

```
10 ; CHR$(12)
20 ; "BERÄKNING AV X**N"
30 ; "VARJE GÅNG TANGENTEN FÖR X"
40 ; "NEDTRYCKS ÖKAS N EN ENHET OCH"
50 ; "N OCH X**N SKRIVS UT PÅ SKÄRMEN"
60 GET A$
70 B$ = A$
80 N = 1
90 N = N + 1
100 A$ = MUL$(A$, B$, 0)
110 IF N > 2 THEN 150
120 ;
130 ; "  N"; TAB(5); "X**N"
140 ;
150 ; N; TAB(5); A$
160 GET C$
170 IF C$ = B$ THEN 90
```

Värdet på x matas in till den numeriska strängen A med en GET-sats. Då tangenten x trycks ned ökar N med en enhet och ett nytt värde beräknas. Om du håller tangenten x nedtryckt hela tiden, går programkörningen fortare, eftersom ju alla tangenter är repeterande.

Skriv in programmet och kör det för några olika värden på x .

Följande figur visar en programkörning där $x = 2$. Endast de sista värdena, just innan resultatet överstiger 29 siffror, visas på TV-skärmen.



Beräkning av x^N med ASCII-aritmetik

ÖNSKAT ANTAL DECIMALER

Med hjälp av ASCII-aritmetiken finns det alltid möjlighet att välja det antal decimaler som är av intresse vid en beräkning, t ex två decimaler för ören i ekonomiska beräkningar. Dessutom kan man använda fler siffror än vid övriga variabeltyper.

Följande program använder ASCII-aritmetik för beräkningarna i det program på sid 32, som användes för att bokföra förändringarna på ett bankkonto. Variablerna måste givetvis ändras till strängar.

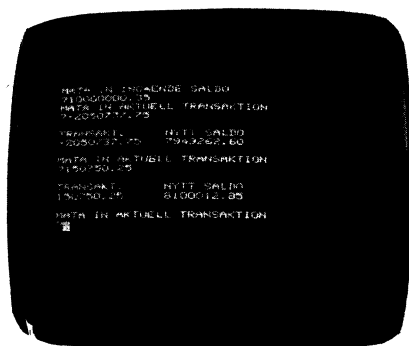
```

10 PRINT
20 PRINT "MATA IN INGÅENDE SALDO";
30 INPUT A$
40 PRINT "MATA IN AKTUELL TRANSAKTION";
50 INPUT B$
60 PRINT
70 A$ = ADD$(A$, B$, 2)
80 PRINT "TRANSAKT.", "NYTT SALDO"
90 PRINT B$, A$
100 PRINT
110 GOTO 40

```

Mångmiljonären Riklund ska använda sin ABC 80 för att beräkna saldot på sitt bankkonto efter att han först köpt en lustyacht för 2050737,75 och därefter fått betalt för försålda maskindetaljer för 150750,25. Hans ingående saldo är 10000000,35. Skriv in programmet och mata in de aktuella transaktionerna.

Efter körning av programmet erhålls följande resultat.



Kontoutdrag med ASCII-aritmetik

SAMMANFATTNING

Instruktion	Kort beskrivning	Exempel
RESTORE	Får nästa READ-sats att läsa data från och med en viss DATA-sats. Följande varianter finns: Läser data från och med den första DATA-satsen. Läser data från och med rad 200.	10 RESTORE 10 RESTORE 200
ON... GOTO...	Hoppar till olika rad-nummer i programmet, beroende på värdet av en variabel. I exemplet ger N = 1 hopp till rad 200, N = 2 hopp till rad 260 o s v.	10 ON N GOTO 200, 260, 310

ON...GOSUB...	Hoppar till olika subrutiner, beroende på värdet av en variabel. I exemplet ger N = 1 hopp till rad 30, N = 2 hopp till rad 120 o s v.	10 ON N GOSUB 30, 120, 400
ON...RESTORE...	Gör RESTORE till olika DATA-satser, beroende på värdet av en variabel. I exemplet hämtas för N = 1 data från och med rad 60, för N = 2 hämtas data från och med rad 70 o s v.	10 ON N RESTORE 60,70, 80
ON ERROR GOTO	Ger hopp till visst radnummer om fel skulle uppstå under programkörningen. Endast de felkoder som i appendix 2 markerats med * medför uthopp. I exemplet ger ett felmeddelande hopp till rad 500.	10 ON ERROR GOTO 500
	Förkortning för instruktionen PRINT.	10; "HEJ"

Strängfunktion	Kort beskrivning
ADDα (Aα, Bα, N)	Adderar strängarna A α och B α tolkade som tal. Resultatet avrundas till N decimaler. Strängarna får innehålla max 29 siffror, samt tecknen + och - och decimalpunkt men inte exponent.

SUB α (Aα, Bα, N)	Dito subtraktion. (A α subtraherat med B α och resultatet med N decimaler.)
MUL α (Aα, Bα, N)	Dito multiplikation. (A α multiplicerat med B α och resultat med N decimaler.)
DIV α (Aα, Bα, N)	Dito division. (A α dividerat med B α och resultatet med N decimaler.)
COMP%(Aα, Bα)	Jämför A α och B α tolkade som tal. Ger följande resultat: -1 om A α < B α 0 om A α = B α 1 om A α > B α

ÖVNINGSEXEMPEL

- 12.1 Skriv ett program som beräknar en fastighetsmäklares 4-procentiga mäklalarvode och beräkna arvudet för följande husförsäljningar.

Försäljningspris: Hus A 5240733:-
Hus B 1750116:-
Hus C 3777750:-
Hus D 2125388:-

Skriv ut resultatet i tabellform (med 2 decimaler)

KAPITEL 13

Att avhjälpa programfel

ABC80

PLANERING

De flesta programfel kan undvikas, genom att den arbetsuppgift som datorn ska utföra definieras noggrant innan programmet skrivs. Programmet är ju egentligen bara en mycket detaljerad beskrivning av datorns uppgift. Vi har redan tidigare, i kapitel 7, beskrivit fördelarna med att använda flödesschema vid den första genomgången av problemet.

Använd gärna subrutiner i dina program istället för hoppinstruktioner, där det är möjligt. Konstruktionen av programmen underlättas betydligt av detta och överskådligheten förbättras.

Ju mera information programmet innehåller i form av REM-satser och PRINT-satser, desto lättare är det att förstå programmet.

Datorns felmeddelanden är en värdefull hjälp när det gäller att avhjälpa fel i form av felaktiga instruktioner, filer på kassett som inte återfinns m m.

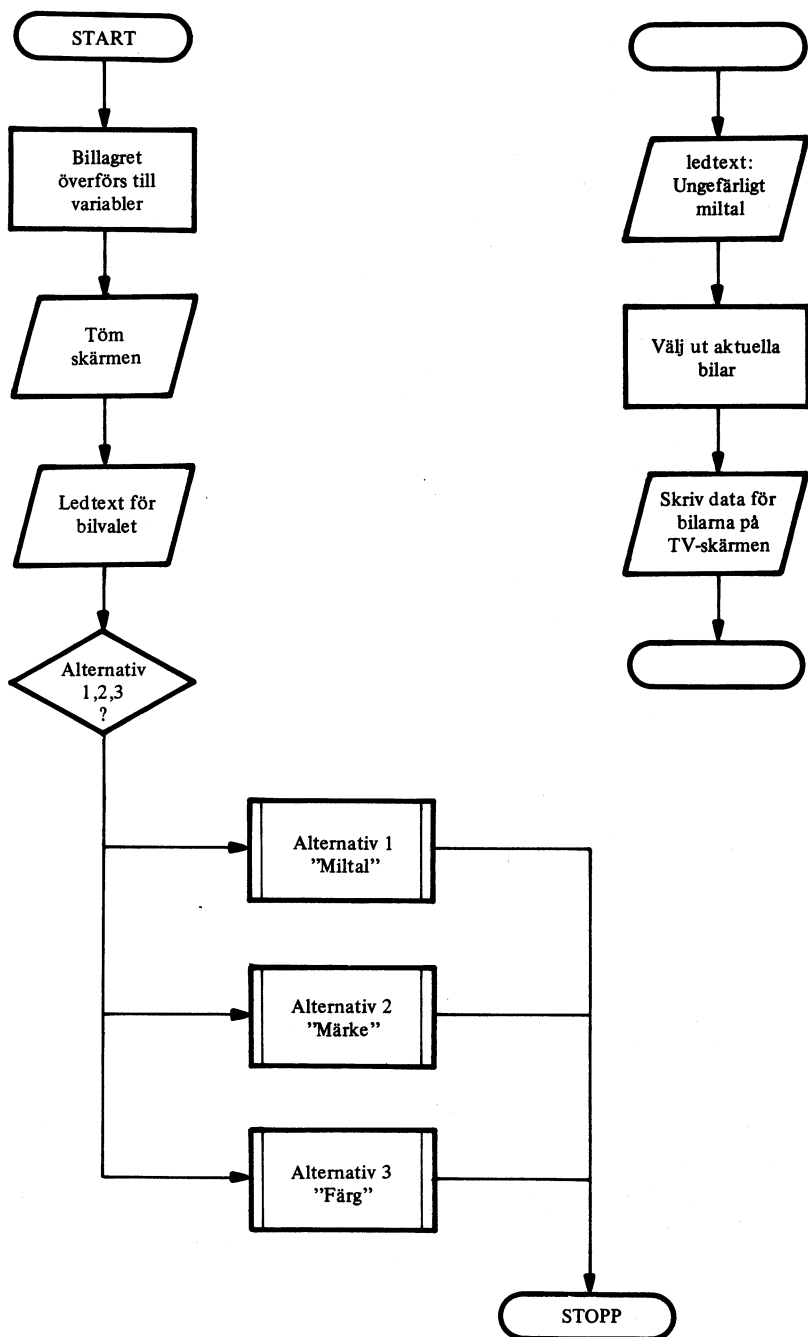
PROGRAMEXEMPEL

Som utgångspunkt för diskussionen ska vi använda ett programexempel, där ett och annat fel insmugit sig. Den uppgift som datorn ska lösa är att presentera vissa delar av en bilhandlares billager, utgående från de krav eller önskemål som kunden har.

En bilköpare som önskar sig en bil som är körd ungefär 7000 mil ska alltså inte behöva läsa en lång lista över alla tillgängliga bilar, utan ska istället kunna fråga datorn vilka bilar som passar in på hans önskemål.

Om bilköparen föredrar ett visst fabrikat eller en speciell färg ska datorn också kunna välja ut de bilar som uppfyller dessa önskemål. Programavsnitt för de två senare alternativen får du själv konstruera, se övningsuppgifterna 13.1 och 13.2.

Programexemplet beskrivs i grova drag av flödesschemat på vidstående sida. Lösningen bygger på en kort inledning, som följs av tre alternativa subrutiner, av vilka endast den första beskrivs av flödesschemat. De två subrutiner som återstår, ersätts tills vidare med ett kort meddelande om vilket alternativ som valts, och att programmet inte är färdigutvecklat för dessa alternativ.



Det program, som skrivits med ledning av flödesschemat, och som alltså innehåller några fel, följer här.

```
10 FOR K = 1 TO 6
20 READ A $\alpha$ , B $\alpha$ , C
40 PRINT CHR $\alpha$ (12)
50 PRINT "ÖNSKAR DU VÄLJA BIL UTGÅENDE
FRÅN"
60 GOSUB 120
70 PRINT "1. MILTAL" : PRINT "2. MÄRKE"
80 PRINT "3. FÄRG"
90 INPUT A
100 ON A GOSUB 140, 210, 230
110 END
```

Resten av programmet utgörs av subrutiner:

```
120 PRINT "(SVAR GES MED ETT TAL)"
130 PRINT : RETURN
```

```
140 PRINT "VILKET UNGEFÄRLIGT MILTAL"
150 GOSUB 120
160 INPUT A
170 FOR K = 1 TO 6
180 IF ABS(A - C(K)) > 2500 THEN 200
190 PRINT A $\alpha$ (K), B $\alpha$ (K), C(K)
200 NEXT K : RETURN
```

```
210 PRINT "PROGRAMMET ÄR INTE FÄRDIGT VA
D AVSER ALTERNATIV 2" : RETURN
220 PRINT "PROGRAMMET ÄR INTE FÄRDIGT VA
D AVSER ALTERNATIV 3" : RETURN
```

```
230 DATA PORSCHE, BLÅ, 4000
240 DATA LAMBORGHINI, GRÖN, 5000
250 DATA FIAT, LILA, 2000
260 DATA VOLVO, RÖD, 7000
270 DATA CHEVROLET, VIT, 8000
280 DATA SAAB, BRUN, 3000
```

DATORN GER FELMEDDELANDE

När du nu kör det felaktiga programmet i exemplet, ger datorn beskedet

ERR 24 LINE 10

Detta innebär att en instruktion som innehåller en NEXT-sats saknas. Felmeddelandet innebär att den saknade loopvariabeln återfinns på rad 10.

10 FOR K = 1 TO 6

Den saknade instruktionen är alltså

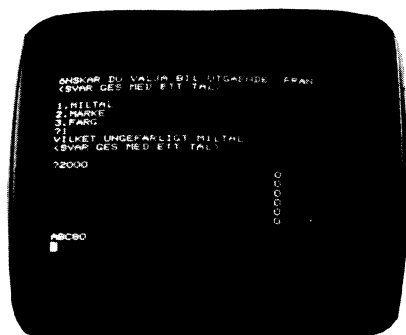
30 NEXT K

För in den i programmet!

Fel av denna typ, som medför att datorn inte känner igen en instruktion eller av någon anledning inte kan utföra en instruktion, är lätt avhjälpta. En förteckning över alla datorns felmeddelanden återfinns i appendix 2. Sist i kapitlet ska vi beskriva möjligheten att låta datorn ge ett felmeddelande i klartext.

SPÅRA PROGRAMFELEN MED TRACE

Kör programmet några gånger och använd då alternativ 1 till att börja med. När du besvarat frågan om vilket ungefärligt miltal som du önskar, ger datorn tyvärr ett felaktigt besked. Svaret skulle enligt programmet ges i form av tre kolumner, en kolumn för vardera märke, färg och miltal. Den utskrift som erhålls består dock bara av två tomma kolumner och en kolumn med enbart nollor, eller också enbart ABC 80.



Körning av alternativ 1

För att man ska kunna undersöka i vilken ordning instruktionerna utförs i programmet, finns kommandot TRACE.

Ge nu kommandot TRACE och kör sedan på nytt. Efter att du matat in 1, för alternativ, och 2000, för miltal, blir hela utskriften.



Körning med TRACE

De tal, som finns längst till vänster på skärmen, d v s

50
60 120
130
70 o s v

är numren på de programrader som utförs.

Exempelvis raden

60 GOSUB 120

innebär ju ett hopp till subrutinen på raderna 120 och 130. Från subrutinen sker återhopp till rad 70.

Efter radnumren följer ibland text, t ex

**50 ÖNSKAR DU VÄLJA BIL UTGÅENDE FRÅN
60 120 (SVAR GES MED ETT TAL)**

Förklaringen till detta är, att de aktuella programraderna innehåller PRINT-satser, t ex

120 PRINT "(SVAR GES MED ETT TAL)"

Om programmet saknar PRINT-satser, skrivs enbart radnumren på TV-skärmen. Loopen på raderna 170, 180, 190 och 200 utförs 6 gånger som sig bör, men den utskrift som ges i rad 190 är varje gång en nolla. Det är felaktigt, och vi måste därför undersöka orsaken till detta.

Vi upptäcker samtidigt ett annat problem. TRACE ger här inte någon information om vad som inträffar på raderna 10 t o m 40. Detta beror på att skärmen töms av instruktionen

40 PRINT CHR\$(12)

Den svårigheten ska vi nu klara upp genom att lägga in ett stopp i programmet.

TILLFÄLLIGA STOPP I PROGRAMMET

Det kan alltså ofta vara praktiskt att införa ett temporärt stopp i programmet. Instruktionen STOP är utmärkt för det ändamålet. Den huvudsakliga skillnaden mellan STOP och END är att STOP inte nollställer variablerna.

Lägg till följande rad och kör programmet.

35 STOP

Eftersom kommandot TRACE fortfarande gäller fås nu följande utskrift.



Körning avbruten med STOP

Nu blir alltså utförandet av instruktionerna på raderna 10, 20, 30 och 35 redovisade.

I ABC 80 finns möjlighet att använda många instruktioner som kommandon. Skriver du t ex en PRINT-sats utan radnummer, tolkas PRINT som ett kommando och utskriften erhålls omedelbart efter tryckning på returtangenten. Eftersom STOP inte nollställer variablerna, kan du nu ge kommandot

```
PRINT A $\alpha$ , B $\alpha$ , C
```

Utskriften blir då

```
SAAB      BRUN      3000
```

som är uppgifterna om den sista bilen i DATA-satserna.

Med TRACE har vi tidigare konstaterat, att utskriften enligt rad 190 blev felaktig. Raden

```
190 PRINT A $\alpha$  (K), B $\alpha$  (K), C(K)
```

gav ingen utskrift i första och andra kolumnerna, d v s A α (K) och B α (K) har inte tilldelats några värden. I tredje kolumnen erhöles nollor, d v s C(K) har heller inte tilldelats något värde av programmet.

Tilldelning sker i loopen på raderna 10, 20 och 30. Vi har nu fått så pass mycket information att vi kan spåra felet, som finns på 20. Där skulle uppenbarligen indexerade variabler ha använts!

Raden ska alltså istället skrivas

```
20 READ A $\alpha$ (K), B $\alpha$ (K), C(K)
```

För att programmet nu ska kunna köras på normalt sätt, måste du först ge kommandot NOTRACE. Observera att TRACE och NOTRACE också kan användas som instruktioner. Avlägsna också rad 35 STOP. Nu fungerar programmet, åtminstone vad avser alternativ 1.

UNDERSÖK PROGRAMMET MED PRINTSATSER

I det här fallet kunde programfelet ganska lätt avslöjas med hjälp av TRACE, eftersom loopen på raderna 170 - 200 innehåller en PRINT-sats. Om den behövliga PRINT-satsen inte är en naturlig del av programmet, kan det ofta vara lämpligt att tillfälligt tillfoga en PRINT-sats för att kontrollera ingående variabelvärden. Särskilt gäller detta i loopar, där ju en tillfogad STOP-instruktion skulle medföra att programavsnittet i loopen bara utfördes en enda gång.

Tillfoga en PRINT-sats i den nu korrigerade loopen på raderna 10, 20 och 30 och placera också in ett tillfälligt stopp.

```
25 PRINT A$(K), B$(K), C(K)
35 STOP
```

Kör programmet igen. Resultatet blir



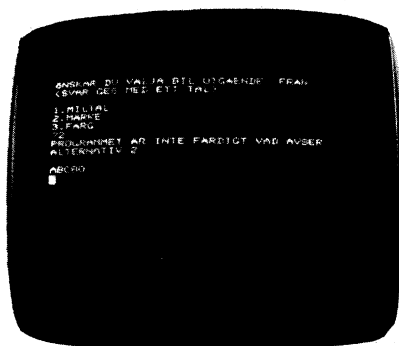
Tillfogad PRINT-sats underlättar felsökning

KONTROLLERA ALLA ALTERNATIV

Inför slutkontrollen av programmet ska du nu först avlägsna de tillfälliga raderna 25 PRINT A\$(K), B\$(K), C(K) och 35 STOP.

Gå tillbaka till flödesschemat och ta reda på vad som återstår att testa i detta program. Alternativen 2, där valet gäller bilmärke, och 3, där färgen bestämmer valet, är ännu inte provade.

Vilket resultat ger en provkörning? Alternativ 2 fungerar utmärkt. Utskriften blir så här.



Körning av alternativ 2

När alternativ 3 provas, sker något oväntat. Datorn ger inte det avsedda meddelandet på TV-skärmen!

Ge kommandot TRACE och kör igen!

Resultatet blir då



Körning av alternativ 3 med TRACE

Instruktionen på rad 100 ger alltså ett hopp till rad 230. Eftersom den avsedda texten i programmet står på rad 220 måste rad 100, ändras till

100 ON A GOSUB 140, 210, 220

När du nu ger kommandot NOTRACE och sedan provkör igen, fungerar programmet även för alternativ 3.

EGNA FELMEDDELANDEN

När ett program utvecklas av en skicklig programmerare, är det inte någon nackdel att datorn ger sina felmeddelanden i form av en kod. Situationen blir annorlunda när det färdiga programmet ska köras av en oinvigd datoranvändare, som naturligtvis inte kan tolka en felkod.

Eftersom de fel som kan förekomma är möjliga att förutse, kan man oftast förse sitt program med textavsnitt, som förklarar varför ett fel har uppstått. Användaren bör också få besked om hur felet ska kunna rättas till.

Anta t ex att du kör vårt programexempel igen och gör det enkla misstaget att svara MILTAL istället för 1 på datorns första fråga.

Som en jäktad bilköpare blir du då ganska förkrossad när dina önskemål oväntat besvaras med

ERR 12 LINE 90

Om programmet utökas obetydligt, kan bilköparen förskonas från detta obehag. Börja med att tillfoga dessa rader och kör programmet.

```
55 ON ERROR GOTO 500  
500 STOP
```

Körningen resulterar i utskriften

STOP LINE 500

Man kan alltså åstadkomma att datorn hoppar till rad 500 när ett fel uppstår. Instruktionen ON ERROR GOTO ... ger hopp till angiven rad endast vid sådana fel, som markerats med en asterisk (*) i appendix 2.

Ersätt nu rad 500 med raderna

```
500 PRINT "FÖRSÖK IGEN"  
510 GOTO 55
```

Övertyga dig om att programmet nu fungerar, även om frågorna får felaktiga svar.

Den enkla metod som används här består i att datorn skriver ut texten FÖRSÖK IGEN, oberoende av vilket misstag som användaren har gjort. Det finns också möjlighet att variera utskriften alltefter felets art. När ett fel har inträffat, kan datorn ge besked om typen av fel genom att svara med talet för felkoden.

Lägg till raden

```
505 PRINT "FEL NR"; ERRCODE
```

När felet har inträffat skriver datorn bl a

FEL NR 12

ERRCODE kan sedan användas för att låta datorn välja ut ett passande felmeddelande.

SAMMANFATTNING

Instruktion	Kort beskrivning	Exempel
TRACE	Radnummer för utförda programrader skrivs ut under körning.	130 TRACE
NOTRACE	Stänger av TRACE.	160 NOTRACE
STOP	Avslutar programkörningen, ger akustisk signal och skriver ut STOP-satsens radnummer. Variablerna nollställs ej.	50 STOP

ON ERROR GOTO

Åstadkommer ett hopp till angiven programrad om ett fel uppstår under körningen. Endast de * -märkta felkoderna i appendix 2 medför hopp. Alla övriga felkoder ger normala felmeddelanden.

**10 ON ERROR
GOTO 500**

Kommando

Kort beskrivning

Exempel

TRACE

Radnummer för utförda programrader skrivs ut under körning.

TRACE

NOTRACE

Stänger av TRACE.

NOTRACE

Funktion

Kort beskrivning

Exempel

ERRCODE

Ger som resultat senaste felkoden.

**500 A = ERRCODE
520 PRINT ERRCODE**

ÖVNINGSEXEMPEL

- 13.1 I programexemplet i detta kapitel finns bara en PRINT-sats som subrutin för alternativ 2. Skriv en fullständig subrutin för detta alternativ. Datorn kan t ex ställa frågan **VILKET MÄRKE FÖREDRAR DU?** När datorn fått svar på denna fråga, ska samtliga bilar av efterfrågat märke redovisas på TV-skärmen. Bilarna presenteras med data för märke, färg och antal körda mil.
- 13.2 Fullborda programexemplet i detta kapitel genom att konstruera en subrutin för alternativ 3, där valet gäller bilens färg.
- 13.3 Hur ska programexemplet i detta kapitel ändras, för att bilhandlaren ska kunna lagra uppgifter om 53 bilar i datorn?

KAPITEL 14

Matrisberäkningar

ABC80

MATRISBERÄKNINGAR

TABELLER

Du har tidigare i kap 8 lärt dig hur man kan tilldela ett sk fält ett eller två index. Ett fält med ett index kallas för lista eller vektor och ett fält med två index kallas för tabell eller matris.

Följande utdrag ur en lagerlista för en beg bilhandlare är ett exempel på en tabell:

Årsmodell

	1974	1975	1976	1977	1978
Fabrikat					
Volvo	4	8	6	6	2
Saab	1	7	3	3	1
Ford	8	7	5	6	2
Mercedes	1	2	2	2	0

Låt oss kalla tabellen ovan för A. Den skuggade siffran står i tredje raden och fjärde kolumnen. Den får då beteckningen $A(3,4)$ där $A(3,4) = 6$. Varje siffra i tabellen ovan kallas element.

Elementen i en tabell kan hos ABC 80 vara flyttal, heltal eller strängar. Det antal element som får plats i en lista eller tabell definieras med instruktionen DIM.

Om man inte använt DIM för att definiera storleken hos en lista eller tabell, reserveras automatiskt plats som om man skrivit DIM A(10) eller DIM A(10,10).

Följande program läser med hjälp av INPUT-instruktionen in tabellen (matrisen) ovan och skriver sedan ut tabellen radvis.

```
10 FOR I = 1 TO 5
20 PRINT "KOLUMN"; I
30 FOR J = 1 TO 4
40 INPUT A(J,I)
```

```

50 NEXT J
60 PRINT
70 NEXT I
80 PRINT " TABELL A"
90 FOR J = 1 TO 4
100 FOR I = 1 TO 5
110 PRINT A(J,I);
120 NEXT I
130 PRINT
140 NEXT J

```

Skriv in och kör programmet. Mata in kolumn efter kolumn i tabellen ovan och kontrollera att utskriften blir korrekt.

Tekniken ovan för att läsa in matriser (tabeller) kolumnvis och därefter skriva ut resultatet, används i de program för matrisberäkningar som vi ska gå igenom i detta kapitel.

MATRISADDITION

Summan av två matriser blir en ny matris, som erhålls genom att elementen på platser med samma rad- och kolumn- nummer adderas.

$$\begin{array}{|c|c|c|} \hline 1 & 7 & 8 \\ \hline 3 & 3 & 2 \\ \hline \end{array} + \begin{array}{|c|c|c|} \hline 1 & 1 & 1 \\ \hline 2 & 1 & 2 \\ \hline \end{array} = \begin{array}{|c|c|c|} \hline 2 & 8 & 9 \\ \hline 5 & 4 & 4 \\ \hline \end{array}$$

A + B = C

Följande program beräknar summan av två matriser A och B och skriver ut resultatet. Antalet rader R och kolumner K definieras med en INPUT-sats. Båda matriserna måste ha samma antal rader resp kolumner. Matriserna A och B definieras kolumnvis med INPUT-satser. Observera att ingen DIM-sats finns med i programmet utan du får skriva dit en sådan själv, om det skulle behövas.

```

10 PRINT "MATRISADDITION"
20 PRINT
30 PRINT "MATRISDIMENSION R,K";
40 INPUT R,K
50 PRINT
60 PRINT "MATRIS A"
70 FOR I = 1 TO K
80 PRINT "KOLUMN"; I
90 FOR J = 1 TO R
100 INPUT A(J,I)
110 NEXT J
120 PRINT
130 NEXT I
140 PRINT "MATRIS B"
150 FOR I = 1 TO K
160 PRINT "KOLUMN"; I
170 FOR J = 1 TO R
180 INPUT B(J,I)
190 NEXT J
200 PRINT
210 NEXT I
220 PRINT
230 PRINT
240 PRINT "SUMMAN AV MATRISERNA BLIR:"
250 FOR J = 1 TO R
260 FOR I = 1 TO K
270 S = A(J,I) + B(J,I)
280 PRINT S;
290 NEXT I
300 PRINT
310 NEXT J

```


MATRISSUBTRAKTION

Skillnaden mellan två matriser fås genom att subtrahera elementen på motsvarande platser med samma rad- och kolumn-nummer subtraheras.

$$\begin{bmatrix} 1 & 7 & 8 \\ 3 & 3 & 2 \end{bmatrix} - \begin{bmatrix} 1 & 1 & 1 \\ 2 & 1 & 2 \end{bmatrix} = \begin{bmatrix} 0 & 6 & 7 \\ 1 & 2 & 0 \end{bmatrix}$$

A - B = C

Programmet för matrisaddition kan enkelt göras om till subtraktion genom att raderna 10, 240 resp 270 ändras till

```
10 PRINT "MATRISSUBTRAKTION"
240 PRINT "SKILLNADEN MELLAN MATRISERNA
BLIR:"
270 S = A(J,I) - B(J,I)
```

MATRISMULTIPLIKATION

Hur produkten av två matriser definieras framgår av följande exempel.

$$\begin{bmatrix} x & x & x \\ a & b & c \\ x & x & x \\ x & x & x \end{bmatrix} * \begin{bmatrix} x & d \\ x & e \\ x & f \end{bmatrix} = \begin{bmatrix} x & x \\ x & g \\ x & x \\ x & x \end{bmatrix}$$

A * B = C

4 rader 3 rader 4 rader
3 kolumner 2 kolumner 2 kolumner

Elementet $g = C(2,2)$ fås ur de skuggade elementen i A och B enligt följande:

$$g = a \cdot d + b \cdot e + c \cdot f$$

Övriga element fås på motsvarande sätt.

Observera att antalet kolumner i matris A måste vara lika med antalet rader i kolumn B.

Följande program beräknar produkten av två matriser A och B och skriver ut resultatet. Antalet rader och kolumner i respektive matris definieras med två INPUT-satser. Matriserna A och B definieras kolumnvis med INPUT-satser.

```
10 PRINT "MATRISMULTIPLIKATION"
20 PRINT
30 PRINT "DIMENSION HOS MATRIS A R,K";
40 INPUT R1, K1
50 PRINT "DIMENSION HOS MATRIS B R,K";
60 INPUT R2, K2
70 IF K1 = R2 THEN 100
80 PRINT "A OCH B KAN INTE MULTIPLICERAS
. FEL DIMENSION"
90 GOTO 20
100 PRINT
110 PRINT "MATRIS A"
120 FOR I = 1 TO K1
130 PRINT "KOLUMN"; I
140 FOR J = 1 TO R1
150 INPUT A(J,I)
160 NEXT J
170 PRINT
180 NEXT I
190 PRINT
200 PRINT "MATRIS B"
210 FOR I = 1 TO K2
220 PRINT "KOLUMN"; I
230 FOR J = 1 TO R2
240 INPUT B(J,I)
250 NEXT J
260 PRINT
270 NEXT I
280 PRINT
290 PRINT
300 PRINT "PRODUKTEN AV MATRISERNA BLIR:
"
310 FOR J = 1 TO R1
320 FOR I = 1 TO K2
330 S = 0
340 FOR K = 1 TO R2
350 S = S + A(J,K) * B(K,I)
360 NEXT K
```

```

370 PRINT S;
380 NEXT I
390 PRINT
400 NEXT J

```

MATRISINVERTERING

Den inverterade matrisen till A brukar betecknas A^{-1} . Definitionsmässigt gäller att

$$A^{-1} * A = E$$

där E är den s k enhetsmatrisen. Alla element i E är lika med noll, utom ena diagonalen där alla element är ett.

Exempel på enhetsmatris:

$$E = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

För att en matris ska kunna inverteras måste antalet rader och kolumner vara lika.

Följande program beräknar den inverterade matrisen till A och skriver ut resultatet, avrundat till tre decimaler. Antalet rader (= antalet kolumner) definieras med en INPUT-sats. Matrisen A definieras kolumnvis med en INPUT-sats.

```

10 PRINT "MATRISINVERTERING"
20 PRINT
30 PRINT "MATRISDIMENSION R";
40 INPUT R
50 PRINT
60 PRINT "MATRIS A"
70 FOR I = 1 TO R
80 PRINT "KOLUMN"; I
90 FOR J = 1 TO R
100 INPUT A(J,I)
110 NEXT J
120 PRINT

```

```

130 B(I,I) = 1
140 NEXT I
150 FOR J = 1 TO R
160 FOR I = J TO R
170 IF A(I,J) <> 0 THEN 210
180 NEXT I
190 PRINT "MATRISEN SINGULÄR. KAN EJ INV
ERTERAS"
200 GOTO 20
210 FOR K = 1 TO R
220 S = A(J,K)
230 A(J,K) = A(I,K)
240 A(I,K) = S
250 S = B(J,K)
260 B(J,K) = B(I,K)
270 B(I,K) = S
280 NEXT K
290 T = 1 / A(J,J)
300 FOR K = 1 TO R
310 A(J,K) = T * A(J,K)
320 B(J,K) = T * B(J,K)
330 NEXT K
340 FOR L = 1 TO R
350 IF L = J THEN 410
360 T = - A(L,J)
370 FOR K = 1 TO R
380 A(L,K) = A(L,K) + T * A(J,K)
390 B(L,K) = B(L,K) + T * B(J,K)
400 NEXT K
410 NEXT L
420 NEXT J
430 PRINT
440 PRINT
450 PRINT "INVERTERAD MATRIS"
460 FOR I = 1 TO R
470 FOR J = 1 TO R
480 PRINT INT (B (I,J) * 1000 + .5) / 1000;
490 NEXT J
500 PRINT
510 NEXT I

```

LÖSNING AV EKVATIONSSYSTEM

Betrakta följande allmänna ekvationssystem

$$\begin{cases} a_{11} x_1 + a_{12} x_2 + \dots + a_{1n} x_n = c_1 \\ a_{21} x_1 + a_{22} x_2 + \dots + a_{2n} x_n = c_2 \\ \vdots \\ a_{n1} x_1 + a_{n2} x_2 + \dots + a_{nn} x_n = c_n \end{cases}$$

$x_1, \dots, x_n$ är obekanta. Med hjälp av matriser kan ekvationssystemet skrivas på följande sätt.

$$A * X = C$$

där

$$A = \begin{bmatrix} a_{11} & a_{12} & \dots & a_{1n} \\ a_{21} & a_{22} & \dots & a_{2n} \\ \vdots & \vdots & & \vdots \\ a_{n1} & a_{n2} & \dots & a_{nn} \end{bmatrix}$$

$$X = \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{bmatrix}$$

$$C = \begin{bmatrix} c_1 \\ c_2 \\ \vdots \\ c_n \end{bmatrix}$$

Lösningen till ekvationssystemet, skriven i matrisform, blir $X = A^{-1} * C$, dvs A inverteras och multipliceras därefter med C.

Följande program bestämmer lösningen till ekvationssystemet $A * X = C$ och skriver ut resultatet. Antalet obekanta definieras med en INPUT-sats. Matrisen A och vektorn C definieras med INPUT-satser.

```
10 PRINT "LÖSNING AV EKVATIONSSYSTEM"
20 PRINT
30 PRINT "ANTAL OBEKANTA";
40 INPUT R
50 PRINT
60 PRINT "MATRIS A"
70 FOR I = 1 TO R
80 PRINT "KOLUMN"; I
```

```

90 FOR J = 1 TO R
100 INPUT A(J,I)
110 NEXT J
120 PRINT
130 B(I,I) = 1
140 NEXT I
150 PRINT "VEKTOR C"
160 FOR L = 1 TO R
170 INPUT Y(L)
180 NEXT L
190 FOR J = 1 TO R
200 FOR I = J TO R
210 IF A(I,J) <> 0 THEN 250
220 NEXT I
230 PRINT "EKVATIONSSYSTEMET EJ LÖSBART"

240 GOTO 20
250 FOR K = 1 TO R
260 S = A(J,K)
270 A(J,K) = A(I,K)
280 A(I,K) = S
290 S = B(J,K)
300 B(J,K) = B(I,K)
310 B(I,K) = S
320 NEXT K
330 T = 1 / A(J,J)
340 FOR K = 1 TO R
350 A(J,K) = T * A(J,K)
360 B(J,K) = T * B(J,K)
370 NEXT K
380 FOR L = 1 TO R
390 IF L = J THEN 450
400 T = - A(L,J)
410 FOR K = 1 TO R
420 A(L,K) = A(L,K) + T * A(J,K)
430 B(L,K) = B(L,K) + T * B(J,K)
440 NEXT K
450 NEXT L
460 NEXT J
470 PRINT
480 PRINT

```

```

490 PRINT "EKVATIONSSYSTEMETS LÖSNING:"
500 FOR I = 1 TO R
510 S = 0
520 FOR K = 1 TO R
530 S = S + B(I,K) * Y(K)
540 NEXT K
550 PRINT "X "; I; " = "; S
560 NEXT I

```

SAMMANFATTNING

Instruktion	Kort beskrivning	Exempel
DIM	Reserverar utrymme för fältvariabler.	
	Exempel:	
	Reserverar utrymme för 26 flyttalsvariabler A(0) t o m A(25).	10 DIM A(25)
	Reserverar utrymme för en sträng A\$ med max 200 tecken.	20 DIM A\$ = 200
	Reserverar utrymme för 31 strängvariabler A\$(0) t o m A\$(30) om vardera max 20 tecken.	30 DIM A\$(30) = 20
	Reserverar utrymme för en matris med heltalsvariabler innehållande 11 rader och 21 kolumner.	40 DIM A%(10,20)
	Reserverar utrymme för en strängmatris med 5 rader och 6 kolumner och där varje element omfattar max 30 tecken.	50 DIM A\$(4,5) = 30

ÖVNINGSEXEMPEL

1. $A = \begin{bmatrix} 1 & 5 & 4 \\ 2 & 2 & 4 \\ 3 & 7 & 1 \end{bmatrix}$ $B = \begin{bmatrix} 3 & -2 & -1 \\ -4 & 1 & -1 \\ 2 & 0 & 1 \end{bmatrix}$

$C = \begin{bmatrix} 2 & 3 \\ 6 & 5 \\ 1 & 7 \end{bmatrix}$ $D = \begin{bmatrix} 1 & 2 & 1 \\ 3 & 4 & 3 \end{bmatrix}$

- a) Beräkna $A + B$
- b) Beräkna $A - B$
- c) Beräkna $C * D$
- d) Beräkna B^{-1}

2. Lös följande ekvationssystem

$$\begin{cases} x + y + z = 6 \\ 2x + y + 3z = 13 \\ 3x + y + 2z = 11 \end{cases}$$

APPENDIX 1

Kommandon och instruktioner

ABC80

Kommando	Kort beskrivning	Exempel
CLEAR	Nollställer alla variabler och stänger alla filer. Avbryter programkörningen.	CLEAR
ED	Ger möjlighet till ändring i en programrad utan att den behöver skrivas om.	ED 40
LIST	Ger utskrift på bildskärmen av det program som finns i minnet. Följande varianter finns: Samtliga rader. Endast en rad. Allt mellan två radnummer. Allt t o m viss rad. Allt fr o m viss rad.	LIST LIST 100 LIST 100-200 LIST - 200 LIST 100 -
LOAD...	Överför program med angivet namn från kassett till minnet. Raderar vid inläsning ev gammalt program i ABC 80.	LOAD PROG2
LOAD CAS:	Överför närmaste program från kassett till minnet.	LOAD CAS:
MERGE...	Hämtar angivet program från kassett utan att radera det program som finns i minnet. Om samma radnummer både finns på kassett och i minnet kommer raden från kassetten att användas.	MERGE PROGRAM3
MERGE CAS:	Hämtar närmaste program från kassett utan att radera det program som finns i minnet. Om samma radnummer både finns på kassett och i minnet kommer raden från kassetten att användas.	MERGE CAS:
NEW	Raderar programmet i minnet och stänger alla filer.	NEW

Kommando	Kort beskrivning	Exempel
NOTRACE	Stänger av TRACE. Jfr TRACE.	NOTRACE
REN	Numrerar om alla raderna i programmet och ändrar satser med GOTO, GOSUB o s v så att de syftar på rätt rad. Följande varianter finns: Numrerar raderna med början på 10 och med intervall 10. Numrerar raderna med början på ett visst radnummer, t ex 25, och med intervall 25. Numrerar raderna med början på visst radnummer, t ex 500, och med ett önskat intervall, t ex 20.	REN REN 25 REN 500, 20
RUN	Kör det program, som finns i minnet efter att ha nollställt alla variabler.	RUN
RUN...	Hämtar program med angivet namn från kassett och kör programmet efter att ha raderat ev gammalt program i ABC 80.	RUN PROG1
RUN CAS:	Hämtar närmaste program från kassett och kör programmet efter att ha raderat ev gammalt program i ABC 80.	RUN CAS:
SAVE...	Lagrar ett namngivet program på kassett.	SAVE PROG1
SCR	Raderar programmet i minnet och stänger alla filer. Har samma effekt som NEW.	SCR
TRACE	Radnummer för utförda programrader skrivs ut under körning.	TRACE

Instruktion	Kort beskrivning	Exempel
CHAIN "..."	Hämtar angivet program från kassett. Har samma effekt som RUN... men är en instruktion.	100 CHAIN "PROG2"
CLRDOT R,K	Släcker en grafisk punkt på skärmen. Jfr SETDOT R,K.	40 CLRDOT 30,40
CLOSE	Stänger en fil, dvs avslutar läsning eller skrivning mot filen. Om filen öppnats med PREPARE skrivs ett filslutmärke. Bandspelare stoppas.	10 CLOSE 2
DATA	Lagrar variabelvärden som ska läsas av en READ-sats. Om strängar innehåller mellanslag eller kommatecken måste de omges av citations-tecken. DATA-satser kan stå var som helst i programmet.	20 DATA 5.31, 4, HEJ 30 DATA "HEJ DÄR"
DEF FN...()	Definierar en egen funktion.	10 DEF FNA(X,Y) = X + X * Y
DIM	Reserverar utrymme för fältvariabler. Exempel. Reserverar utrymme för 26 flyttalsvariabler A(0) t o m A(25). Reserverar utrymme för en sträng A\$ med max 200 tecken. Reserverar utrymme för 31 strängvariabler A\$(0) t o m A\$(30) om vardera maximalt 20 tecken. Reserverar utrymme för en matris med heltalsvariabler innehållande 11 rader och 21 kolumner. Reserverar utrymme för en strängmatris med 4 rader och 6 kolumner och där varje element omfattar maximalt 30 tecken.	10 DIM A(25) 20 DIM A\$ = 200 30 DIM A\$(30) = 20 40 DIM A%(10,20) 50 DIM A\$(3,5) = 30

Instruktion	Kort beskrivning	Exempel
END	Avslutar programkörning, stänger alla filer och nollställer alla variabler.	200 END
FOR...TO...STEP...	Utför ett programavsnitt det antal gånger som bestäms av loopvariabelns start- och slutvärde samt stegvärdet.	10 FOR X = 10 TO 25 STEP 5
	Programavsnittet avslutas alltid med NEXT...	50 NEXT X
GET	Väntar på tangentnedtryckning. Lagrar mottaget tecken i en strängvariabel, utan att skriva på TV-skärmen.	10 GET A ♂
GOSUB	Medför hopp till subrutin. När ett RETURN påträffas sker återhopp till satsen efter motsvarande GOSUB.	30 GOSUB 90
GOTO	Får programkörningen att fortsätta på viss rad.	20 GOTO 10
IF...THEN...ELSE...	Om uttrycket efter IF är sant så utförs det som står mellan THEN och ELSE annars utförs det som står efter ELSE.	50 IF A = B THEN PRINT A ELSE GO TO 300
	Om det bara står ett radnummer efter THEN eller efter ELSE utförs hopp till detta radnummer.	60 IF A = B THEN 90 ELSE 200
INPUT	Skriver ett frågetecken på bildskärmen och väntar därefter på inmatning av data via tangentbordet. För data inmatade via tangentbordet gäller samma regler som för data i DATA-satser.	20 INPUT A 30 INPUT A% 40 INPUT A ♂
INPUT # ...	Läser från en fil med angivet nummer. I övrigt se INPUT.	10 INPUT # 1, A
INPUTLINE	Läser in en rad från tangentbordet, precis som den står, komplett med mellanslag, kommatecken och citationstecken. Även tecknen CR (vagnretur) och LF (radframmatning) ingår i slutet på raden.	20 INPUTLINE A ♂

Instruktion	Kort beskrivning	Exempel
INPUTLINE # ...	Läser från en fil med angivet nummer. I övrigt se INPUTLINE.	30 INPUTLINE # 1, A
LET	Tilldelar en variabel ett värde. Uttrycket kan även skrivas direkt utan LET.	30 LET A = 5 30 A = 5 40 LET A = B 40 A = B
NEXT	Avslutar ett FOR-avsnitt, d v s begränsar en loop.	50 NEXT X
NOTRACE	Stänger av TRACE. Jfr TRACE.	60 NOTRACE
ON ERROR GOTO	Ger hopp till visst radnummer om fel skulle uppstå under programkörningen. Endast de felkoder som i appendix 2 markerats med * ger uthopp. I exemplet medför ett felmeddelande hopp till rad 500.	10 ON ERROR GOTO 500
ON...GOSUB...	Hoppar till olika subrutiner, beroende på värdet av en variabel. I exemplet ger N = 1 hopp till rad 30, N = 2 hopp till rad 120 o s v.	10 ON N GOSUB 30, 120, 400
ON...GOTO...	Hoppar till olika radnummer i programmet, beroende på värdet av en variabel. I exemplet ger N = 1 hopp till rad 200, N = 2 hopp till rad 260 o s v.	10 ON N GOTO 200, 260, 310
ON...RESTORE...	Gör RESTORE till olika DATA-satser, beroende på värdet av en variabel. I exemplet hämtas för N = 1 data från och med rad 60, för N = 2 hämtas data från och med rad 70 o s v.	10 ON N RESTORE 60, 70, 80
OPEN...ASFILE...	Tilldelar en befintlig fil visst nummer från 1 till 255. Bandspelare startas.	20 OPEN "BREV" ASFILE 2
OUT 6, N	Ger ljud från den inbyggda ljudgeneratorn. N är ett udda tal, $1 \leq N \leq 255$.	10 OUT 6,7

Instruktion	Kort beskrivning	Exempel
PREPARE...ASFILE...	Skapar en ny fil för skrivning med ett visst nummer från och med 1 till och med 255. Bandspelare startas.	10 PREPARE "BREV" ASFILE 2
PRINT	Skriver på TV-skärmen. Kan även innehålla beräkningsuttryck eller specialfunktionerna TAB och CUR. Förkortning för instruktionen PRINT.	10 PRINT "HEJ", A 20 PRINT A, A% 30 PRINT (A + B) * 2 10 ; "HEJ", A
PRINT # ...,	Skriver på en fil med angivet nummer. I övrigt, se PRINT.	10 PRINT # 1, "HEJ"
RANDOMIZE	Ger RND-funktionen ett slumpmässigt startvärde.	10 RANDOMIZE
READ	Läser variabelvärden från DATA-satser och tilldelar dem till variabler.	10 READ A, A%, A
REM	Kommentar kan skrivas i programmet.	30 REM ** START **
RESTORE	Får nästa READ-sats att läsa data från och med en viss DATA-sats. Följande varianter finns: Läser data från och med den första DATA-satsen. Läser data från och med rad 200.	10 RESTORE 10 RESTORE 200
RETURN	Avslutar subrutin och medför återhopp till huvudprogrammet till instruktionen efter motsvarande GOSUB.	120 RETURN

Instruktion	Kort beskrivning	Exempel
SETDOT R,K	Tänder en grafisk punkt på skärmen. (Raden måste vara satt i grafisk mod.) $0 \leq R \leq 72$ $2 \leq K \leq 79$ Jfr CLRDOT R,K.	30 SETDOT 30, 40
STOP	Avslutar programkörningen, ger akustisk signal och skriver ut STOP-satsens radnummer. Variabler nollställs ej.	50 STOP
TRACE	Radnummer för utförda programrader skrivs ut under körning.	10 TRACE
Strängfunktion	Kort beskrivning	Exempel
ASC (A$\square$)	Ger ASCII-värdet på första tecknet i A $\square$.	50 V = ASC (A$\square$)
CHR$\square$ (I)	Ger ett tecken med ASCII-värdet I. Kan innehålla upp till och med fyra ASCII-tecken.	10 PRINT CHR$\square$ (7) 20 PRINT CHR$\square$ (65, 66, 67)
INSTR (I,A$\square$,B$\square$)	Söker efter strängen B $\square$ i strängen A $\square$ med början i position I. Funktionen ger läget för första förekomsten av B $\square$ i A $\square$. Om B $\square$ ej finns i A $\square$ erhålls värdet 0.	60 P = INSTR (I, A$\square$, B$\square$)
LEFT$\square$ (A$\square$,J)	Ger de J första tecknen i strängen A $\square$.	10 C$\square$ = LEFT$\square$ (A$\square$,J)
LEN (A$\square$)	Ger aktuella längden av strängen A $\square$.	40 L = LEN (A$\square$)
MID$\square$ (A$\square$,I,J)	Ger J tecken från och med teckenposition I.	20 D$\square$ = MID$\square$ (A$\square$,I,J)
NUM$\square$ (A)	Ger en sträng med de tecken man skulle fått om man skrivit ut A med en PRINT-sats.	70 A$\square$ = NUM$\square$ (A)
RIGHT$\square$ (A$\square$,I)	Ger alla tecken i A $\square$ från och med tecken I.	30 D$\square$ = RIGHT$\square$ (A$\square$,I)
SPACE$\square$ (I)	Ger en sträng av I st mellanslag (space).	50 PRINT SPACE$\square$ (I)
STRING$\square$ (I,C)	Ger en sträng av I st CHR $\square$ (C).	60 PRINT STRING$\square$ (I,C)

Strängfunktion	Kort beskrivning	Exempel
VAL (A)	Ger värdet av A tolkat som ett tal.	80 A = VAL (A)
+	Sammanfogar två strängar.	10 A = B + C + "I"
ASCII-funktion	Kort beskrivning	Exempel
ADD (A ,B ,N)	Adderar strängarna A och B tolkade som tal. Resultatet avrundas till N decimaler. Strängarna får innehålla max 29 siffror, samt tecknen + och - samt decimalpunkt men inte exponent.	10 C = ADD (A,B,N)
SUB (A ,B ,N)	Dito subtraktion. (A subtraherat med B och resultatet med N decimaler.)	10 C = SUB (A,B,N)
MUL (A ,B ,N)	Dito multiplikation. (A multiplicerat med B och resultatet med N decimaler.)	10 C = MUL (A,B,N)
DIV (A ,B ,N)	Dito division. (A dividerat med B och resultatet med N decimaler.)	10 C = DIV (A,B,N)
COMP% (A,B)	Jämför A och B tolkade som tal. Ger följande resultat: -1 om $A < B$ 0 om $A = B$ 1 om $A > B$	10 A = COMP% (A,B ,N)
Matematisk funktion	Kort beskrivning	Exempel
SIN (X)	$\sin x$, x i radianer	10 Y = SIN (X)
COS (X)	$\cos x$, x i radianer	20 Y = COS (5 * X)
TAN (X)	$\tan x$, x i radianer	30 Y = TAN (X+2)
ATN (X)	$\arctan x$	40 Y = ATN (.5)
LOG (X)	e-logaritmen av x	50 Y = LOG (756)

Matematisk funktion	Kort beskrivning	Exempel
LOG 10 (X)	10-logaritmen av x	60 Y = LOG 10 (1000%)
EXP (X)	e^x	70 Y = EXP (2)
SQR (X)	Kvadratroten ur x, $\sqrt{x}$	80 Y = SQR (A ** 2 + B ** 2)
INT (X)	Största heltalet mindre än eller lika med x	90 Y = INT (Z)
FIX (X)	Heltalsdelen av x, $[x]$	100 Y = FIX (Z)
ABS (X)	Absolutbeloppet av x, $ x $	110 Y = ABS (-6)
SGN (X)	-1 om $x < 0$ 0 om $x = 0$ 1 om $x > 0$	120 Y = SGN (X)
PI	π , d v s talet 3,14159	130 X = PI * R ** 2
RND	Ger ett slumpstal mellan 0 och 0.999999 inklusive dessa.	10 LET A = RND 20 LET B = 10 * RND

Specialfunktion	Kort beskrivning	Exempel
DOT (R,K)	Ger värdet SANT (-1) om den grafiska punkten R,K är tänd annars värdet FALSKT (0).	30 IF DOT (R,K) THEN CLRDOT R,K ELSE SETDOT R,K
ERRCODE	Ger som resultat senaste felkoden.	10 A = ERRCODE
TAB (K)	Flyttar markören till position K på raden. Förekommer endast i PRINT-satser. $0 \leq K \leq 39$	10 PRINT TAB (10); " * "
CUR (R,K)	Flyttar markören till rad R och kolumn K på TV-skärmen. Förekommer endast i PRINT-satser. $0 \leq R \leq 23$ $0 \leq K \leq 39$	10 PRINT CUR (11,12); "MITT PÅ SKÄRMEN"

SPECIALTECKEN

Specialtecken	Kort beskrivning	Exempel
CHR ⌘ (7)	Ger pip i den inbyggda högtalaren.	10 PRINT CHR ⌘ (7)
CHR ⌘ (12)	Tömmer bildskärmen och flyttar markören till övre vänstra hörnet.	10 PRINT CHR ⌘ (12)
CHR ⌘ (151)	"START GRAFIK", startar grafisk mod på en rad.	10 PRINT CHR ⌘ (151); "1234"
CHR ⌘ (135)	"SLUT GRAFIK", avslutar grafisk mod på raden.	10 PRINT CHR ⌘ (151); "1234"; CHR ⌘ (135); "1"

Uttryck

Ett uttryck består av ett antal tal och funktioner åtskilda av operatorer. Följande operatorer finns:

Prioritet	Operator	Betydelse	Exempel
1	** alt. Ü	exponentiering	B**C%
2	*	multiplikation	A*B
2	/	division	A/C
3	+	addition	A+B
3	-	subtraktion	A-B

Prioriteten, som anger i vilken ordning operationerna utförs, är den gängse. Dvs först utförs exponentiering, därefter multiplikation och division och slutligen addition och subtraktion. Vid lika prioritet utförs operationerna från vänster till höger. Ordningsföljden kan ändras med parenteser.

Vid exponentiering av negativa tal måste exponenten vara av heltalstyp.

Det finns också ett antal relationsoperatorer:

>, > =, <, < =, =, <>

Dessa har prioritet 4.

Relationsoperator	Kort beskrivning	Exempel
>	Större än. Minnesregel: stor öppning mot stort tal.	10 IF A>B THEN GOTO 50
<	Mindre än.	20 IF A<B THEN 50
=	Lika med.	30 IF A = B THEN PRINT A
>=	Större än eller lika med.	40 IF A>= B THEN 70
<=	Mindre än eller lika med.	50 IF A<= B THEN 10
<>	Ej lika med. (Skilt från.)	60 IF A<>B THEN GOTO 10

Relationsoperatoreorna kan också användas mellan två stränguttryck. Då jämförs tecknens ASCII-representation tecknvis från vänster. Jämförelsen avslutas då två olika tecken påträffas. Strängen med den tidigare bokstaven (i alfabetet) anses då vara minst. Om någon sträng "tar slut" anses den vara den mindre av de två.

APPENDIX 2

Felmeddelanden

ABC80

Felmeddelanden

ERROR	0	EJ TILLÅTET ÖKA "DIM". Ett fält får inte ökas utöver sin ursprungliga längd.
ERROR	1	FEL ANTAL INDEX. Antalet index överensstämmer ej med DIM.
ERROR	2	OTILLÅTET SOM KOMMANDO. En instruktion ges direkt, som endast kan utföras som del av ett program.
ERROR	3	MINNET FULLT. Program och data får ej plats.
* ERROR	4	FÖR STORT FLYTTAL.
ERROR	5	FÖR STORT INDEX. Försök att använda index större än motsvarande DIM.
ERROR	6	HITTAR EJ DETTA RADNUMMER. Referens till ett radnummer som inte finns i programmet.
* ERROR	7	FÖR STORT HELTAL. Försök att använda ett tal som ligger utanför datorns kapacitetsområde för heltal.
ERROR	8	FINNS EJ I DETTA SYSTEM. Förekommer endast vid programutveckling.
ERROR	9	INDEX UTANFÖR STRÄNGEN. Index för stort eller negativt.
ERROR	10	TEXTEN FÅR EJ PLATS I STRÄNGEN. För liten dimension på den mottagande strängen.
ERROR	11	FÖRSTÅR EJ. Formellt BASIC-fel. Datorn kan inte känna igen ett kommando eller en instruktion.
* ERROR	12	FELAKTIGT TAL. Talet innehåller tecken som inte är siffror.
ERROR	13	FEL ANTAL ELLER TYP AV ARGUMENT.
ERROR	14	OTILLÅTET TECKEN EFTER SATSEN. Formellt BASIC-fel. Datorn förväntade RETURN eller kolon (:).

- ERROR 15 "=" SAKNAS ELLER PÅ FEL PLATS. Formellt BASIC-fel vid LET och FOR-satser.
- ERROR 16 RADNUMMER SAKNAS. En rad utan radnummer påträffades.
- ERROR 17 OTILLÅTEN BLANDNING AV TAL OCH STRÄNGAR. Strängar och numeriska operationer får inte hopblandas.
- ERROR 18 ")" SAKNAS ELLER PÅ FEL PLATS. Formellt BASIC-fel.
- * ERROR 19 KAN EJ ÖPPNA FLER FILER. För många filer redan öppnade.
- ERROR 20 FÖR LÅNG RAD (>120 tkn). Längsta tillåtna program- eller datarad är 120 tecken.
- * ERROR 21 HITTAR EJ FILEN. Datorn kan inte hitta filen. Filen finns inte tillgänglig eller har sökts under fel namn.
- ERROR 22 OTILLÅTEN SATS.
- ERROR 23 "TO" SAKNAS. Förekommer i FOR-satser.
- ERROR 24 "NEXT" SAKNAS. En FOR-sats finns i programmet utan motsvarande NEXT-sats som ska avsluta loopen.
- ERROR 25 FELAKTIG SATS EFTER "ON". Formellt BASIC-fel.
- ERROR 26 FEL I ON-UTTRYCK.
- ERROR 27 "NEXT" UTAN "FOR". En NEXT-sats har skrivits in i programmet utan motsvarande FOR-sats.
- ERROR 28 FEL VARIABEL EFTER "NEXT". Formellt BASIC-fel.
- ERROR 29 "RETURN" UTAN "GOSUB". En RETURN-sats påträffad i programmet utan att en föregående GOSUB-sats har blivit utförd.
- * ERROR 30 DATA SLUT. Datalistan har blivit tömd och en READ-sats efterfrågade fler data.

- ERROR 31 FEL DATA TILL KOMMANDO. Felaktigt argument till kommandot t ex UNSAVE utan filnamn.
- ERROR 32 FILEN EJ ÖPPNAD. INPUT- eller PRINT-sats till en stängd kanal.
- ERROR 33 "AS FILE" SAKNAS. Förekommer i OPEN- och PREPARE-satser.
- * ERROR 34 SLUT PÅ FILEN. Försök att läsa efter filens slut.
- * ERROR 35 CHECKSUMMAFEL VID LÄSNING. Skivan eller kassetten skadad.
- * ERROR 37 FELAKTIGT RECORDFORMAT. Icke kompatibelt inspelningsformat. Kan också betyda fel på skiva eller kassett.
- * ERROR 38 RECORDNUMMER UTANFÖR FILEN. Försök att läsa längre än filen medger.
- ERROR 50 KVADRATROT UR NEGATIVT TAL. Logiskt fel i programmet.
- ERROR 51 ENHETEN UPPTAGEN.
- ERROR 52 EJ TILL DENNA ENHET. Exempelvis med INPUT från PR:
- ERROR 53 FELAKTIG RAD.
- ERROR 57 FUNKTIONEN EJ DEFINIERAD. En sökt funktion för vilken det inte finns någon DEF-sats.
- ERROR 58 OGILTIGT TECKEN INLÄST.
- ERROR 59 FEL PROGRAMFORMAT. Programmet är sparat under en icke-kompatibel BASIC-version.
- ERROR 60 BIT ADRESS 16 BITAR.
- ERROR 61 KOMMA SAKNAS.
- ERROR 62 DOT-ADRESS UTANFÖR SKÄRMEN.

ERROR 63 "AS" SAKNAS. Fel i NAME AS.

ERROR 64 FELAKTIG "RENAME". Det nya namnet är ej ett korrekt filnamn.

ERROR 65 SPILL I ASCII-ARITMETIK.

ERROR 66 STRÄNG EJ NUMERISK.

ANM: * avser fel som kan hanteras med ON ERROR GOTO.

APPENDIX 3

Svar till övningsuppgifter

ABC80

Tecknet ₤ (sol) har införts i Europa istället för dollartecknet (\$). Utskriften av lösningarna till övningsexemplen har gjorts på en DIABLO-printer med dollartecknet istället för sol.

2.1 HEJ
 PÅ
 DEJ

2.2 HEJ PÅ DEJ

2.3 10 PRINT "HEJ ";
 20 PRINT "PÅ ";
 30 PRINT "DEJ"

2.4 ANNA PETTERSSON
 STORGATAN 10
 580 05 LINKÖPING

2.5 LISA PETTERSSON
 STORGATAN 10
 580 05 LINKÖPING

2.6 10 GOTO 40
 20 PRINT "K";
 30 GOTO 60
 40 PRINT "O";
 50 GOTO 20
 60 PRINT "!"

2.7 Utskriften hamnar i kolumn 3.

```

3.1      5 PRINT
          6 PRINT "TILL VEM ÖNSKAS TELEFONNR"
          7 INPUT N$
          10 READ A$,B$
          20 DATA "PELLE","123456"
          21 DATA "OLLE","234567"
          22 DATA "KURT","122113"
          23 DATA "GUNNAR","433415"
          24 DATA "LENA","545325"
          29 DATA "KARIN","345678"
          30 IF N$=A$ THEN GOTO 50
          35 IF A$="KARIN" THEN GOTO 90
          40 GOTO 10
          50 PRINT "NUMRET ÄR",B$
          60 GOTO 100
          90 PRINT "FINNS EJ MED I REGISTRET"
          100 PRINT "KLART"

```

```

3.2      5 PRINT
          6 PRINT "ANGE ÖNSKAT TELEFONNUMMER"
          7 INPUT N$
          10 READ A$,B$
          20 DATA "PELLE","123456"
          21 DATA "OLLE","234567"
          22 DATA "KURT","122113"
          23 DATA "GUNNAR","433415"
          24 DATA "LENA","545325"
          29 DATA "KARIN","345678"
          30 IF N$=B$ THEN GOTO 50
          35 IF A$="KARIN" THEN GOTO 90
          40 GOTO 10
          50 PRINT "NAMNET ÄR",A$
          60 GOTO 100
          90 PRINT "FINNS EJ MED I REGISTRET"
          100 PRINT "KLART"

```

```

3.3 a    160 DATA "VOLVO","RÖD","3000"
          170 DATA "TOYOTA","BLÅ","6000"

```

```

3.3 b    RAD 130 AVLÄGSNAS.

```

```

3.4      10 PRINT
          20 PRINT "VILKEN FÄRG ÖNSKAS"
          30 INPUT A$
          40 PRINT
          50 PRINT "MÄRKE", "FÄRG", "MIL"
          60 PRINT
          70 READ B$, C$, D$
          80 IF A$=C$ THEN GOTO 200
          90 IF B$="SLUT" THEN GOTO 220
          100 GOTO 70
          110 DATA "FIAT", "BLÅ", "4000"
          120 DATA "SAAB", "GRÖN", "5000"
          130 DATA "VOLVO", "GRÖN", "1000"
          140 DATA "SAAB", "GUL", "2000"
          150 DATA "VOLVO", "BLÅ", "12000"
          160 DATA "VOLVO", "RÖD", "3000"
          170 DATA "TOYOTA", "BLÅ", "6000"
          190 DATA "SLUT", "SLUT", "SLUT"
          200 PRINT B$, C$, D$
          210 GOTO 70
          220 PRINT

3.5      10 PRINT "MATA IN DEN MISSTÄNKTES PERS.NR."
          20 INPUT A$
          30 READ B$
          40 IF A$=B$ THEN 100
          50 IF B$="SLUT" THEN 150
          60 DATA 181207-9273,210116-9249
          70 DATA 250428-9253,261103-8387
          80 DATA 270805-9130,281122-8390
          90 DATA 350109-6998,350529-6776
          91 DATA 361109-7076,SLUT
          92 GOTO 30
          100 PRINT "DEN MISSTÄNKTE ÄR FÖRRÄDARE"
          110 GOTO 200
          150 PRINT "INGET MISSTÄNKTT"
          200 PRINT "SÖKNINGEN KLAR"

```

4.1 a 10 PRINT (2+5)*3

4.1 b 20 PRINT (2+5)/7

```

4.1 c  30 LET A=2
        40 LET B=3
        50 PRINT ((A+B)*A*B)/A

4.2    10 READ A,B,C,D,E
        20 LET S=A+B+C+D+E
        30 DATA 2,4,7,11,16
        50 PRINT "SUMMAN ÄR",S

4.3    4                                2

4.4    10 READ L,A,P,E
        20 PRINT "MEDELÅLDERN=";(L+A+P+E)/4;"ÄR"
        30 DATA 36,38,4,6

4.5    10 PRINT "MATA IN ANTALET KÖRDA MIL"
        20 INPUT M
        30 PRINT "MATA IN BENSINFÖRBR. I LITER"
        40 INPUT B
        50 PRINT "BENSINFÖRBR=";B/M;" LITER/MIL"
        55 PRINT
        60 GOTO 10

4.6    10 PRINT "FÖR VILKET TAL";
        20 INPUT X
        30 PRINT
        40 PRINT "MULTIPLIKATIONSPROGRAM"
        50 LET A=A+1
        60 PRINT X;" *";A="";X*A
        70 IF A=12 THEN 90
        80 GOTO 50
        90 PRINT

4.7    10 PRINT "HUR MÅNGA FAMILJEMEDLEMMAR";
        20 INPUT N
        30 LET A=A+1
        40 PRINT "HUR GAMMAL ÄR FAM.MEDL. ";A
        50 INPUT X
        60 LET S=S+X
        70 IF A=N THEN GOTO 100
        80 GOTO 30
        100 LET M=S/N
        110 PRINT "MEDELÅLDERN ÄR";M;" ÄR"

```

```

5.1      10 FOR I=1 TO 20
          20 PRINT "MOR ROR OCH FAR ÄR EN ORM"
          30 NEXT I

5.2      10 PRINT CHR$(12)
          20 RANDOMIZE
          30 PRINT
          40 PRINT "VINSTLISTA"
          50 PRINT
          60 FOR I=1 TO 10
          70 LET V=INT(100*RND)
          80 PRINT "VINST";I,"NUMMER";V
          90 NEXT I

5.3      10 PRINT CHR$(12)
          20 RANDOMIZE
          30 PRINT
          40 PRINT "VINSTLISTA"
          50 PRINT
          60 FOR I=1 TO 15
          70 LET V=100+INT(200*RND)
          80 PRINT "VINST";I,"NUMMER";V
          90 NEXT I

5.4      10 RANDOMIZE
          20 PRINT
          30 LET T=INT(1+6*RND)
          40 PRINT "TÄRNINGEN STANNAR PÅ";T
          50 IF T=1 THEN T1=T1+1

5.5      10 RANDOMIZE
          20 LET A=INT(100*RND)
          30 LET B=INT(100*RND)
          40 PRINT
          50 PRINT "TALEN ÄR";A;" OCH";B
          60 PRINT "VAD ÄR SUMMAN";
          70 INPUT S
          80 IF S=A+B THEN 130
          90 PRINT
          100 PRINT "**** FEL ****"
          110 PRINT
          120 GOTO 50
          130 PRINT
          140 PRINT "**** RÄTT SVARAT ****"

```



```

5.6      10 PRINT
          20 FOR I=1 TO 10
          30 PRINT "PROFESSOR NR";I
          40 PRINT "ANTAL RÄTT ";
          50 INPUT P(I)
          60 PRINT
          70 LET S=S+P(I)
          80 NEXT I
          90 LET M=S/10
          100 PRINT
          110 FOR I=1 TO 10
          120 PRINT "PROF. NR";I,"ANTAL RÄTT";P(I)
          130 NEXT I
          140 PRINT
          150 PRINT "MEDELVÄRDET BLIR:";M;" RÄTT"

```

```

5.7      10 PRINT
          20 PRINT "GISSA ETT SLUMPTAL MELLAN 0 OCH 99"
          30 PRINT
          40 RANDOMIZE
          50 LET S=INT(RND*100)
          60 PRINT "DIN GISSNING"
          70 INPUT G
          80 LET F=F+1
          90 IF G=S THEN GOTO 170
          100 IF G>S THEN GOTO 140
          110 PRINT "FÖR LITET"
          120 PRINT
          130 GOTO 60
          140 PRINT "FÖR STORT"
          150 PRINT
          160 GOTO 60
          170 PRINT "RÄTT GISSAT"
          180 PRINT "PÅ";F;" FÖRSÖK"

```

5.8 a Sannolikheten för etta är 0,6.
 Sannolikheten för kryss är 0,2.
 Sannolikheten för tvåa är 0,2.

```

5.8 b  10 PRINT CHR$(12)
        20 RANDOMIZE
        30 PRINT "TIPSFÖRSLAG"
        40 PRINT
        50 FOR X=1 TO 13
        60 PRINT "MATCH NR";X,
        70 T=RND
        80 IF T<.5 THEN 130
        90 IF T<.75 THEN 150
        100 PRINT "      2"
        110 NEXT X
        120 GOTO 170
        130 PRINT "1"
        140 GOTO 110
        150 PRINT "  X"
        160 GOTO 110
        170 PRINT
        180 PRINT "LYCKA TILL"

```

```

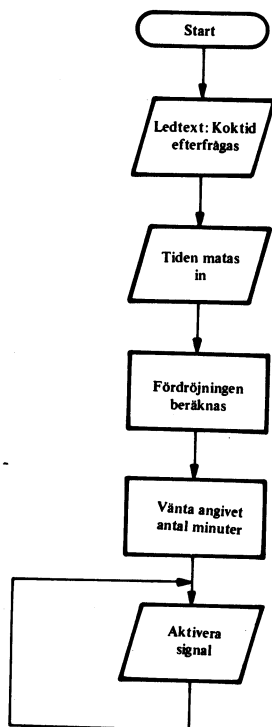
5.9      10 PRINT CHR$(12)
        20 FOR I=1 TO 7
        30 PRINT "SÖKANDE NR ";
        40 INPUT S$(I)
        50 PRINT "IQ ";
        60 INPUT A(I)
        70 NEXT I
        80 FOR I=1 TO 6
        90 FOR J=I+1 TO 7
        100 IF A(I)>A(J) THEN 170
        110 X=A(I)
        120 A(I)=A(J)
        130 A(J)=X
        140 Y$=S$(I)
        150 S$(I)=S$(J)
        160 S$(J)=Y$
        170 NEXT J
        180 NEXT I
        190 PRINT CHR$(12)
        200 PRINT "AGENT BLEV SÖKANDE ";S$(1)
        210 PRINT
        220 PRINT "RESERVER"
        230 PRINT
        240 FOR I=2 TO 7
        250 PRINT "SÖKANDE NR: ";S$(I)
        260 NEXT I

```

```

7.1  10 PRINT "HUR MÅNGA MINUTERS KOKTID ÖNSKAS"
      20 INPUT K
      30 F=K*1000*60
      40 FOR L=1 TO F
      50 NEXT L
      60 PRINT CHR$(7)
      70 GOTO 60

```



7.2 Cirka 35 år.

```

7.4      10 LET N=1
          20 OUT 6,N
          30 FOR I=1 TO 1000 : NEXT I
          40 N=N+2
          50 IF N<256 THEN 20
          60 END

7.5      10 GET A$
          20 IF A$="A" THEN GOSUB 90
          30 IF A$="B" THEN GOSUB 100
          40 IF A$="C" THEN GOSUB 110
          50 IF A$="D" THEN GOSUB 120
          60 IF A$="E" THEN GOSUB 130
          70 IF A$="F" THEN GOSUB 140
          80 GOTO 10
          90 OUT 6,5 : RETURN
          100 OUT 6,3 : RETURN
          110 OUT 6,251 : RETURN
          120 OUT 6,19 : RETURN
          130 OUT 6,9 : RETURN
          140 OUT 6,39 : RETURN

8.1      10 PRINT
          20 PRINT "VAD HETER DU MIN VÄN"
          30 INPUT A$
          40 PRINT
          50 PRINT "HEJ ";A$
          60 PRINT A$;" ÄR ETT VACKERT NAMN"

8.2      10 A$="ABC OM PROGRAMMERING"
          20 PRINT
          30 FOR I=1 TO 20
          40 IF MID$(A$,I,1)=" " THEN PRINT " -";
          50 PRINT MID$(A$,I,1);
          60 NEXT I
          70 PRINT

8.3      10 PRINT "MATA IN DITT PERSONNUMMER"
          20 INPUT A$
          30 B$=LEFT$(A$,2)
          40 PRINT
          50 PRINT "DU ÄR FÖDD ÄR 19";B$

```

- 8.4 10 RANDOMIZE
 20 FOR X=1 TO 60
 30 FOR I=1 TO 2
 40 A(I)=INT(65+29*RND)
 50 NEXT I
 60 PRINT CHR\$(83,A(1),A(2)),
 70 NEXT X
- 8.5 Namnen sorteras i rätt bokstavsordning
 endast med avseende på första bokstaven.
- 8.6 10 PRINT "MATA IN ÖNSKAD STRÄNG"
 20 INPUTLINE A\$
 30 PRINT CHR\$(12)
 40 A=LEN(A\$)
 50 X\$=STRING\$(A,42)
 60 B\$=LEFT\$(A\$,A-2)
 70 PRINT X\$
 80 PRINT "*" ; B\$; "*"
 90 PRINT X\$
- 8.7 10 PRINT "MATA IN DELTAGARLISTAN"
 20 FOR X=1 TO 4
 30 INPUTLINE A\$(X)
 40 PRINT
 50 NEXT X
 60 PRINT CHR\$(12)
 70 PRINT "D E L T A G A R L I S T A"
 80 PRINT
 90 PRINT
 100 FOR X=1 TO 4
 110 PRINT "ÖVER"+A\$(X)
 120 NEXT X
- 8.8 10 DIM A\$=120
 20 INPUTLINE A\$
 30 A=LEN(A\$)-2 : S=0
 40 FOR X=1 TO A
 50 IF MID\$(A\$,X,1)<>"S" THEN 70
 60 S=S+1
 70 NEXT X
 80 PRINT CHR\$(12)
 90 F=(S/A)*100
 100 PRINT "S-FREKVENSEN = ";F;" %"
 110 IF F<5 THEN 140
 120 OUT 6,7
 130 PRINT "***** SKRIV OM *****"
 140 GOTO 10

```

8.9      10 DIM A$=120
          20 INPUTLINE A$
          30 B$="KUNG SUNE"
          40 X=INSTR(1,A$,B$)
          50 IF X=0 THEN 185
          60 PRINT
          70 PRINT
          80 PRINT "UTBYTES MOT"
          90 PRINT
          100 FOR I=1 TO X-1
          110 PRINT MID$(A$,I,1);
          120 NEXT I
          130 PRINT "KING KONG";
          140 FOR I=X+9 TO LEN(A$)
          150 PRINT MID$(A$,I,1);
          160 NEXT I
          170 GOTO 200
          180 PRINT
          185 PRINT
          190 PRINT "GODKÄNNES"
          200 GOTO 20

9.1      10 PRINT CHR$(12)
          20 PRINT
          30 PRINT "X=";
          40 INPUT X
          50 LET Y=X^5%+X^2%+2
          60 PRINT "Y=";Y
          70 GOTO 20

9.2      10 PRINT "X","LOG(X)","EXP(X)"
          20 FOR X=10 TO 19
          30 PRINT X,LOG(X),EXP(X)
          40 NEXT X

9.3      10 PRINT "X","LOG(X)","EXP(X)"
          20 FOR X=10 TO 19 STEP .5
          30 PRINT X,LOG(X),EXP(X)
          40 NEXT X

```

```

9.4      10 PRINT "BERÄKNING AV C=SQR(A**2%+B**2%)"
          20 PRINT
          30 PRINT "A,B";
          40 INPUT A,B
          50 LET C=SQR(A^2%+B^2%)
          60 PRINT "C="C
          70 GOTO 20

9.5      10 PRINT CHR$(12)
          20 PRINT "BERÄKNING AV (SQR(X))*EXP(-X**2%)"
          30 PRINT
          40 DEFFNA(X)=(SQR(X))*EXP(-X^2%)
          50 PRINT "X","FNA(X)"
          60 READ X
          70 IF X=999 THEN 100
          80 PRINT X,FNA(X)
          90 GOTO 60
          100 DATA 1.7,2.25,2.5,999
          110 END

9.6      10 PRINT CHR$(12)
          20 PRINT "BERÄKNING AV KVADRATROTEN MED TVÅ DECIMALER"
          30 PRINT
          40 PRINT "X=";
          50 INPUT X
          60 PRINT "SQR(X)=";INT(100*SQR(X)+.5)/100
          70 GOTO 30

9.7      10 PRINT CHR$(12)
          20 FOR X=-10 TO 10 STEP .5
          30 LET Y=ATN(X)
          40 PRINT CUR(12-7*Y,2*X+20);""
          50 NEXT X

9.8      10 PRINT CHR$(12)
          20 FOR X=0 TO 6.4 STEP .4
          30 LET Y=SIN(X)
          40 PRINT CUR(12-10*ABS(Y),5*X);""
          50 NEXT X
          60 PRINT CUR(20,0);"RESULTATET BLIR HELVÄGSLIKRIKTNING"

```

10.1 a 10 PRINT CHR\$(12)
 20 FOR I=0 TO 23
 30 PRINT CUR(I,0);CHR\$(151);
 40 NEXT I
 50 FOR K=30 TO 50
 60 SETDOT 40,K
 70 NEXT K

10.1 b 10 PRINT CHR\$(12)
 20 FOR I=0 TO 23
 30 PRINT CUR(I,0);CHR\$(151);
 40 NEXT I
 50 FOR K=30 TO 50
 60 SETDOT 40,K
 65 SETDOT K,40
 70 NEXT K

10.2 10 FOR A=32 TO 127
 20 PRINT
 30 PRINT A,CHR\$(A),CHR\$(151);CHR\$(A)
 40 FOR X=1 TO 1000 : NEXT X
 50 NEXT A

10.3 10 PRINT "MATA IN TIO VÄRDEN MELLAN"
 20 PRINT "O OCH 50 ÅTSKILDA AV RETURN"
 30 FOR N=1 TO 10
 40 INPUT A(N)
 50 NEXT N
 60 PRINT CHR\$(12)
 70 FOR I=0 TO 23
 80 PRINT CUR(I,0);CHR\$(151)
 90 NEXT I
 100 FOR K=10 TO 55 STEP 5
 110 X=X+1
 120 FOR R=50 TO 50-A(X) STEP -1
 130 SETDOT R,K
 140 NEXT R
 150 NEXT K
 160 FOR K=5 TO 60
 170 FOR R=0 TO 55
 180 IF DOT(R,K) THEN CLRDOT R,K ELSE SETDOT R,K
 190 NEXT R
 200 NEXT K
 210 GOTO 210


```

10.4  10 PRINT CHR$(12)
      20 FOR I=0 TO 23
      30 PRINT CUR(I,0);CHR$(151);
      40 NEXT I
      50 FOR X=2 TO 79
      60 SETDOT 0,X
      70 SETDOT 71,X
      80 NEXT X
      90 FOR X=0 TO 71
      100 SETDOT X,2
      110 SETDOT X,79
      120 NEXT X
      130 GOTO 130

```

```

10.7  10 PRINT CHR$(12)
      20 FOR I=0 TO 23
      30 PRINT CUR(I,0);CHR$(151);
      40 NEXT I
      50 PRINT CUR(5,5);
      60 PRINT "TRYCK PÅ TANGENTEN Ø"
      70 PRINT CUR(6,5);
      80 PRINT "SÅ ÖPPNAS KASSASKÅPET"
      90 GET A$
      100 FOR X=10 TO 60
      110 SETDOT 40,X
      120 SETDOT 60,X
      130 NEXT X
      140 FOR X=40 TO 60
      150 SETDOT X,10
      160 SETDOT X,60
      170 NEXT X
      180 PRINT CUR(15,8);"L A R M"
      190 PRINT CUR(17,8);"VARNING FÖR HUNDEN"
      200 OUT 6,7
      210 FOR X=0 TO 200 : NEXT X
      220 PRINT CUR(15,8);"      "
      230 FOR X=0 TO 300 : NEXT X
      240 GOTO 100

```

```

11.1    10 DIM A(20)
        20 PRINT "MATA SUCCESSIVT IN 20 TAL"
        30 FOR I=1 TO 20
        40 INPUT A(I)
        50 NEXT I
        60 PREPARE "DATAFIL" ASFILE 1
        70 FOR I=1 TO 20
        80 PRINT #1,A(I)
        90 NEXT I
        100 CLOSE 1

11.2    10 DIM A(20)
        13 PRINT "FÖRDRÖJNING NÄR DATA"
        15 PRINT "HÄMTAS FRÅN KASSETT"
        20 OPEN "DATAFIL" ASFILE 1
        30 FOR I=1 TO 20
        40 INPUT #1,A(I)
        50 NEXT I
        60 CLOSE 1
        70 FOR I=1 TO 20
        80 PRINT A(I)
        90 NEXT I

12.1    10 ; CHR$(12)
        20 ; "FÖRSÄLJN.PRIS","MÄKLARARVODE"
        30 ;
        40 P$=".04"
        50 FOR K=1 TO 4
        60 READ A$
        70 ; A$,MUL$(P$,A$,2)
        80 NEXT K
        90 DATA 5240733,1750116,3777750,2125388

13.1    10 FOR K=1 TO 6
        20 READ A$(K),B$(K),C(K)
        30 NEXT K
        40 PRINT CHR$(12)
        50 PRINT "ÖNSKAR DU VÄLJA BIL UTGÅENDE FRÅN "
        60 GOSUB 120
        70 PRINT "1.MILTAL" : PRINT "2.MÄRKE"
        80 PRINT "3.FÄRG"
        90 INPUT A
        100 ON A GOSUB 140,210,260
        110 END

```

```

120 PRINT "(SVAR GES MED ETT TAL)"
130 PRINT : RETURN
140 PRINT "VILKET UNGEFÄRLIGT MILTAL"
150 GOSUB 120
160 INPUT A
170 FOR K=1 TO 6
180 IF ABS(A-C(K))>2500 THEN 200
190 PRINT A$(K),B$(K),C(K)
200 NEXT K : RETURN
210 PRINT "VILKET MÄRKE FÖRDRAR DU";
220 INPUT A$
230 FOR K=1 TO 6
240 IF A$=A$(K) THEN PRINT A$(K),B$(K),C(K)
250 NEXT K : RETURN
260 PRINT "PROGRAMMET ÄR INTE FÄRDIGT "
270 PRINT "VAD AVSER ALTERNATIV 3"
280 RETURN
290 DATA "PORCHE","BLÅ",4000
300 DATA "LAMBORGHINI","GRÖN",5000
310 DATA "FIAT","LILA",1000
320 DATA "VOLVO","RÖD",7000
330 DATA "CHEVROLET","VIT",8000
340 DATA "SAAB","BRUN",3000

```

```

13.2  10 FOR K=1 TO 6
      20 READ A$(K),B$(K),C(K)
      30 NEXT K
      40 PRINT CHR$(12)
      50 PRINT "ÖNSKAR DU VÄLJA BIL UTGÅENDE FRÅN "
      60 GOSUB 120
      70 PRINT "1.MILTAL" : PRINT "2.MÄRKE"
      80 PRINT "3.FÄRG"
      90 INPUT A
     100 ON A GOSUB 140,210,270
     110 END
     120 PRINT "(SVAR GES MED ETT TAL)"
     130 PRINT : RETURN
     140 PRINT "VILKET UNGEFÄRLIGT MILTAL"
     150 GOSUB 120
     160 INPUT A
     170 FOR K=1 TO 6
     180 IF ABS(A-C(K))>2500 THEN 200
     190 PRINT A$(K),B$(K),C(K)
     200 NEXT K : RETURN
     210 PRINT "VILKET MÄRKE FÖRDRAR DU";

```

```

220 INPUT A$
230 FOR K=1 TO 6
240 IF A$=A$(K) THEN PRINT A$(K),B$(K),C(K)
250 NEXT K : RETURN
270 PRINT
280 PRINT "VILKEN FÄRG FÖRDRAR DU";
290 INPUT A$
300 FOR K=1 TO 6
310 IF A$=B$(K) THEN PRINT A$(K),B$(K),C(K)
320 NEXT K : RETURN
340 DATA "PORCHE","BLÅ",4000
350 DATA "LAMBORGHINI","GRÖN",5000
360 DATA "FIAT","LILA",1000
370 DATA "VOLVO","RÖD",7000
380 DATA "CHEVROLET","VIT",8000
390 DATA "SAAB","BRUN",3000

```

```

13.3  1 DIM A$(53),B$(53),C(53)
      10 FOR K=1 TO 53
      20 READ A$(K),B$(K),C(K)
      30 NEXT K
      40 PRINT CHR$(12)
      50 PRINT "ÖNSKAR DU VÄLJA BIL UTGÅENDE FRÅN "
      60 GOSUB 120
      70 PRINT "1.MILTAL" : PRINT "2.MÄRKE"
      80 PRINT "3.FÄRG"
      90 INPUT A
      100 ON A GOSUB 140,210,270
      110 END
      120 PRINT "(SVAR GES MED ETT TAL)"
      130 PRINT : RETURN
      140 PRINT "VILKET UNGEFÄRLIGT MILTAL"
      150 GOSUB 120
      160 INPUT A
      170 FOR K=1 TO 53
      180 IF ABS(A-C(K))>2500 THEN 200
      190 PRINT A$(K),B$(K),C(K)
      200 NEXT K : RETURN
      210 PRINT "VILKET MÄRKE FÖRDRAR DU";
      220 INPUT A$
      230 FOR K=1 TO 53
      240 IF A$=A$(K) THEN PRINT A$(K),B$(K),C(K)
      250 NEXT K : RETURN
      270 PRINT
      280 PRINT "VILKEN FÄRG FÖRDRAR DU";
      290 INPUT A$

```

```

300 FOR K=1 TO 53
310 IF A$=B$(K) THEN PRINT A$(K),B$(K),C(K)
320 NEXT K : RETURN
340 DATA "PORCHE","BLÅ",4000
350 DATA "LAMBORGHINI","GRÖN",5000
360 DATA "FIAT","LILA",1000
370 DATA "VOLVO","RÖD",7000
380 DATA "CHEVROLET","VIT",8000
390 DATA "SAAB","BRUN",3000
398 REM
399 REM *****
400 REM ** YTTERLIGARE 47 DATASATSER **
410 REM ** KRÄVS FÖR ATT PROGRAMMET **
420 REM ** SKA FUNGERA **
430 REM *****

```

14.1 a $A+B = \begin{bmatrix} 4 & 3 & 3 \\ -2 & 3 & 3 \\ 5 & 7 & 2 \end{bmatrix}$

14.1 c $C*D = \begin{bmatrix} 11 & 16 & 11 \\ 21 & 32 & 21 \\ 22 & 30 & 22 \end{bmatrix}$

14.1 b $A-B = \begin{bmatrix} -2 & 7 & 5 \\ 6 & 1 & 5 \\ 1 & 7 & 0 \end{bmatrix}$

14.1 d $B^{-1} = \begin{bmatrix} 1 & 2 & 3 \\ 2 & 5 & 7 \\ -2 & -4 & -5 \end{bmatrix}$

14.2 $x = 1.00006$
 $y = 2.00004$
 $z = 3$

(Den exakta lösningen
är

$x = 1$
 $y = 2$
 $z = 3$

men på grund av avrundningar
vid flyttalsberäkningen i ABC 80
blir resultatet inte helt
korrekt.)

APPENDIX 4

Sakregister

ABC80

ABS	104	ERRCODE	160
ADD	141	EXP	104
argument	104	exponent	36
aritmetiskt uttryck	104		
ASC	86	felmeddelande	22, 159
ASCII-aritmetik	140	FIX	104
ASCII-kod	86	flexskiva	65
ASCII-tabell	88	flyttalsvariabel	39, 140
ATN	104	flödesschema	68
avrundning	107	flödesschemasympboler	79
		FOR...TO...	44
bandspelare	62	funktion	37, 103
bas	102	fält	56
BASIC	15	fördröjning	74
CHAIN	65	GET	72
CHR	87	GOSUB	75
citationstecken	8	GOTO	12
CLOSE	133	grafik	116
CLRDOT	119	grafiska tecken	117
COMP	141		
COS	104	heltalsexponent	102
cursor	4	heltalsvariabel	39, 140
		hjälpvariabel	48
data	15	huvudprogram	77
DATA	22		
datafil	132	IF...THEN...	23
datalagen	27	IF...THEN...ELSE	119
datalagring	130	indexerad variabel	51
decimalpunkt	35	INPUT	21
DEF FN...	105	INPUT ...,	133
DIM	51, 173	INPUTLINE	89
dimensionering	51	INPUTLINE ...,	138
DIV	141	INSTR	95
DOT	119	instruktion	15
drop out	63	INT	37, 104
ED	34	kassettbandspelare	62
ekvationssystem	171	kolontecken	74
element	164	kolumn	164
END	77	kommatecken	14
enhetsmatris	169	kontoutdrag	32
ERR	22	kontrolltangent	12

kopieringstangent 34
kurvritning 120

LEFT 85
LEN 90
LET 20
LIST 10
lista 51
listsortering 52
LOAD... 66
LOAD CAS: 64
LOG 104
logisk operator 57
loop 44
loopvariabel 44

markör 4
matris 164
matrisaddition 165
matrisberäkningar 164
matrisinvertering 169
matrismultiplikation 167
matrissubtraktion 167
MERGE... 66
MERGE CAS: 65
MID 85
MUL 141
multiplikation 35

NEW 11
NEXT 44
NOTRACE 156
NUM 95
numerisk sträng 140

ON ERROR GOTO... 142
ON...GOSUB... 142
ON...GOTO... 142
ON...RESTORE... 142
OPEN...ASFILE... 133

PI 104
plottning 109

potensuttryck 102
PREPARE...ASFILE... 132
PRINT 5
PRINT ..., 132
PRINT CUR 110
prioritet 38
procenttecken 39
programfel 150
programlagring 62

rad 164
radering 11
radertangent 5
radnummer 9
RANDOMIZE 45
rangordning 38
READ 22
redigering 34
REM 77
REN 78
RESTORE 146
RETURN 75
returtangent 8
RIGHT 85
RND 44, 104
RUN 9
RUN... 64
RUN CAS: 64

sats 15
SAVE 63
sekvensselement 68
semikolon 13
SETDOT 119
SGN 104
siffertecken 132
SIN 104
skifttangent 5
slumptal 44
SLUT GRAFIK 116
sol 20
sortering 47
SPACE 96
SQR 104

stapeldiagram 123
START GRAFIK 116
STEP 106
STOP 155
STOP LINE... 13
STRING 95
sträng 84
strängelement 84
strängtecken 39
strängvariabel 20, 39
SUB 141
subrutin 74
systemkommando 15

TAB 109
tabell 164
talvariabel 34
TAN 104
tangentbord 2
tiopotens 36
TRACE 153
TV-skärm 2

understrykning 95
upper case 5

VAL 93
valutatecken 20
variabel 20
variabelsortering 50
variabeltyper 39
vektor 51
videografikkarta 126
villkorselement 68

Den snabba utvecklingen inom elektroniken har medfört att kraftfulla datorer nu kan användas i många nya tillämpningar. Datorer, som idag kan köpas av privatpersoner för en kostnad som motsvarar priset på en färg-TV, har samma kapacitet som de datorer, som för ett tiotal år sedan kostade en miljon kronor. En helt ny användarkategori, utan tidigare erfarenhet av datorer, har därför ett stort behov av utbildning för att kunna möta den nya tekniken. För att utnyttja datorer på rätt sätt, samt framför allt ha glädje av den stora mängd färdiga program som finns tillgängliga, krävs vissa grundläggande kunskaper om hur datorer programmeras. Denna bok ger dessa kunskaper. Boken är avsedd att användas tillsammans med den svenska datorn **ABC80**. Formerna för utbildning kan variera, men denna bok kan med fördel användas både för självstudier och för gruppundervisning i gymnasieskolan, inom arbetsmarknadsutbildningen, för internutbildning på företag och för kurser i studieförbundens regi.

DIDACT

ISBN 91-7260-254-6

 **Liber**

Artikelnummer 40-11285-3