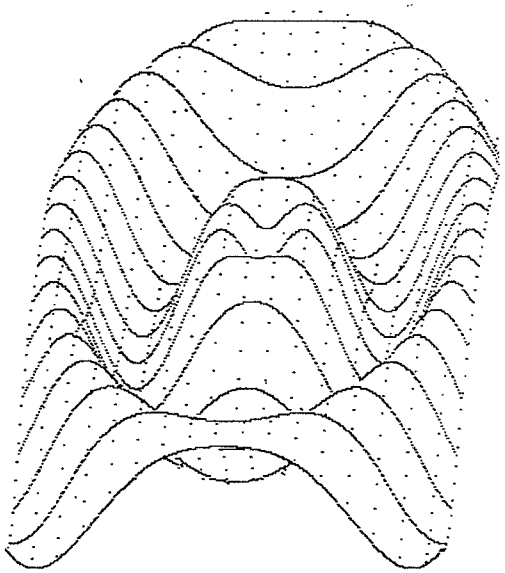
PROGRAM BITEN

82-83



NYA FÖRBÄTTRINGAR AV GRAPHICS MODE:

(Se s. 12!) Så här ser ytan

$z = 30 \sin[(x^2+y^2)/12]$ ut.

ISSN 0281-1146

Innehåll

Redaktören...

Efter mycken möda och stort besvär har Programbiten äntligen dimpt ner i din brevlåda. Tyvärr har den blivit några veckor försenad, vilket jag hoppas att du har överseende med. På tal om försenad: Texas nya guldägg CC-40, väntas inte komma ut i handeln förrän sensommaren. Det är i varje fall vad Texas Instruments i Sverige säger. Enligt vad ryktet förtäljer har CC-40 dock börjat säljas i U.S.A. och England.

Inte heller detta nummer nådde upp till 40-sidors gränsen. Trots detta tycker jag att numret innehåller en del läsvärda artiklar: Avancerade programmeringsmetoder del 4 om sorteringsmetoder är ett exempel. Något som heller inte får glömmas bort är att Utmaningen har kommit åter, nu med Robert Prins som redaktör. Detta borde borga för att kvaliteten bibehålls, Robert torde inte vara helt okänd för Programbitens regelbundna läsare. Robert är som bekant holländare, han välkomnar trots detta bidrag på svenska.

Uppropet i förra numret fick en hel del respons; vilket jag härmed vill tacka för. Det behövs ändå fler bidrag från medlemmarna, skicka därför in några av dina program, programmeringsknep et.c. Redaktionens nuvarande adress är: c/o Bo Nordlin, Stora Björnens Gata 70, 136 64 HANDEN.

TI PPC Notes har, liksom Programbiten, sedan årsskiftet en ny redaktör, vid namn Palmer O' Hansen. I sitt första nummer av tidningen gjorde han en jämförelse mellan två typer av programmerare: Den första kategorin, i vilken Maurice Swinnen (tidigare redaktören för TI PPC Notes) kunde placeras, innehåller perfektionister. "Helvetet hava ingen vrede lik Maurice Swinnens, när han upptäcker ett onödigt Nop". Den andra gruppen intar en motsatt ståndpunkt: "så länge ett program gör vad det ska göra, är det oväsentligt om programmet upptar 228 eller 179 programsteg". Personligen måste jag instämma med den sistnämnda gruppen, därför behöver ingen tänka: "Det är ingen idé att jag skickar in mitt program, det är inte tillräckligt bra."

Ytterligare en sak jag har funderat på är blandningen mellan program för TI 58 och för 59:an med skrivare. Har den varit den rätta? Eller har TI 59/PC 100 program fått för stort utrymme? Om så är fallet kan detta bero på att den "inre kretsen" (Lars Hedlund, Björn Gustavsson, Ingvar Magnusson och i viss mån mig själv) har bestått av ägare till TI 59/PC 100. Jag skulle gärna vilja ha dina åsikter om detta. Överhuvudtaget efterlyser jag mer "feedback" från läsekretsen; offra 1:80 i porto på att ge oss dina synpunkter.

Den uppmärksamme läsaren kan ha upptäckt att vi numera har ett licensnummer från Datainspektionen. En ny regel har nämligen införts: alla personregister, förutom de över släkt och vänner, som lagras i datorer måste registreras hos Datainspektionen. Detta nöje, som väl ingen har förstått vitsen med, tar Datainspektion ett par hundra kronor för, årligen! Nåväl, nu kan Programbitens medlemmar känna sig trygga; vårt adressregister, som både innehåller namn och adress, är vederbörligen registrerat.

Mässan i Sollentuna är omnämnd i alla datortidningar av rang. Programbiten vill inte vara sämre, utan kommer här med några reflexioner. Värmen och trängseln är de mest bestående intrycken. Därefter kommer nog det överväldigande antalet hemdatorer. Det är nog inte otroligt att det inom kort kommer att råda köparens marknad, med samma priskrig som redan nu finns i de anglosachiska länderna. På den portabla sidan fanns det en hel del nyheter: RONEX, som tidigare enbart sålde telefonsvarare, lanserar en CC-40 konkurrent till ett avgjort förmånligare pris. En svensk hand-dator i glada färger för 12000:-, skräddarsydd för svinuppfödare (Det är inte första april idag). I mitt tycke var dock en CASIO-produkt den mest intressanta. För cirka 3000 skulle CASIO sälja en portabel, batteridriven BASIC-dator med 32 KBytes ROM och lika mycket i RAM. LCD-displayen var 8 rader gånger 40 tecken stor, tangentbordet stort och bekvämt som en skrivmaskins. Ett VisiCalc liknande program ingick också i priset. Allt detta till ett pris understigande CC-40:s låter som rena himmelriket. Jag misstänker dock att CASIO kommer att få svårigheter med att hålla priset; en i princip identisk maskin (programvaran var olika) sälj på den amerikanska marknaden för nästan $1100, vilket i praktiken motsvarar ett svenskt pris på 15000 kr.

Om det händelsevis skulle finnas någon läsare som har stått ut med att läsa så här långt, vill jag tillönska den person en skön sommar. Väl mött i höst!

Bo Nordlin ■

DYRE MEDLEM

Som Du kanske sett av särskilt meddelande har användarna av hemdatorn TI-99/4A återvänt till föreningen Programbiten, och det medför att alla medlemmar åtminstone under resten av året får två medlemsblad: både "gamla" Programbiten och Nittinian.

Försäljningen av CC-40 har inte kommit i gång till det här numrets utgivning, och vi har faktiskt också legat ganska lågt med testandet. Enligt uppgift har man haft vissa problem med displayen, och modulanslutningens utförande har ändrats då det visat sig smutssamlande. Men dessa problem måste ändå anses som små och i augusti skall CC-40 finnas till försäljning.

Vi hoppas på en trevlig sommar och att eventuella regndagar resulterar i trevliga bidrag till Programbiten och Nittinian.

Lars Hedlund ordf

TÄVLING

Nu är det dags! Den största pristävlingen i föreningen Programbitens historia går nu av stapeln.

Uppgiften går ut på att skriva en TI-57 kompilator, dvs. ett program som översätter TI-57 program till TI 58/59. Den programmerare som aspirerar på första pris bör använda sig av Björn Gustavssons Programladdare (PB 81-4 s. 11), eller ett liknande program. Andra faktorer som är av betydelse kan vara att kompileringen går snabbt, att det kompilerade programmet är effektivt, att alla TI-57 funktioner översätts, att kompilatorn är lätt att använda m.m. m.m.

Vad blir då belöningen för pristagaren: ja, förutom äran, som naturligtvis är det väsentliga, blir vinnaren överöst med priser:

1 Matematikmodul
Ett antal programpaketter
Ett par böcker om Programmering av TI-räknare.
Programbitens T-shirt
Programmerigsblock
5 valfria program ur programbiblioteket
Precis som en känd svensk popgrupp säger vi "The winner takes it all", bara ett program får pris. Eventuell vinstskatt svarar vinnaren för.

För att största möjliga antal ska få möjlighet att deltaga är sista inskickningsdag den första november 1983. För de som inte har tillgång till Björns program och/eller TI 57:s instruktionsuppsättning, skickar vi dem mot svarsporto 1:80. ■

Ur TI 58/59 Software Club's medlemsblad saxar vi detta lilla hårdvarutips:
Sätt en röd lysdiod mellan ben 10 (katod) och 21 (anod) på displayen. Lysdiod kommer att lysa när 2nd har blivit nedtryckt (praktiskt vid långa Del eller Ins sekvenser!) Ben 21 är placerat längst bort till höger från displayens L-ben (dvs. "C" som lyser i displayen under programkörning). Robert Prins har prövat detta. Han använde en liten lysdiod (1x1x1 mm), som han applicerade längst ut till väster i displayen. Han uppmanar oss att använda en liten lödpenna, helst med låg effekt. Den här ombyggnaden upphäver säkerligen all garanti, var därför försiktig!

****
* LICENSNUMMER *
* 82100489 *
****

PROGRAMBITEN och FÖRENINGEN PROGRAMBITEN
c/o Hedlund
Årstavägen 27 6 tr
121 68 JOHANNESHOV

Tryck: K-TRYCK AB STOCKHOLM

UTMANINGEN!

Redaktörens adress:
Robert Prins
Alfred Nobellaan 112
NL-3731 DX De Bilt
Holland

Ja, ni lata (fyll i resten själv), det är tur att Lars Hedlund råkar vara en god vän till mig, och vem kan neka en god vän en tjänst? Så välkomna tillbaka till Utmaningen. Jag hoppas att jag kan bibehålla den höga standard som Björn Gustavsson har skapat. För att kunna göra detta behöver jag er hjälp: sänd in era problem, men ännu viktigare: era lösningar. Varför denna betoning på lösningar? Titta på tabellen över samtliga 37 Utmaningar; bara två (nummer 12 och 19) av de 20 första lösningarna är fortfarande olösta, men 10 av de senaste 17 saknar lösning. (Nåväl, låt oss säga nio, här nedan finns nämligen en lösning till det berömda problemet Damerna på schackbrädet, Utmaning 28).

NUMERISK INTEGARATION (30)

Ingen av er har gjort en lösning till det här problemet. Här har ni lite hjälp i form av en trevlig formel som ger extremt noggranna svar. Varför inte försöka göra ett program med den här formeln:

$$\int_a^b f(x)\,dx = \frac{b-a}{2n} \sum_{k=1}^{n} \sum_{j=1}^{2} \sum_{q=1}^{5} f\left[w_{2q-1} * \frac{(b-a)}{2n} * (-1)^j + \frac{a-b}{2n} + a + \frac{k(b-a)}{n}\right] w_{2q} + E(x)$$

$E(x) \approx f^{20}(x)$!

Faramet rarna är

a	nedre gräns
b	övre gräns
c	antal delintervall, valfritt heltal!

w_1	0.1488743389816	w_6	0.2190863625160
w_2	0.2955242247148	w_7	0.8650633666890
w_3	0.4333953941292	w_8	0.1494513491506
w_4	0.2692667193100	w_9	0.9739065285172
w_5	0.6794095682990	w_{10}	0.0666713443087

Försök att få plats med programmet på en kortsida, inklusive alla w! (Det är möjligt att göra det här programmet kortare än ML-19) Nu till några nya Utmaningar:

38. FÖRBÄTTRAD ML - 3

Skriv en ny version av ML - 3, utan labels, prt:s, adv:s and "=". Körinstruktionerna bör vara desamma, men programmet bör inte överskrida 240 steg.

39. NUMERISK TILL ALFA KONVERTERARE

Gör ett program som genererar skrivkoden för bokstäverna i alfabet från talen 1 -26. Försök att undvika att använda minnen och tester, se slutet av programmet Cirkeldiagram i PB 82-4 för ledtrådar.

40. BACHET UTTRYCK

Någon gång mellan 1581 och 1638 bevisade den franske vetenskapsmannen Claude Gaspard Bachet de Meziriac att det är möjligt att uttrycka varje heltal med hjälp av 3 upphöjt till olika dignitet och plus- och minustecknen. Min uppgift: skriv ett program som kan ta fram Bachet uttryck av samtliga heltal mindr än 10^{13}.
Ett par exempel:

$123 = 3^5 - 3^4 - 3^3 - 3^2 - 3^1$

$547 = 3^6 - 3^5 + 3^4 - 3^3 + 3^2 - 3^1 + 3^0$

$10^{13} - 1 = 3^{27} + 3^{26} - 3^{24} + 3^{23} + 3^{22} - 3^{21} -$
$- 3^{17} + 3^{16} - 3^{15} + 3^{14} - 3^{13} - 3^{12} + 3^{11} -$
$- 3^{10} + 3^8 - 3^7 + 3^6 - 3^5 + 3^4 + 3^3 + 3^2$

DAMERNA PÅ SCHACKBRÄDET (28)

Den här lösningen, som jag själv har skrivit, är baserad på en artikel i Byte, årgång 1979, nr. 2, sidorna 146-148. I artikeln omformuleras problemet på följande vis: "Hitta alla åtta-siffriga tal på formen $D_1D_2...D_8$, bestående av siffrorna 1 - 8, så att absolutvärdet av differensen av något sifferpar (D_a & D_b) inte är lika med absolutvärdet av differensen mellan siffrornas relativa position i talet, dvs. $|D_a - D_b| \neq |a - b|$."

TI 58/59 programmet behöver ungefär 8 timmar för att hitta samtliga 92 lösningar (i Fast Mode). Vid den här punkten kommer kanske många att ställa sig frågande och undra: Hur använder vi Fast Mode?
Låt oss börja med "Vanlig Mode"; tryck bara A och alla lösningar skrivs ut på printern och visas med hjälp av Fause i displayen. Om du inte har en skrivare, byter du förslagsvis ut steg 79 mot R/S.
Men hur är det nu med Fast Mode? Det är nästan lika enkelt. Tryck RST . 1 ×x x^2 1/x Stf Ind 7 (inte 07) Inv och programmet kommer att starta direkt för att stanna 8 timmar senare. Göra samma ändring som tidigare om du inte har skrivare.

Jag även tagit med ett SR-56 program, vilket är ursprungsprogrammet, med Utmaningen att göra om programmet för TI-57, om det är möjligt. Utelämna steg 90 - 92 om du vill, de räknar bara antal lösningar (Dsz utförs på register 0, de följande två stegen är hoppadressen).

```
000  92 RTN    023  39  39    046  80  80    069  97 DSZ
001  76 LBL    024  29 CP     047  97 DSZ    070  09  09
002  11  A     025  53  (     048  09  09    071  00  00
003  47 CMS    026  73 RC*    049  00  00    072  58  58
004  69 DP     027  00  00    050  25  25    073  65  ×
005  20  20    028  75  -     051  61 GTO    074  93  .
006  08  8     029  73 RC*    052  00  00    075  09  9
007  72 ST*    030  09  09    053  04  04    076  94 +/-
008  00  00    031  54  )     054  08  8     077  54  )
009  43 RCL    032  67  EQ    055  42 STO    078  66 PAU
010  00  00    033  00  00    056  09  09    079  99 PRT
011  42 STO    034  80  80    057  53  (     080  97 DSZ
012  09  09    035  53  (     058  73 RC*    081  40 IND
013  32 X:T    036  50 I×I    059  09  09    082  00  00
014  09  9     037  85  +     060  65  ×     083  00  00
015  67  EQ    038  43 RCL    061  43 RCL    084  09  09
016  00  00    039  09  09    062  09  09    085  97 DSZ
017  54  54    040  75  -     063  22 INV    086  00  00
018  01  1     041  43 RCL    064  28 LOG    087  00  00
019  67  EQ    042  00  00    065  52  EE    088  80  80
020  00  00    043  54  )     066  22 INV    089  00  0
021  04  04    044  67  EQ    067  57 ENG    090  81 RST
022  69 DP     045  00  00    068  85  +
```

TI 58/59

```
00   38  CMS    25   33  STO    50   94   =     75   34  RCL
01   00   0     26   03   3     51   37  X=T    76   01   1
02   32  X:T    27   34  RCL    52   06   6     77   64   ×
03   34  RCL    28   01   1     53   03   3     78   01   1
04   00   0     29   33  STO    54   28  I×I    79   00   0
05   35  SUM    30   04   4     55   74   -     80   74   -
06   01   1     31   01   1     56   34  RCL    81   29  INT
07   92   .     32   35  SUM    57   03   3     82   33  STO
08   01   1     33   03   3     58   94   =     83   00   0
09   30  PRD    34   34  RCL    59   12  INV    84   94   =
10   01   1     35   04   4     60   37  X=T    85   33  STO
11   34  RCL    36   37  X=T    61   03   3     86   01   1
12   02   2     37   00   0     62   01   1     87   22  GTO
13   74   -     38   03   3     63   27  DSZ    88   06   6
14   08   8     39   64   ×     64   02   2     89   03   3
15   33  STO    40   01   1     65   04   4     90   01   1
16   00   0     41   00   0     66   01   1     91   35  SUM
17   94   =     42   74   -     67   12  INV    92   09   9
18   37  X=T    43   12  INV    68   35  SUM    93   34  RCL
19   09   9     44   29  INT    69   02   2     94   01   1
20   00   0     45   33  STO    70   34  RCL    95   41  R/S
21   01   1     46   04   4     71   02   2     96   22  GTO
22   35  SUM    47   74   -     72   37  X=T    97   07   7
23   02   2     48   34  RCL    73   09   9     98   05   5
24   00   0     49   00   0     74   09   9     99   41  R/S
```

SR-56

1. EXPONENTEN
   80-2/5, 80-3/4

2. DUBBEL PRECISION
   80-2/5, 81-1/6, 81-3/5, 82-2/4-6, 82-3/7-9

3. PRIMTAL
   80-2/5, 80-3/5-7, 80-4/4-5, 81-1/6-7, 81-2/4, 81-3/6, 81-4/6-8

4. KONSTANTEN e
   80-2/5, 80-3/4-5, 80-4/5-7, 81-1/8

5. SKRIVARKOD
   80-3/6, 81-3/4-5, 81-4/7

6. HIR-STACKEN
   80-3/6, 81-1/8

7. PERMUTATIONER OCH KOMBINATIONER
   80-3/6-7, 81-1/8-9

8. EKVATIONSSYSTEM
   80-4/7, 81-1/9-10, 81-2/4-6

9. EKVATIONEN f(x)=0
   81-2/7, 82-1/6-8, 82-3/9-10

10. LABYRINTEN
    81-2/7-8, 81-3/4, 81-4/6-8

11. DSE-FUNKTION
    81-3/6, 81-4/4, 82-1/9

12. LUFFARSCHACK PÅ STOR SPELPLAN
    81-3/6

13. SORTERING
    81-3/6, 81-4/4-5

14. 1/A + 1/B = 1/C
    81-3/7, 82-1/13-14, 82-2/7, 82-3/7

15. $1^1 + 2^2 + 3^3 + \ldots + 20^{20}$
    81-3/7, 81-4/5

16. SUPERPRIMTAL
    81-4/9, 82-1/9-11

17. SKRIVKODSTABELL
    81-4/8-9, 82-1/11-12, 82-2/7

18. a^x mod n I DUBBEL PRECISION
    81-4/9, 82-1/12-13

19. Mr Magic
    82-1/14-15

20. AOS PA HP-41C
    82-1/15, 82-2/7

21. PRAKTISK SKRIVKODSTABELL
    82-1/16

22. PATIENSLÄGGARE
    82-1/16

23. PRIMTALSTEST
    82-1/16, 82-2/6-7

24. TRAFIKRÄKNING
    82-1/16

25. REGISTERBLOCKHANTERING
    82-1/16, 82-4/6

26. CIRKELDIAGRAM
    82-1/16, 82-3/4-6, 82-4/4-6

27. PASKENS DATUM
    82-2/8

28. DAMERNA PA SCHACKBRÄDET
    82-2/8

29. BERÄKNING AV 3.14159265358979323846264 33...
    82-2/8

30. NUMERISK INTEGRATION
    82-2/8-9

31. PARTIALBRAKSUPPDELNING
    82-2/9

32. INV PGM 20
    82-2/9, 82-4/8-9

33. **FLERA PROGRAM PA ETT KORT**
82-2/9, 82-4/6-8

34. **FÖRENKLAD SNABBRITARE**
82-3/10

35. **RUBIKS KUB**
82-4/9, 83-1/21-23 + 29

36. **KOMBINATIONER**
82-4/9

37. **"MERGE"-instruction**
82-4/9

Jag hoppas att få se många lösningar, skicka dem på enkel svenska och utan magnetkort till:
Robert AH Prins
Alfred Nobellaan 112
3731 DX DE BILT
HOLLAND

Lycka till.

BERÄKNING AV π (29)

(Detta avsnitt har lagts till av redaktionen i Stockholm och har således inte redigerats av Robert Prins. Text Lars Hedlund efter TI PPC Notes V7N4/5P27 och V8N1P21.)

Den formel som Bob Fruit rekommenderar - $\pi = 16*\arctan(1/5) - 4*\arctan(1/239)$ - går lätt att verifiera om man så vill genom att man använder formlerna $\tan(\alpha+\beta) = (\tan\alpha+\tan\beta)/(1-\tan\alpha\tan\beta)$ och dess specialfall $\tan 2\alpha = 2\tan\alpha/(1-\tan^2\alpha)$. För serieutvecklingen använder Bob serien $\arctan x = x - x^3/3 + x^5/5 - x^7/7 + \ldots$

Bob Fruit publicerade själv våren 1982 en lösning i TI PPC Notes, där han räknade fram 460 decimaler i fast mode på 6 timmar och 18 minuter.

I vintras kom emellertid vår gamle vän Dejan Ristanović och meddelade TI PPC Notes att han granskat ett program av sin landsman Jovan Puzović där denne lyckats kringgå bristen på minnesutrymme genom att spela in mellanresultat på magnetkort och på detta sätt beräknat π med 1188 decimaler, dvs 99 register med 12 siffror. Och därmed var HP-folket (tillfälligt?) slagna - deras senaste resultat var 1160 siffror på 15 1/4 timmar. Jovans beräkningstid är emellertid avsevärt längre - hela åttio timmar i fast mode!!! Fast mode anropas f ö på det kortaste sätt som någonsin förekommit i ett program - med hjälp av flaggregistret och tangenterna "7 EE".

Program A beräknar arctan(1/5) på 62 timmar i fast mode och program B arctan(1/239) på 18 timmar. Båda dessa program som är identiska t o m steg 156 har en lucka mellan detta steg och steg 202 för att "StFlg Ind" skall komma på de sista stegen i uppdelningen och fast mode därmed anropas på nämnda enkla sätt.

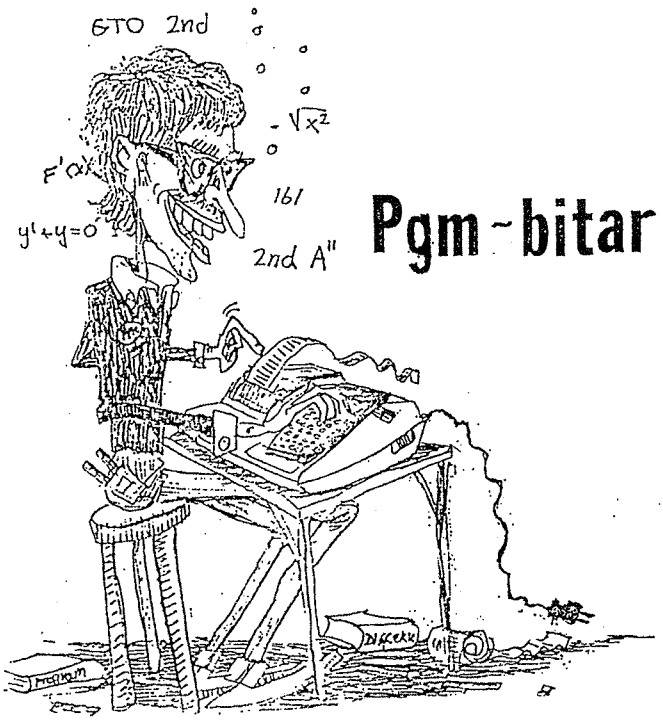
Alla tre programmen spelas in i normaluppdelning.

Köranvisningar

1. Stäng av och sätt på räknaren. Läs program A först. Tryck A så blinkar displayen efter en stund. Ta inte bort blinkningen utan tryck "7 EE".
2. Efter ungefär 62 timmar visar displayen noll. Gå ur fast mode med RST och spela in block 1 - 4 och märk kortsidorna A1 - A4.
3. Stäng av och sätt på räknaren. Läs program B. Tryck B. Ta inte bort blinkningen utan tryck "7 EE".
4. Efter ungefär 18 timmar visar displayen noll. Gå ur fast mode med RST och spela in block 1 - 4 och märk kortsidorna B1 - B4.
5. Stäng av och sätt på räknaren. Läs program C. Tryck A. Displayen visar -3. Läs kortsida A4 och tryck R/S. Displayen visar -2. Läs kortsida B4 och tryck R/S. Efter några minuter börjar skrivaren skriva decimalerna 841 - 1188. Räknaren stannar med blinkande display.
6. Tryck B. Displayen visar -3. Läs kortsida A3 och tryck R/S. Displayen visar -2. Läs kortsida B3 och tryck R/S. Nu skrivs decimalerna 481 - 840.
7. Tryck C. Som ovan men läs i stället kortsidor A2 och B2. Nu skrivs decimalerna 121 - 480.
8. Tryck D. Som ovan men läs nu kortsidor A1 och B1. De 120 första decimalerna skrivs ut (utan heltal, alltså 141592...).

Den gemensamma delen av program A och B gör följande:
1. Eftersom 1705 är nämnaren i den högsta termen med önskad noggrannhet i program A (501 i program B) beräknas 1/1705 (1/501) med 1188 decimaler (steg 012 - 059).
2. Minskar nämnaren med 2 och byter tecken (steg 060 - 078).
3. Multiplicerar med 1/25 resp. $1/239^2$ ("x²").
4. Som 1, dvs -1/1703 (-1/499) beräknas och adderas.
5. Som 2.
6. Som 3. Registren innehåller nu $(x^2/1705 - 1/1703)x^2 = x^4/1705 - x^2/1703$ (motsv)
7. Som 1 etc.

Till sist multipliceras med 1/5 (eller 1/239, "x") varefter registren innehåller summan av serien $x - x^3/3 + x^5/5 - x^7/7 + \ldots$



Pgm-bitar

CONNY BONET:

Man matar in n, x, p och eventuellt k; programmet räknar sedan ut sannolikheten P(p) att ett utfall inträffar x gånger av n försök. P är sannolikheten för x att inträffa, k förklaras senare i exemplen här nedan. Programmet använder minnena R_1 - R_9. Observera att programmet inte stör vilande operationer.

KÖRINSTRUKTIONER

1. Mata in n, tryck A
2. Mata in x, tryck B
3. Om nödvändigt: mata in k, tryck A'
4. Mata in p, tryck C. I displayen visas nu sannolikheten P(p)

Efter körning måste x, k och p matas in på nytt om man vill försöka igen med samma värden.

EXEMPEL

Om du slår en tärning 10 gånger. Hur stor är sannolikheten att du får exakt 2 sexor?

Mata in	Tryck	resultat	kommentar
10	A	10	n
2	B	2	x
1/6	C	0.2907100492	p_i P(p)

Sannolikheten för att få en sexa på en tärning är 1/6.

Svar: Sannolikheten för att få exakt två sexor på tio slag är 29.07%.

Du har köpt sex stycken räknare; sannolikheten för att var och en av räknarna att de behöver repareras är 0.2, dvs. 20%. Vad är då sannolikheten att åtminstone två måste repareras?

Mata in	Tryck	resultat	kommentar
6	A	6	n
2	B	2	x
4	A'	4	k se not
0.2	C	0.34464	p_i P(p)

Sannolikheten är 34.46%.

Ett mynt kastas 100 gånger. Beräkna sannolikheten att antalet klave är minst 45 och högst 55.

Mata in	Tryck	resultat	kommentar
100	A	100	n
45	B	45	x
10	A'	10	k (k=55-45)
0.5	C	0.7287469759	P_i P(p)

Sannolikheten är 72.87%.

Chansen att få en krona vid ett kast är 0.5, dvs. 50%.

Not När det sägs att åtminstone 2 räknare måste repareras, menas att 2 - 6 kan behöva repareras. Sannolikheten P(p) blir då summan av alla dessa, dvs. n = 6; x = 2, 3, 4, 5 och 6 samt p = 0.2. Därför blir k lika med 6 - x = 6 - 2 = 4. I det sista exemplet repeteras sekvensen till och med x = 45 + k, där n och p ärkonstanter, medan x ökar från 45 till 55.

```
000  76 LBL    024  65  ×     048  00  0     059  76 LBL
001  13  C     025  53  (     049  48 EXC     060  11  A
002  42 STO    026  01  1     050  05  05     061  42 STO
003  09  09    027  75  -     051  92 RTN     062  07  07
004  43 RCL    028  43 RCL     052  76 LBL     063  92 RTN
005  07  07    029  09  09     053  16 A'      064  76 LBL
006  36 PGM    030  54  )     054  42 STO     065  12  B
007  16  16    031  45 Y^x     055  06  06     066  42 STO
008  11  A     032  53  (     056  69 DP      067  08  08
009  43 RCL    033  43 RCL     057  26  26     068  92 RTN
010  08  08    034  07  07     058  92 RTN
011  36 PGM    035  75  -
012  16  16    036  43 RCL
013  12  B     037  08  08
014  53  (     038  54  )
015  36 PGM    039  54  )
016  16  16    040  44 SUM
017  15  E     041  05  05
018  65  ×     042  69 DP
019  43 RCL    043  28  28
020  09  09    044  97 DSZ
021  45 Y^x    045  06  06
022  43 RCL    046  00  00
023  08  08    047  04  04
```

CHECKSUM:

BANK 1
7317439801.

PROG
7317439801.

Angående spelet Psycho-Logic :
Vid provspel upptäckte jag att resultatet av $3-3^1$ blev 2 EE-12 , alltså icke noll. Mitt förslag för att åtgärda felet är följande fom steg 458;

(RC✱ 14 Y^x RC✱ 15 + , 1) INT RTN.
alltså tilläggs 6 steg

Ang utskrift av program på skrivmaskin :
Stjärnan vid indirekta instruktioner kan skrivas genom X och _ som tex. RC✱ . Det betyder att man får backa ett steg samt dra ner papperet i valsen ett klick. Det fungerar tillfredställande på dom skrivmaskiner jag har testat det på.
62 PG✱ 63 EX✱ 64 PD✱
72 ST✱ 73 RC✱ 74 SM✱
83 GO✱ 84 OP✱

P2PG Sven-Arne Wallin

VÄGEN TILL 13 RÄTT

JAN BJÖRKLUND

Programbitens läsare har bjudits många möjligheter att bli rika på tips, här nedan ges ytterligare en chans. Programmet bygger på ett matchvärderingssystem hämtat ur Carl-Axel Ericsons bok Spela tippa vinn. Matchvärderingen ger, på en TI-59, med utgångspunkt från ligatabell och resultatet från de senaste omgångarna, ett värde för varje match. Det värdet indikerar vilka matcher man bör ta ut som säkra och vilka som bör garderas. Ett högt värde visar på trolig hemmaseger, ett värde nära noll tolkas som en osäker match som bör garderas. Kraftigt negativa värden anger att bortaseger är en trolig utgång av matchen.

PROGRAM OCH DATAINMATNING

Mata in programmet i normaluppdelning. Övergå sedan till 10 Op 17 och lagra registerinnehållet. Återgå till normaluppdelning och spela in block 1 och 2.

KÖRINSTRUKTION

1.	Initiering	A'
2.	Mata in hemmalagets poäng	A
3.	Mata in bortalagets poäng	B
4.	Om matchen är ett derby	C
5.	Hemmalagets form: mata in hemmalagetspoäng de fem senaste omgångarna.	D
6.	Bortalagets form: mata in antalet tagna poäng under de fem senaste matcherna.	E

När punkt sex är utförd kommer matchvärdet fram i displayen. För nytt matchvärde gå till punkt 2.

EXEMPEL PÅ ANVÄNDNING

Följande tabeller är hämtade ur Expressen den 15/3 -83.

Eftersom man i England får 3 poäng för varje vunnen match istället för normalt 2 poäng måste man, när kupongen innehåller engelska matcher, korrigera tabbelvärdena genom att dra bort 1 poäng för varje vunnen match.

DIVISION I

	HEMMA				BORTA					SAMMANLAGT				
Liverpool	14	2	0	50-13	7	4	3	23-11	30	21	6	3	73-24	69
Watford	12	2	1	37-11	5	2	8	18-23	30	17	4	9	55-34	55
Manchester U	9	5	0	25- 6	5	4	6	14-17	29	14	9	6	39-23	51
Aston Villa	12	2	2	35-14	3	1	10	10-25	30	15	3	12	45-39	48
Nottingham	9	3	3	24-11	4	4	7	18-26	30	13	7	10	42-37	46
West Bromwich	8	5	2	30-14	3	6	7	12-23	31	11	11	9	42-37	44
Coventry	10	4	2	26- 9	2	3	9	14-30	30	12	7	11	40-39	43
Southampton	9	3	3	26-14	3	4	8	14-30	30	12	7	11	40-44	43
Ipswich	8	2	5	28-17	3	6	6	19-20	30	11	8	11	47-37	41
Everton	8	6	1	30-14	3	3	10	16-24	30	11	8	11	46-38	41
West Ham	9	2	4	31-17	4	0	10	14-28	29	13	2	14	45-45	41
Tottenham	9	4	2	32-13	2	4	9	8-28	30	11	8	11	40-41	41
Stoke	9	3	3	26-17	3	2	10	15-30	30	12	5	13	41-47	41
Notts Co	9	2	5	28-19	3	2	11	17-38	32	12	4	16	45-57	40
Sunderland	7	5	3	24-15	3	4	8	12-29	30	10	9	11	36-44	39
Arsenal	7	4	3	22-14	3	4	7	12-20	28	10	8	10	34-34	38
Manchester C	8	5	3	23-15	2	3	11	17-40	32	10	8	14	40-54	38
Luton	4	6	4	24-21	4	4	7	24-35	29	8	10	11	48-57	34
Swansea	8	2	5	25-17	0	5	11	15-30	31	8	7	16	40-47	31
Norwich	7	2	5	20-13	1	4	10	11-36	29	8	6	15	31-48	30
Brighton	6	5	4	19-16	1	2	12	8-40	30	7	7	16	28-56	28
Birmingham	5	5	4	19-17	0	7	7	5-22	28	5	12	11	24-39	27

DIVISION II

Wolverhampton	13	1	2	37-12	5	6	4	18-21	31	18	7	6	55-33	61
Queens PR	11	2	2	34-11	7	3	5	18-16	30	18	5	7	52-26	59
Fulham	9	4	2	26-17	7	3	5	25-18	30	16	7	7	51-35	55
Leicester	8	2	6	28-14	6	2	6	23-19	30	14	4	12	51-33	46
Oldham	6	7	2	30-19	4	7	5	19-18	31	10	14	7	49-37	44
Newcastle	8	5	2	24-14	3	6	6	22-26	30	11	11	8	46-40	44
Shrewsbury	6	6	3	15-11	6	2	8	25-30	31	12	8	11	40-41	44
Sheffield W	7	8	2	27-17	4	4	6	17-19	29	11	10	8	44-36	43
Barnsley	8	6	1	30-16	3	4	8	14-24	30	11	10	9	44-40	43
Leeds	6	6	2	20-14	3	9	4	18-21	30	9	15	6	38-35	42
Grimsby	9	5	2	29-17	3	1	11	13-33	31	12	6	13	41-50	42
Blackburn	7	6	2	28-16	4	3	9	14-26	31	11	9	11	42-42	42
Chelsea	8	5	2	29-13	2	3	11	16-33	31	10	8	13	45-46	38
Bolton	9	1	5	27-19	1	7	8	11-25	31	10	8	13	38-44	38
Charlton	8	2	5	27-23	2	4	9	18-40	30	10	6	14	45-63	36
Rotherham	5	5	6	16-21	3	6	6	17-24	31	8	11	12	33-45	35
Crystal P	7	5	3	22-15	1	5	9	9-23	30	8	10	12	31-38	34
Carlisle	7	3	5	31-23	1	6	9	20-34	31	8	9	14	51-57	33
Middlesbrough	6	6	4	19-26	2	6	8	15-34	31	7	12	12	34-59	33
Cambridge	8	5	2	21-13	0	3	12	10-33	30	8	8	14	31-46	32
Burnley	7	3	5	30-19	1	2	10	14-30	28	8	5	16	44-49	29
Derby	4	7	4	20-21	1	6	7	14-24	29	5	13	11	34-45	28

Liverpool	5	4	1	0	13- 3 13
Southampton	5	4	1	0	11- 3 13
Aston Villa	5	4	1	0	10- 5 12
Watford	5	4	1	0	11- 8 12
Newcastle	5	3	2	0	8- 4 11
Burnley	5	3	1	1	12- 6 10
Derby	5	2	3	0	10- 6 9
Fulham	5	2	2	1	4- 3 8
Wolverhampton	5	2	2	1	5- 8 8
Oldham	5	2	1	2	5- 2 7
Ipswich	5	2	1	2	8- 7 7
Chelsea	5	2	1	2	12-12 7
Grimsby	5	2	1	2	5- 5 7
Stoke	5	2	1	2	5- 8 7
Charlton	5	2	1	2	8-12 7
Sheffield W	5	1	3	1	5- 4 6
Leeds	5	1	3	1	8- 8 6
Tottenham	5	1	3	1	4- 6 6
Everton	5	1	2	2	6- 6 5
Blackburn	5	1	2	2	3- 4 5
Coventry	5	1	1	3	7- 9 5
Crystal P	5	1	1	3	4- 6 5
West Ham	5	1	1	3	3- 9 5
Nottingham	5	0	3	2	2- 4 3
Carlisle	5	0	3	2	5- 8 3
Manchester C	5	0	0	5	4-14 0

CHECKSUM:

BANK 1
9989032901

BANK 2
8845956500

PROG
8834989401

▶ 11

SOLENS UPP- OCH NEDGÅNG

av BERTIL WALLIN Karlskoga

Detta program beräknar tidpunkten för solens upp och nedgång. Dessutom beräknas timmvinkel t_s, tidsekvation E och deklination δ för solen vid aktuell tidpunkt. Som jämförelse kan nämnas att Svenska Almanackan ger E för 12^h medelsoltid och δ för solens meridianpassage för orten i fråga, dvs. när solen står i söder.

De ingångsdata som behövs är följande:

1. Ortens longitud l i tid samt inom vilken tidszon Z orten ligger. Följande ska beaktas vid beräkning av ortens longitud och zon:
   Z är positivt väst om Greenwich (London), annars negativt. Tidszon anges i heltal $\pm 0^h - 12^h$. Vidare motsvarar 1^h 15°.
   I de flesta fall går nollpunkt för zonen genom varje 15° och med en utsträckning av ± 7.5°, med andra ord: 0^h30^m. Se exempelvis Svenska Almanackan eller någon fickkalender beträffande olika länders Z (zon). För Stockholm som har koordinat l = 18°03'30" erhålles l i tid till $1^h12^m14^s$, Z blir -1. Värdet som ska matas in är alltså Z.hhmmss = -1.011214.
2. Ortens latitud ϕ, dvs. höjd över ekvatorn. Latitud inmatas i grader, positivt värde för norra halvklotet, negativt för södra halvklotet. För Stockholm som har ϕ = 59°21' norra halvklotet, inmatas 59.21.

3. Datum inmatas i format YYYY.MMDD.

Beräkningstid för upp och nedgång är cirka 2.5 minuter. Resultaten som programmet ger är:

tidpunkt: hh.mmss
timvinkel: GG°.mm'ss"
tidsekvation: hh.mmss
deklination: GG°.mm'ss"

Dessutom hänger två stycken decimaler med på sekunddelen.

Något om noggrannheten

Noggrannheten ligger ungefär i klass med Svenska Almanackan. Dessutom kan nämnas att överensstämmelse råder till åtminstone 1849, vilket jag har kunnat kontrollera. Detta borde betyda att programmet ger goda värden framåt i tiden inom ett lika stort intervall. Här kan också sägas att de teorier som används i detta program (Newcomb), ger ett fel på cirka 1 bågminut för år 1 i solens koordinater. Något optimistiskt skattat ger detta ett fel på, i runda tal, 0.5 timmar (ett orimligt stort värde, då en skattning ej bör avvika med mer än några få minuter från rätt värde). Programmet är på intet sätt optimerat, och vidare är konstruktionen ej av bästa slag. Detta beror på att vissa bitar kommer från andra program och på enklast möjliga sätt är "interfacade" med det övriga. Som exempel kan nämnas beräkning av det julianska dagtalet, vilket avsevärt skulle kunna förenklas.

Beräkningsgång och ekvationer

Knuten ligger vid att deklinationen för solen ej är känd. En beräkning görs för medelmiddag (12^h) för orten i fråga, detta något grova värde används sedan för den exaktare beräkningen. Detta förklarar också programmets långsamhet (två beräkningar för upp- respektive nedgång). Med upp/nedgång menas att solen befinner sig 35' under horisonten (horisontalreflektion). I steg 450 till 454 finns detta värde, som kan ändras för beräkning av borgerlig, nautisk eller astronomisk skymmning (6°, 12° eller 18° under horisont).
Ekvationer enligt kapitel 3.18 och 21 i Astronomical Formulae for Calculators, samt Astronomi och Astrofysik av Larsson och Leander, kapitel 2. Vidare förklaras några grundläggande begrepp nedan.
För storcirklar gäller enligt den sfäriska trigonometrin sambanden A och B:

A. Cosinus-teoremet

$$\cos(90 - h) = \cos(90 - \phi)\ \cos(90 - \delta) + \sin(90 - \phi)\ \sin(90 - \delta)\ \cos(t_s)$$

B. Sinus-teoremet

$$\frac{\sin(e)}{\sin(\delta)} = \frac{\sin(90)}{\sin(1)}$$

C. Tidsrelationer

Medelsoltid: $T_m = t_m + 12^h$
tidsekvationen: $E = t_s - t_m$
borgerlig tid: $T_b = T_m$ - tsk (fröken ur-tid)
tidsskillnad: ± tsk (- öst tidsmeridianen)

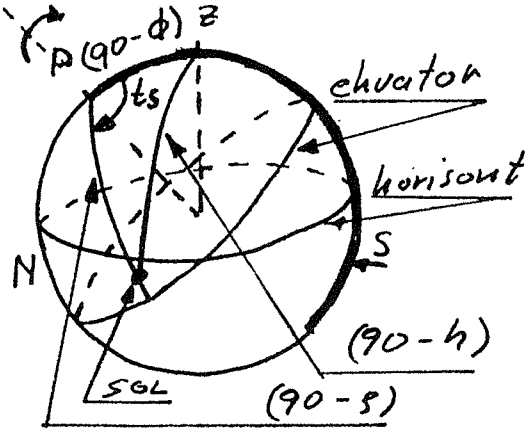
universal time: UT = T_b + Z

UT = medelsoltid vid Greenwich

Z = tidszon (Sverige har -1)

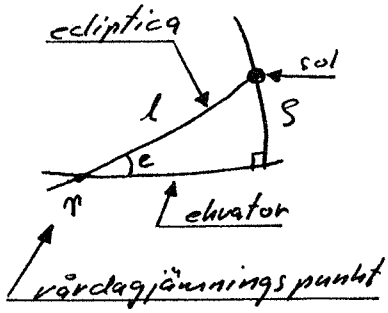
Dessa samband under punkt C ger det mycket användbara uttrycket:

$$t_s = t_b + E - tsk - 12$$



000	76	LBL	060	37	37	120	00	0	180	02	02	240	13	13	300	85	+	360	75	-	420	38	SIN
001	15	E	061	43	RCL	121	00	0	181	95	=	241	75	-	301	53	(	361	93	.	421	95	=
002	32	X:T	062	38	38	122	01	1	182	71	SBR	242	43	RCL	302	03	3	362	05	5	422	22	INV
003	04	4	063	55	÷	123	65	×	183	06	06	243	35	35	303	65	×	363	65	×	423	38	SIN
004	69	DP	064	01	1	124	53	(	184	02	02	244	33	X²	304	43	RCL	364	43	RCL	424	42	STO
005	17	17	065	00	0	125	43	RCL	185	42	STO	245	65	×	305	39	39	365	33	33	425	34	34
006	61	GTO	066	00	0	126	37	37	186	38	38	246	43	RCL	306	54	)	366	33	X²	426	29	CP
007	05	05	067	95	=	127	85	+	187	43	RCL	247	12	12	307	38	SIN	367	65	×	427	82	HIR
008	78	78	068	59	INT	128	01	1	188	35	35	248	75	-	308	65	×	368	53	(	428	16	16
009	42	STO	069	42	STO	129	54	)	189	65	×	249	43	RCL	309	43	RCL	369	04	4	429	61	GTO
010	39	39	070	35	35	130	54	)	190	43	RCL	250	35	35	310	19	19	370	65	×	430	04	04
011	59	INT	071	55	÷	131	59	INT	191	04	04	251	65	×	311	95	=	371	43	RCL	431	41	41
012	42	STO	072	04	4	132	85	+	192	85	+	252	43	RCL	312	42	STO	372	38	38	432	94	+/-
013	38	38	073	95	=	133	53	(	193	43	RCL	253	11	11	313	34	34	373	54	)	433	42	STO
014	43	RCL	074	59	INT	134	03	3	194	03	03	254	85	+	314	43	RCL	374	38	SIN	434	37	37
015	39	39	075	48	EXC	135	06	6	195	75	-	255	43	RCL	315	36	36	375	75	-	435	01	1
016	50	I×I	076	35	35	136	05	5	196	43	RCL	256	10	10	316	55	÷	376	05	5	436	94	+/-
017	22	INV	077	94	+/-	137	93	.	197	35	35	257	95	=	317	02	2	377	55	÷	437	49	PRD
018	59	INT	078	85	+	138	02	2	198	33	X²	258	42	STO	318	95	=	378	04	4	438	34	34
019	42	STO	079	02	2	139	05	5	199	65	×	259	36	36	319	30	TAN	379	65	×	439	43	RCL
020	37	37	080	95	=	140	29	CP	200	43	RCL	260	43	RCL	320	33	X²	380	43	RCL	440	37	37
021	65	×	081	44	SUM	141	65	×	201	05	05	261	39	39	321	42	STO	381	37	37	441	42	STO
022	01	1	082	35	35	142	43	RCL	202	75	-	262	38	SIN	322	33	33	382	33	X²	442	37	37
023	00	0	083	01	1	143	38	38	203	43	RCL	263	65	×	323	65	×	383	65	×	443	38	SIN
024	00	0	084	05	5	144	77	GE	204	35	35	264	53	(	324	53	(	384	53	(	444	65	×
025	75	-	085	08	8	145	01	01	205	65	×	265	43	RCL	325	02	2	385	02	2	445	43	RCL
026	59	INT	086	02	2	146	51	51	206	33	X²	266	14	14	326	65	×	386	65	×	446	34	34
027	42	STO	087	93	.	147	75	-	207	65	×	267	75	-	327	43	RCL	387	43	RCL	447	38	SIN
028	37	37	088	01	1	148	93	.	208	43	RCL	268	43	RCL	328	38	38	388	39	39	448	94	+/-
029	95	=	089	00	0	149	07	7	209	06	06	269	35	35	329	54	)	389	54	)	449	85	+
030	65	×	090	01	1	150	05	5	210	95	=	270	65	×	330	38	SIN	390	38	SIN	450	09	9
031	01	1	091	05	5	151	54	)	211	71	SBR	271	43	RCL	331	75	-	391	95	=	451	00	0
032	00	0	092	32	X:T	152	59	INT	212	06	06	272	15	15	332	02	2	392	65	×	452	93	.
033	00	0	093	43	RCL	153	95	=	213	02	02	273	75	-	333	65	×	393	01	1	453	03	3
034	95	=	094	38	38	154	42	STO	214	42	STO	274	43	RCL	334	43	RCL	394	08	8	454	05	5
035	42	STO	095	22	INV	155	34	34	215	39	39	275	35	35	335	37	37	395	00	0	455	88	DMS
036	36	36	096	77	GE	156	75	-	216	43	RCL	276	33	X²	336	65	×	396	55	÷	456	39	COS
037	32	X:T	097	01	01	157	43	RCL	217	07	07	277	65	×	337	43	RCL	397	89	π	457	95	=
038	68	NOP	098	02	02	158	25	25	218	75	-	278	43	RCL	338	39	39	398	55	÷	458	55	÷
039	55	÷	099	43	RCL	159	95	=	219	43	RCL	279	16	16	339	38	SIN	399	01	1	459	43	RCL
040	02	2	100	35	35	160	55	÷	220	35	35	280	54	)	340	85	+	400	05	5	460	34	34
041	04	4	101	85	+	161	43	RCL	221	65	×	281	85	+	341	04	4	401	95	=	461	39	COS
042	95	=	102	01	1	162	26	26	222	43	RCL	282	53	(	342	65	×	402	22	INV	462	55	÷
043	44	SUM	103	07	7	163	95	=	223	08	08	283	02	2	343	43	RCL	403	88	DMS	463	43	RCL
044	36	36	104	02	2	164	42	STO	224	75	-	284	65	×	344	37	37	404	42	STO	464	37	37
045	98	ADV	105	00	0	165	35	35	225	43	RCL	285	43	RCL	345	65	×	405	33	33	465	39	COS
046	03	3	106	09	9	166	43	RCL	226	35	35	286	39	39	346	43	RCL	406	43	RCL	466	95	=
047	32	X:T	107	09	9	167	00	00	227	33	X²	287	54	)	347	33	33	407	38	38	467	22	INV
048	43	RCL	108	04	4	168	85	+	228	65	×	288	38	SIN	348	65	×	408	85	+	468	39	COS
049	37	37	109	93	.	169	43	RCL	229	43	RCL	289	65	×	349	43	RCL	409	43	RCL	469	55	÷
050	77	GE	110	05	5	170	35	35	230	09	09	290	53	(	350	39	39	410	34	34	470	01	1
051	00	00	111	85	+	171	65	×	231	95	=	291	43	RCL	351	38	SIN	411	95	=	471	05	5
052	61	61	112	43	RCL	172	43	RCL	232	42	STO	292	17	17	352	65	×	412	42	STO	472	95	=
053	01	1	113	36	36	173	01	01	233	37	37	293	75	-	353	53	(	413	32	32	473	22	INV
054	22	INV	114	85	+	174	85	+	234	43	RCL	294	43	RCL	354	02	2	414	43	RCL	474	87	IFF
055	44	SUM	115	53	(	175	43	RCL	235	35	35	295	35	35	355	65	×	415	36	36	475	00	00
056	38	38	116	03	3	176	35	35	236	65	×	296	65	×	356	43	RCL	416	38	SIN	476	04	04
057	01	1	117	00	0	177	33	X²	237	33	X²	297	43	RCL	357	38	38	417	65	×	477	79	79
058	02	2	118	93	.	178	65	×	238	65	×	298	18	18	358	54	)	418	43	RCL	478	94	+/-
059	44	SUM	119	06	6	179	43	RCL	239	43	RCL	299	54	)	359	39	COS	419	32	32	479	42	STO

480	30	30	511	59	INT	541	71	SBR	571	22	INV	601	09	09	279.69668	00	0.00569	22
481	75	-	512	95	=	542	06	06	572	88	DMS	602	42	STO	36000.76892	01	0.00479	23
482	43	RCL	513	32	X:T	543	22	22	573	99	PRT	603	39	39	0.0003025	02	0.00256	24
483	33	33	514	87	IFF	544	61	GTO	574	22	INV	604	59	INT	358.47583	03	2415020.	25
484	88	DMS	515	01	01	545	05	05	575	58	FIX	605	55	÷	35999.04975	04	36525.	26
485	85	+	516	05	05	546	97	97	576	98	ADV	606	03	3	0.00015	05	0.276919398	27
486	01	1	517	23	23	547	22	INV	577	92	RTN	607	06	6	0.0000033	06	100.0021359	28
487	02	2	518	86	STF	548	86	STF	578	32	X:T	608	00	0	0.01675104	07	0.000001075	29
488	85	+	519	01	01	549	00	00	579	98	ADV	609	95	=	0.0000418	08		
489	53	(	520	61	GTO	550	98	ADV	580	99	PRT	610	59	INT	0.000000126	09	CHECKSUM:	
490	82	HIR	521	05	05	551	92	RTN	581	82	HIR	611	65	×	23.452294	10		
491	15	15	522	97	97	552	58	FIX	582	05	05	612	03	3	0.0130125	11		
492	59	INT	523	32	X:T	553	06	06	583	91	R/S	613	06	6	0.00000164	12	BANK 1	
493	75	-	524	75	-	554	22	INV	584	99	PRT	614	00	0	0.000000503	13	7345872803.	
494	53	(	525	82	HIR	555	88	DMS	585	88	DMS	615	95	=	1.91946	14	BANK 2	
495	82	HIR	526	15	15	556	99	PRT	586	82	HIR	616	22	INV	0.004789	15	2437578204.	
496	15	15	527	59	INT	557	43	RCL	587	06	06	617	44	SUM	0.0000014	16	BANK 3	
497	22	INV	528	95	=	558	30	30	588	91	R/S	618	39	39	0.020094	17	6739501905.	
498	59	INT	529	71	SBR	559	65	×	589	76	LBL	619	43	RCL	0.0001	18	BANK 4	
499	65	×	530	05	05	560	01	1	590	14	D	620	39	39	0.000293	19	1363942400.	
500	01	1	531	52	52	561	05	5	591	99	PRT	621	92	RTN	259.18	20	PROG	
501	00	0	532	22	INV	562	95	=	592	42	STO	622	01	1	1934.142	21	7886895312.	
502	00	0	533	86	STF	563	22	INV	593	31	31	623	02	2				
503	54	)	534	01	01	564	88	DMS	594	71	SBR	624	85	+				
504	88	DMS	535	87	IFF	565	99	PRT	595	06	06	625	82	HIR				
505	54	)	536	00	00	566	43	RCL	596	22	22	626	15	15				
506	94	+/-	537	05	05	567	33	33	597	43	RCL	627	59	INT				
507	95	=	538	47	47	568	99	PRT	598	31	31	628	95	=				
508	85	+	539	86	STF	569	43	RCL	599	61	GTO	629	32	X:T				
509	82	HIR	540	00	00	570	34	34	600	00	00	630	92	RTN				
510	15	15																



Programinstruktioner

Normal uppdelning vid in och avspelning av magnetkort. Vid inmatning av program och data: 4 Op 17.

1. Mata in longitud: ± Z.hhmmss E
2. Mata in latitud: ± GG°.mm'ss" R/S
3. Mata in datum: yyyy.MMDD R/S

Vid anslutning till printer sker utskrift av data för ned- respektive uppgång, se exempel. Då printer ej används knappas värdena fram för hand, genom att trycka R/S. OBS! efter sista värdet (deklination för uppgång) måste ett ytterligare R/S tryckas för initiering av eventuellt nytt datum.

4. nytt datum, samma ortskoordinater: YYYY.MMDD D

När printer ej används, ska följande ändringar göras i programlistningen:
Ändra till R/S i steg 556, 565, 568 och 573.
Dvs. Prt instruktioner ändras till R/S.

Kuriosa

Om ortskoordinater och tidpunkt antar vissa värden kommer utskrift/display att visa felindikering!! Men haven förtröstan, Du befinner Dig då troligen norr om polcirkeln ock kanske firar midsommar då solen varken går upp eller ned. Dock lär värdena för tidsekvation och deklination ha någon mening, då dessa icke har den minsta känsla för mänskliga traditioner!

Exempel

Stockholms horisont

-1.011214	long.
59.21	lat.
1983.0424	datum

ned

19.213986	tid
113.555769	t_s
0.014998	E^s
12.512662	dekl.

upp

4.115652	
-113.313670	
0.014303	
12.385465	

8▶ VÄGEN TILL 13 RÄTT

Match 1	Aston Villa-Coventry	Tangent
	H.P=48-15=33	A
	B.P=43-12=31	B
	H.F=13- 4= 9	D
	B.P= 5-1= 4	E
	Matchvärde:29	
Match 2	Ipswich-Nottingham	
	H.P=41-11=30	A
	B.P=46-13=33	B
	H.F= 7- 2= 5	D
	B.F= 3	E
	Matchvärde:13	
Match 3	Liverpool-Everton(derby)	
	H.P=69-21=48	A
	B.P=41-11=30	B
	Derby!	C
	H.F=13- 4= 9	D
	B.F= 5- 1= 4	E
	Matchvärde:35	

1	Aston Villa–Coventry .	29	
2	Ipswich–Nottingham .	13	
3	Liverpool–Everton ...	35	
4	Southampt.–Manch. C	52	
5	Watford–Tottenham..	32	
6	West Ham–Stoke	0	
7	Burnley–Newcastle ..	10	
8	Carlisle–Blackburn ...	2	
9	Charlton–Leeds		
10	Crystal P–Chelsea ...	1	
11	Derby–Sheffield W ...	5	
12	Grimsby–Wolverhamp.	13	
13	Oldham–Fulham	1	

000	76	LBL	025	95	=	050	03	3	075	59	INT	100	95	=		70060.	60	79051.62078	78
001	16	A'	026	42	STO	051	22	INV	076	55	÷	101	44	SUM	71060.70061		61	80051.61079	79
002	01	1	027	00	00	052	28	LOG	077	03	3	102	01	01	71059.70062		62	80050.6008	80
003	00	0	028	73	RC*	053	95	=	078	22	INV	103	43	RCL	72059.69063		63	81050.59081	81
004	69	DP	029	00	00	054	42	STO	079	28	LOG	104	01	01	72058.69064		64	81049.59082	82
005	17	17	030	22	INV	055	01	01	080	95	=	105	75	-	73058.68065		65	82049.58083	83
006	91	R/S	031	59	INT	056	91	R/S	081	44	SUM	106	22	INV	73057.68066		66	82048.58084	84
007	76	LBL	032	65	×	057	76	LBL	082	01	01	107	59	INT	74057.67067		67	83048.57085	85
008	11	A	033	02	2	058	13	C	083	91	R/S	108	65	×	74056.67068		68	15010.10015	86
009	42	STO	034	22	INV	059	93	.	084	76	LBL	109	03	3	75056.66069		69	22009.09022	87
010	00	00	035	28	LOG	060	00	0	085	15	E	110	22	INV	75055.6607		70	28008.08028	88
011	91	R/S	036	95	=	061	01	1	086	85	+	111	28	LOG	76055.65071		71	33007.07033	89
012	76	LBL	037	42	STO	062	44	SUM	087	08	8	112	95	=	76054.65072		72	36006.06036	90
013	12	B	038	01	01	063	01	01	088	06	6	113	59	INT	77054.64073		73	38005.05038	91
014	22	INV	039	91	R/S	064	91	R/S	089	95	=	114	91	R/S	77053.64074		74	41004.04041	92
015	44	SUM	040	76	LBL	065	76	LBL	090	42	STO				78053.63075		75	44003.03044	93
016	00	00	041	17	B'	066	14	D	091	00	00				78052.63076		76	46002.02046	94
017	43	RCL	042	06	6	067	85	+	092	73	RC*				79052.62067		77	48001.01048	95
018	00	00	043	00	0	068	08	8	093	00	00							50000.0005	96
019	77	GE	044	44	SUM	069	06	6	094	22	INV								
020	17	B'	045	00	00	070	95	=	095	59	INT								
021	94	+/-	046	73	RC*	071	42	STO	096	65	×								
022	85	+	047	00	00	072	00	00	097	02	2								
023	06	6	048	59	INT	073	73	RC*	098	22	INV								
024	00	0	049	55	÷	074	00	00	099	28	LOG								

3-D PLOT -2

ROBERT PRINS

Robert Prins från Holland var inte riktigt nöjd med 3-d plot, som vi publicerade i PB 83-1. Robert gjorde något åt det, han förkortade programmet avsevärt, han gjorde programmet cirka 20% snabbare, han förenklade proceduren med att skapa de båda hexkoderna, han gjorde om programmet så att det numera kan plotta flera figurer samtidigt. På det hela taget var det så stora förbättringar att vi måste ta in programmet på nytt.

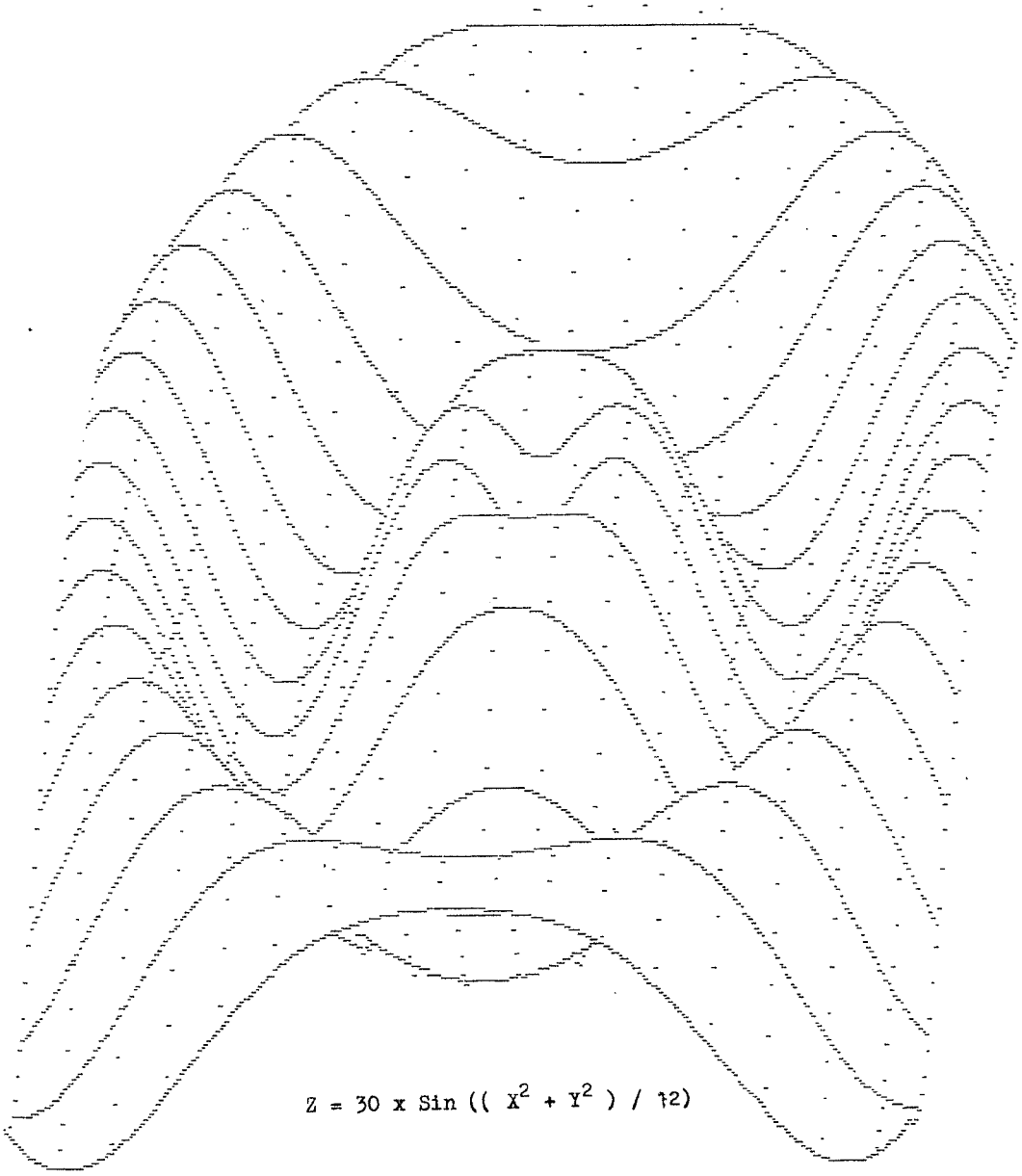
INSTRUKTIONER

Mata in input programmet, tryck A. Besvara frågorna som numera är på engelska.

Byt därefter till output-programmet. Skapa de två hexkoderna genom följande rutin: 10 Op 17 CLR GTO 016 Pgm 19 SBR 045 P/R LRN Ins SST (16 gånger) Ins LRN CLR RST och slutligen 6, 5 eller 4 Op 17. Mata in din(a) funktion(er) från steg 435. X-värdet finns i R38, Y i R39.

```
435:  f1  x:t  ppp GTO 293
ppp:  f2  x:t  qqq GTO 293
qqq:  et.c.
      :
      :
      fn  x:t RCL 29 GTO 293 (sista funktionen)
```

Starta programmet genom att trycka A.



3-d plot Input

```
000  76 LBL     049  00  0     098  03  3     147  42 STO     196  53  (     245  00  00     294  03  3     327  65  ×
001  10 E'      050  03  3     099  04  4     148  08  08     197  42 STO     246  03  3     295  06  6     328  43 RCL
002  69 DP      051  00  0     100  04  4     149  03  3      198  15  15     247  04  4     296  00  0     329  18  18
003  01  01     052  02  2     101  10 E'      150  07  7     199  65  ×     248  08  8     297  00  0     330  54  )
004  69 DP      053  04  4     102  44 SUM     151  01  1     200  43 RCL     249  44 SUM     298  00  0     331  65  ×
005  05  05     054  03  3     103  13  13     152  03  3     201  05  05     250  00  00     299  00  0     332  53  (
006  00  0      055  01  1     104  01  1     153  03  3     202  54  )     251  43 RCL     300  69 DP      333  43 RCL
007  91 R/S     056  10 E'      105  06  6     154  03  3     203  42 STO     252  18  18     301  02  02     334  05  05
008  99 PRT     057  42 STO     106  02  2     155  01  1     204  20  20     253  44 SUM     302  69 DP      335  55  ÷
009  92 RTN     058  19  19     107  04  4     156  07  7     205  42 STO     254  11  11     303  05  05     336  43 RCL
010  76 LBL     059  22 INV     108  02  2     157  03  3     206  11  11     255  43 RCL     304  53  (     337  26  26
011  11  A      060  44 SUM     109  01  1     158  06  6     207  53  (     256  11  11     305  53  (     338  85  +
012  47 CMS     061  05  05     110  02  2     159  10 E'      208  43 RCL     257  35 1/X     306  53  (     339  01  1
013  69 DP      062  04  4     111  00  0     160  42 STO     209  05  05     258  32 X:T     307  32 X:T     340  54  )
014  00  00     063  05  5     112  04  4     161  09  09     210  65  ×     259  43 RCL     308  65  ×     341  54  )
015  04  4      064  02  2     113  04  4     162  03  3      211  43 RCL     260  07  07     309  32 X:T     342  65  ×
016  04  4      065  00  0     114  10 E'      163  01  1     212  14  14     261  22 INV     310  43 RCL     343  43 RCL
017  02  2      066  03  3     115  42 STO     164  02  2     213  54  )     262  49 PRD     311  20  20     344  07  07
018  00  0      067  00  0     116  22  22     165  02  2     214  44 SUM     263  11  11     312  54  )     345  65  ×
019  03  3      068  01  1     117  42 STO     166  02  2     215  13  13     264  43 RCL     313  65  ×     346  43 RCL
020  00  0      069  03  3     118  28  28     167  07  7     216  03  3     265  09  09     314  53  (     347  09  09
021  02  2      070  04  4     119  01  1     168  01  1     217  01  1     266  22 INV     315  43 RCL     348  54  )
022  04  4      071  04  4     120  06  6     169  07  7     218  07  7     267  49 PRD     316  18  18     349  52 EE
023  03  3      072  10 E'      121  02  2     170  00  0     219  01  1     268  13  13     317  55  ÷     350  22 INV
024  01  1      073  44 SUM     122  04  4     171  00  0     220  00  0     269  53  (     318  43 RCL     351  57 ENG
025  10 E'      074  05  05     123  02  2     172  69 DP      221  00  0     270  43 RCL     319  22  22     352  59 INT
026  42 STO     075  04  4     124  01  1     173  02  02     222  00  0     271  14  14     320  85  +     353  99 PRT
027  16  16     076  06  6     125  02  2     174  04  4      223  00  0     272  65  ×     321  01  1     354  69 DP
028  22 INV     077  02  2     126  00  0     175  04  4      224  00  0     273  43 RCL     322  54  )     355  00  00
029  44 SUM     078  00  0     127  04  4     176  02  2      225  00  0     274  19  19     323  85  +     356  69 DP
030  18  18     079  03  3     128  05  5     177  00  0      226  69 DP     275  54  )     324  53  (     357  05  05
031  04  4      080  00  0     129  10 E'      178  04  4     227  02  02     276  44 SUM     325  00  0     358  92 RTN
032  04  4      081  02  2     130  42 STO     179  05  5     228  02  2     277  12  12     326  32 X:T
033  02  2      082  04  4     131  24  24     180  02  2     229  03  3     278  02  2
034  00  0      083  03  3     132  42 STO     181  00  0     230  02  2     279  01  1       CHECKSUM:
035  03  3      084  01  1     133  26  26     182  01  1     231  04  4     280  02  2
036  00  0      085  10 E'      134  02  2     183  03  3     232  01  1     281  00  0       BANK 1
037  01  1      086  42 STO     135  07  7     184  10 E'      233  06  6     282  04  4        6873247901.
038  03  3      087  12  12     136  02  2     185  42 STO     234  01  1     283  02  2       BANK 2
039  04  4      088  22 INV     137  04  4     186  14  14     235  06  6     284  01  1        4130502003.
040  04  4      089  44 SUM     138  03  3     187  38 SIN     236  01  1     285  03  3
041  10 E'      090  13  13     139  01  1     188  48 EXC     237  07  7     286  02  2       PROG
042  42 STO     091  04  4     140  01  1     189  14  14     238  10 E'      287  07  7        1003749904.
043  17  17     092  06  6     141  07  7     190  39 COS     239  94 +/-     288  69 DP
044  44 SUM     093  02  2     142  03  3     191  49 PRD     240  42 STO     289  01  01
045  18  18     094  00  0     143  06  6     192  24  24     241  00  00     290  04  4
046  04  4      095  03  3     144  10 E'      193  22 INV     242  03  3     291  01  1
047  05  5      096  00  0     145  42 STO     194  49 PRD     243  04  4     292  01  1
048  02  2      097  01  1     146  07  07     195  28  28     244  49 PRD     293  07  7
```

3-d plot Output

```
000  92 RTN     058  98 ADV     116  54  )     174  53  (     232  22 INV     290  44 SUM     348  32 X:T     392  02  2
001  76 LBL     059  97 DSZ     117  22 INV     175  53  (     233  44 SUM     291  10  10     349  01  1     393  01  1
002  11  A      060  09  09     118  44 SUM     176  43 RCL     234  23  23     292  92 RTN     350  32 X:T     394  85  +
003  61 GTO     061  00  00     119  23  23     177  23  23     235  43 RCL     293  42 STO     351  77  GE     395  04  4
004  00  00     062  40  40     120  43 RCL     178  75  -     236  26  26     294  30  30     352  40 IND     396  07  7
005  40  40     063  92 RTN     121  17  17     179  43 RCL     237  44 SUM     295  53  (     353  30  30     397  54  )
006  76 LBL     064  42 STO     122  61 GTO     180  05  05     238  25  25     296  53  (     354  29 CP      398  52 EE
007  12  B      065  25  25     123  01  01     181  54  )     239  61 GTO     297  32 X:T     355  22 INV     399  94 +/-
008  61 GTO     066  42 STO     124  56  56     182  55  ÷     240  02  02     298  65  ×     356  77  GE     400  22 INV
009  00  00     067  27  27     125  53  (     183  43 RCL     241  64  64     299  43 RCL     357  40 IND     401  57 ENG
010  22  22     068  43 RCL     126  32 X:T     184  24  24     242  32 X:T     300  39  39     358  30  30     402  82 HIR
011  76 LBL     069  16  16     127  55  ÷     185  85  +     243  42 STO     301  65  ×     359  53  (     403  05  05
012  13  C      070  42 STO     128  43 RCL     186  02  2     244  39  39     302  43 RCL     360  53  (     404  53  (
013  25 CLR     071  05  05     129  22  22     187  54  )     245  43 RCL     303  14  14     361  53  (     405  73 RC*
014  69 DP      072  42 STO     130  54  )     188  52 EE      246  21  21     304  75  -     362  52 EE      406  05  05
015  05  05     073  23  23     131  53  (     189  59 INT     247  42 STO     305  43 RCL     363  65  ×     407  55  ÷
016  74 SM*     074  43 RCL     132  59 INT     190  42 STO     248  38  38     306  12  12     364  04  4     408  82 HIR
017  90  90     075  10  10     133  65  ×     191  06  06     249  02  2     307  54  )     365  85  +     409  16  16
018  90 LST     076  32 X:T     134  53  (     192  53  (     250  05  5     308  55  ÷     366  01  1     410  22 INV
019  61 GTO     077  43 RCL     135  94 +/-     193  53  (     251  05  5     309  43 RCL     367  75  -     411  28 LOG
020  69 DP      078  18  18     136  65  ×     194  43 RCL     252  61 GTO     310  13  13     368  59 INT     412  33 X²
021  68 NOP     079  77  GE     137  43 RCL     195  21  21     253  04  04     311  54  )     369  42 STO     413  82 HIR
022  20 CLR     080  01  01     138  28  28     196  75  -     254  33  33     312  83 GD*     370  05  05     414  45  45
023  29 CP      081  27  27     139  85  +     197  43 RCL     255  43 RCL     313  00  00     371  54  )     415  54  )
024  01  1      082  53  (     140  43 RCL     198  05  05     256  22  22     314  22 INV     372  65  ×     416  53  (
025  34 FX      083  53  (     141  10  10     199  54  )     257  22 INV     315  87 IFF     373  05  5     417  22 INV
026  33 X²      084  32 X:T     142  55  ÷     200  55  ÷     258  44 SUM     316  04  04     374  82 HIR     418  59 INT
027  35 1/X     085  75  -     143  43 RCL     201  43 RCL     259  21  21     317  03  03     375  06  06     419  65  ×
028  22 INV     086  32 X:T     144  15  15     202  22  22     260  43 RCL     318  26  26     376  94 +/-     420  04  4
029  58 FIX     087  54  )     145  54  )     203  54  )     261  28  28     319  22 INV     377  75  -     421  54  )
030  86 STF     088  55  ÷     146  44 SUM     204  52 EE      262  44 SUM     320  86 STF     378  59 INT     422  59 INT
031  71  71     089  32 X:T     147  27  27     205  22 INV     263  27  27     321  04  04     379  82 HIR     423  22 INV
032  35 1/X     090  43 RCL     148  43 RCL     206  57 ENG     264  97 DSZ     322  82 HIR     380  36  36     424  67 EQ
033  54  )     091  15  15     149  22  22     207  59 INT     265  06  06     323  07  07     381  54  )     425  40 IND
034  80 GRD     092  54  )     150  85  +     208  44 SUM     266  02  02     324  82 HIR     382  65  ×     426  30  30
035  50 I×I     093  44 SUM     151  43 RCL     209  06  06     267  10  10     325  08  08     383  03  3     427  82 HIR
036  92 RTN     094  27  27     152  16  16     210  43 RCL     268  00  0     326  32 X:T     384  54  )     428  15  15
037  23 LNX     095  53  (     153  54  )     211  27  27     269  48 EXC     327  82 HIR     385  53  (     429  74 SM*
038  12  B      096  53  (     154  42 STO     212  32 X:T     270  01  01     328  18  18     386  59 INT     430  05  05
039  13  C      097  53  (     155  21  21     213  43 RCL     271  69 DP      329  32 X:T     387  85  +     431  83 GD*
040  97 DSZ     098  32 X:T     156  43 RCL     214  25  25     272  01  01     330  77  GE     388  01  1     432  30  30
041  08  08     099  55  ÷     157  10  10     215  77  GE     273  00  0     331  03  03     389  75  -     433  42 STO
042  00  00     100  43 RCL     158  56 DEL     216  02  02     274  48 EXC     332  46  46     390  50 I×I     434  29  29
043  40  40     101  24  24     159  56 DEL     217  42  42     275  02  02     333  32 X:T     391  65  ×
044  98 ADV     102  54  )     160  44 SUM     218  42 STO     276  69 DP      334  82 HIR
045  43 RCL     103  52 EE      161  23  23     219  39  39     277  02  02     335  17  17       CHECKSUM:
046  09  09     104  59 INT     162  32 X:T     220  43 RCL     278  00  0     336  22 INV
047  99 PRT     105  85  +     163  43 RCL     221  23  23     279  48 EXC     337  77  GE       BANK 1
048  43 RCL     106  01  1     164  20  20     222  42 STO     280  03  03     338  40 IND        1295671601.
049  13  13     107  54  )     165  77  GE     223  38  38     281  69 DP      339  30  30       BANK 2
050  44 SUM     108  65  ×     166  01  01     224  02  2     282  03  03     340  32 X:T        6241410603.
051  12  12     109  43 RCL     167  74  74     225  03  3     283  00  0     341  82 HIR       PROG
052  43 RCL     110  26  26     168  22 INV     226  00  0     284  48 EXC     342  07  07        7537082204.
053  07  07     111  65  ×     169  44 SUM     227  61 GTO     285  04  04     343  61 GTO
054  48 EXC     112  44 SUM     170  05  05     228  04  04     286  69 DP      344  03  03
055  08  08     113  25  25     171  32 X:T     229  33  33     287  04  04     345  48  48
056  42 STO     114  43 RCL     172  44 SUM     230  43 RCL     288  43 RCL     346  82 HIR
057  10  10     115  15  15     173  05  05     231  24  24     289  11  11     347  08  08
```

BRÅK-RÄKNING

Ulf Heiman

Det här är en Fast Mode version av Björn Gustavssons bråkräkningsprogram, publicerat i Programbiten 80-4. Programmet är skrivet för en 58:a, men det fungerar även på TI 59. Samma labels som i Björns program har använts, men körinstruktionerna skiljer sig något från varandra. För att programmet ska fungera måste fem 13-siffriga tal lagras i R25-R29 och uppdelningen måste vara 239.29 (239.89 på 59:an)

ANVÄNDA LABELS

A matar in första bråket i en beräkning
A' förkortar
B adderar
C subtraherar
D multiplicerar
E dividerar
In- och utdata har formen: täljare x:t nämnare.

KÖRINSTRUKTION

1. Täljare matas in, tryck x:t; mata därefter in nämnare och tryck A.
2. Andra bråkets täljare och nämnare matas in. Välj räkneoperation (label B - E). "42444." kommer att blinka i displayen.
3. Tryck 7 EE och beräkningen kommer att starta automatiskt.
4. Vid kedjeberäkningar: fortsätt från steg 2.
5. Vid förkortning: mata in bråket, tryck A'. När "42444." blinkar i displayen, tryck 7 EE.

Läsaren bör observera den eleganta initieringen av Fast Mode. Med ett 13-siffrigt tal i displayen hoppar programmet till sekvensen Fix 0 Stf Ind, där Ind finns på steg 239, det sista steget i programminnet. Programmet stannar då naturligtvis med blinkande display. När sedan användaren trycker 7 EE startar programmet; var programmet startar beror på det 13-siffriga talet. Närmare bestämt siffrorna 9 - 12 i det talet. Multiplicera talet som består av siffrorna 9 - 11 med 8, addera detta med siffra 12 och lägg slutligen till 1 och du har fått stegnumret där programmet startar. Som exempel kan vi ta register 25: 42444.00000802, siffrorna 9 - 11 utgör talet 8, vilket vi multiplicera med 8 och får 64. Siffra 12 är 0, efter addition har vi alltså fortfarande 64. Slutligen lägger vi till 1 och erhåller 65. Programmet startar på steg 65 om register 25 används som initieringskonstant.

De 13-siffriga talen laddas förslagsvis i registren genom att ta heltalsdelen + decimaldelen. T.ex. register 25: 42444 + .00000802 = STO 25.

REGISTERINNEHÅLL

R25	42444.00000802
R26	42444.00001622
R27	42444.00001512
R28	42444.00001862
R29	42444.00001772

```
000  91 R/S    027  36  36    054  43 RCL    081  48 EXC    108  43 RCL    135  04  04
001  76 LBL    028  76 LBL    055  04  04    082  00  00    109  02  02    136  32 X:T
002  16 A'     029  14  D     056  95  =     083  55  ÷     110  32 X:T    137  65  ×
003  42 STO    030  42 STO    057  32 X:T    084  43 RCL    111  43 RCL    138  43 RCL
004  11  11    031  11  11    058  65  ×     085  00  00    112  01  01    139  03  03
005  43 RCL    032  43 RCL    059  43 RCL    086  54  )     113  22 INV    140  85  +
006  25  25    033  28  28    060  03  03    087  94 +/-    114  58 FIX    141  61 GTO
007  61 GTO    034  61 GTO    061  95  ='    088  52 EE     115  42 STO    142  00  00
008  02  02    035  02  02    062  61 GTO    089  22 INV    116  03  03    143  54  54
009  36  36    036  36  36    063  00  00    090  52 EE     117  32 X:T    144  25 CLR
010  76 LBL    037  76 LBL    064  68  68    091  85  +     118  42 STO    145  43 RCL
011  12  B     038  15  E     065  25 CLR    092  82 HIR    119  04  04    146  11  11
012  42 STO    039  42 STO    066  43 RCL    093  12  12    120  99 PRT    147  32 X:T
013  11  11    040  11  11    067  11  11    094  95  =     121  81 RST    148  61 GTO
014  43 RCL    041  43 RCL    068  58 FIX    095  22 INV    122  25 CLR    149  01  01
015  26  26    042  29  29    069  00  00    096  67  EQ     123  43 RCL    150  54  54
016  61 GTO    043  61 GTO    070  42 STO    097  00  00    124  11  11    151  25 CLR
017  02  02    044  02  02    071  01  01    098  79  79    125  32 X:T    152  43 RCL
018  36  36    045  36  36    072  32 X:T    099  43 RCL    126  94 +/-    153  11  11
019  76 LBL    046  76 LBL    073  42 STO    100  00  00    127  32 X:T    154  32 X:T
020  13  C     047  11  A     074  02  02    101  50 I×I    128  61 GTO    155  65  ×
021  42 STO    048  42 STO    075  42 STO    102  22 INV    129  01  01    156  61 GTO
022  11  11    049  03  03    076  00  00    103  49 PRD    130  34  34    157  00  00
023  43 RCL    050  32 X:T    077  25 CLR    104  01  01    131  25 CLR    158  54  54
024  27  27    051  42 STO    078  32 X:T    105  22 INV    132  43 RCL
025  61 GTO    052  04  04    079  65  ×     106  49 PRD    133  11  11
026  02  02    053  91 R/S    080  53  (     107  02  02    134  49 PRD

236  58 FIX    238  86 STF
237  00  00    239  40 IND
```

CHECKSUM:

BANK 1
1224181302.

BANK 4
1079219500.

PROG
2303400802.

T R I m N I N G

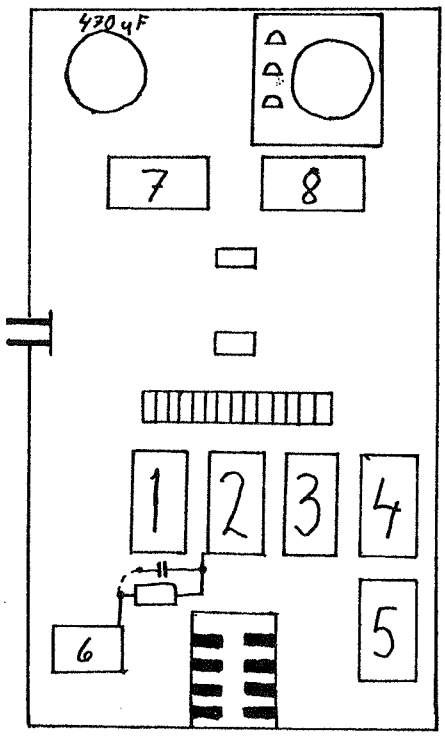
Lars Nilsson

Genom Lars Nilsson har vi fått reda på att det faktiskt är möjligt att "trimma" TI 58C. Trimning innebär att man byter ut en kondensator som styr räknarens interna klocka. Detta innebär att räknaren kör program och utför beräkningar snabbare än tidigare.
Det har varit känt länge att man på detta vis har kunnat öka HP-41:s hastighet med omkring 100%. Jag har hört talas om en lyckad "trimning" av en CASIO 702, i detta fall blev ökningen 270%! Den enda nackdelen med ett sådant förfarande torde vara att strömförbrukningen ökar (förutom att eventuell garanti går ut).
I detta fall är det Lennart Lundblad som har lyckats "trimma" sin 58, vi återger här nedan Lars Nilssons brev om hur Lennart Lundblad gick tillväga.

Den räknare som har använts är ganska ny och kretskortet ser inte riktigt ut som på de äldre räknarna. Spänningsomvandlaren är flyttad upp under displayen och de två kretsar som på gamla räknare låg ovanpå varandra (kretsarna 2 och 3 i figuren) har flyttats ut i bredd (Jag antar att det är samma kretsar). Den kondensator som bytts är på 82 pF i original och sitter ansluten till krets 2:s nedersta vänstra ben. Kondensatorn är parallellt ansluten med en resistor till klockkretsen översta högra ben.

Då Lennart prövade vid första försöket, bytte han original-kondensatorn mot en annan kondensator på 45 pF. Allt verkade fungera som vanligt men 40% snabbare: Efter ett tag upptäckte dock Lennart att ROM-funktionerna inte fungerade dvs. $\Sigma+$, $\bar{x}$, P-R, D.MS m.fl. Efter åtskilligt testande har 51 pF visat sig vara gränsen för fungerande ROM-funktioner. För närvarande sitter en kondensator på 56 pF i räknaren, vilket ger en tidsvinst på 15 - 20%. (Dessa kapacitans värden får inte tas alltför allvarligt, av egen erfarenhet med HP-41CV vet jag räknarna är högst individuella med att acceptera en uppsnabbning. Jag har 23.5 pF (inte att jämföra med 58:ans kondensator) i min egen 41:a (original 150 pF), medan många andra räknare inte klarar lägre värden än 50 pF).

På äldre TI-räknare kan denna kondensator hittas genom att följa ledningen på kretskortet från det nedersta vänstra benet på de kretsar som ligger ovanpå varandra. Då hittar man en plattkondensator märkt 82J N470 (så är den åtminstone märkt på den räknare som jag undersökt), för att vara extra säker kan man fortsätta att följa ledningen, då bör man hamna på översta högra benet på klockkretsen (6). Observera att ledningen går en bit på undersidan av kretskortet. För att bli absolut säker på att rätt kondensator hittats kan en annan kondensator på cirka 50 - 100 pF parallellkopplas. Om rätt kondensator hittats bör räknaren gå långsammare. Detta kan göras utan att börja löda i räknaren.



■

Avancerade programmeringsmetoder -4

Lars Nilsson

Det finns många olika sorteringsmetoder, men de flesta kan räknas till någon av följande huvudgrupper:

A. "Insticks sortering"
De nummer som skall sorteras tas ett och ett, och insättes på rätt position relativt tidigare insorterade nummer. Genom att ta ett kort åt gången och sätta det på rätt ställe i handen sorterar de flesta kortspelare sina kort enligt denna metod.

B. "Utbytes sortering"
Numren jämförs två och två. De nummer som vid denna jämförelse visar sig vara felsorterade byter plats. denna procedur fortsätter tills inga nummer behöver byta plats.

C. "Urvals sortering"
Det största numret letas upp och sätts först i listan, det näst största numret letas upp och sätts tvåa i listan o.s.v.

Var och en av dessa metoder och de program som kan samlas därunder, har sina för- och nackdelar. Vissa är snabba för ett litet antal nummer, andra för ett stort antal, metoderna tar olika stort minne i anspråk.et.c. Dessa huvudmetoder kommer här nedan att beskrivas var och en för sig, tillsammans med någon av de viktigare sorterings algoritmer som hör till denna metod.
För enkelhets skull använder vi oss av en och samma lista till alla metoderna.

A. "Insticks sortering"

Lista: 503, 87, 512, 61, 908, 179, 897, 275

```
        503 : 87
          ↑
      87, 503 : 512
              ↑
   87, 503, 512 : 61
 ↑
61, 87, 503, 512 : 908   osv.
                 ↑
```

Det första talet i listan sätts som starttal, därefter jämförs varje tal med de i den sorterade listan (till vänster om kolonet) och insätts då det hittat sin rätta plats (pilarna visar var insättning sker). Denna procedur upprepas tills samtliga tal satts in i den sorterade listan. (Se även Programbiten 81-4 sid. 4-5)

Denna metod används i nedanstående algoritm:
Med R_i avses ett av de register som innehåller listan där (i) är variabeln som innehåller registrets adress. Jämför STO/RCL Ind nn(där nn motsvaras av (i) i detta fall). Listan består av register R_1-R_N. Med i ← j - 1 menas att i får värdet j - 1. Jämför LET i = J - 1 (BASIC).

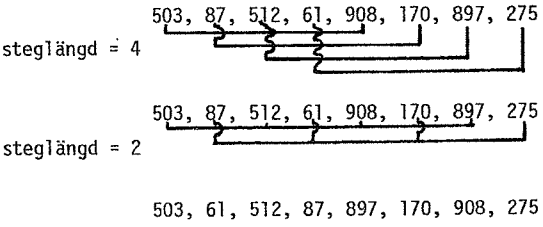
Algoritm A1

1. utför steg 2-5 för j = 2, 3,..., N, avsluta därefter körningen.
2. Sätt i ← j - 1; R_0 ← R_j. R_j kommer nu att jämföras med R_i med minskande värde på i.
3. Jämför R_j med R_i: Om R_j >= R_i gå till steg 5 (Om R_j >= R_i har vi hittat rätt plats för R_j).
4. Flytta R_i och minska i. Sätt R_{i+1} ← R_i. R_i måste flyttas för att det skall finnas plats i listan då R_j hittar sin plats. Sätt i ← i - 1, om i > 0 gå till steg 3
5. Placera R_j i R_{i+1}. Sätt R_{i+1} ← R

Denna metod är mycket snabb då listan är nästan rätt sorterad och inte innehåller alltför många nummer. Dessa egenskaper gör att den ofta förekommer som avslutning i mer avancerade algoritmer, där listan först grovsorteras efter avancerade system och till sist bara behöver finjusteras. Att algoritm A1 är långsam då talen är många och mycket osorterade, beror på att steg 3 och 4 får genomföras ett stortantal gånger. Låt oss anta det sämsta fallet, listan är omvänt sorterad, dvs. största talet först (R_i) och minsta sist (R_n). Detta innebär att steg 3 och 4 hela tiden kommer att köras maximalt antal gånger, till i = 0 i steg 4 (R_j minskar för varje insättning och skall alltså vara längst ner i listan = R_1, eftersom sökningen börjar från i = j - 1 kommer sökningen att innefatta R_{j-1} ner till R_1, vilket tar mer tid desto fler sorterade nummer vi får).

För att få en snabbare sorteringsrutin för ett större antal element måste vi försöka ersätta de små stegen med längre. Donald L. Shell kom 1959 på ett sätt att göra detta. Metoden kallade han "Shellsort", det är metoden som används i Math/Utilities modulen till TI 58/59.
Grundtekniken är den samma som i algoritm A1, men steget på (i) är inte längre 1 utan ändras kontinuerligt. Ursprungslistan delas upp i olika dellistor, som sorteras var för sig. Dessa dellistor läggs sedan samman och sorteras enligt algoritm A1. Steglängden minskas från ett startvärde ner till 1. Värdet på steglängden kan väljas på olika sätt, ett vanligt sätt är att börja på 8 och därefter succesivt halvera steglängden

tills den når 1. Vi kan pröva en sådan sortering på vår lista (steglängden 8 kan inte användas, eftersom vi bara har 8 element. Vi halverar 8 och använder 4 som starttal istället).



503, 61, 512, 87, 897, 170, 908, 275

steglängd = 1 = Algoritm A1

Då steglängden är fyra består den totala listan av fyra dellistor, var och en bestående av två tal. Dessa listor sorteras var för sig (som synes sker ingen förändring i vår lista). När steglängden blir två delas ursprungslistan i två delar, vardera dellista upptar fyra tal, vilka sorteras inbördes. När steglängden blir ett får vi en stor lista som sorteras enligt algoritm A1. I vårt fall hade antagligen A1 varit snabbare än Shells metod, men när antalet element ökar visar sig "Shell sort" överlägsen.

Algoritm A2

1. Utför steg 2 för s = t_1, t_2 ... 1, avsluta därefter körningen. (t_n = steglängder, i vårt exempel var t_1 = 4, t_2 = 2, t_3 = 1; dvs $t_{n+1} = t_n / 2$
2. Utför steg 3-6 för j = s + 1 till j = N.
3. Sätt i ← j - s, $R_0 \leftarrow R_j$
4. Är $R_0 >= R_i$ gå till steg 6
5. Flytta R_i. Sätt $R_{i+s} \leftarrow R_i$. Minska i. Sätt i ← i - s är i > 0 gå till steg 4
6. Sätt $R_{i+s} \leftarrow R_0$

Steglängderna kan varieras på flera olika sätt, att succesivt halvera är ett av de enklare men fortfarande effektiva sätten. (Använd stegen 5, 3, 2, 1; dvs. minska med 2 till s = 2, för att se om listan hade sett bättre ut inför den avslutande A1 sorteringen).

B. "Utbytes sortering"

Den andra av de tre grupperna är utbytes sortering (fritt översatt), vilken grundar sig på att jämföra två element i listan och låta och låta dem byta plats om de visar sig ligga i fel ordning inbördes. Den enklaste metoden kallas "Bubbel sortering". Det lustiga namnet kommer av att de större talen kommer efterhand att "bubbla" upp genom listan och inta sin rätta position. Vi tar ett exempel med vår lista:

antal genomgångar	
0	503, 87, 512, 61, 908, 170, 897, 275
1	87, 503, 61, 512, 170, 897, 275, [908]
2	87, 61, 503, 170, 512, 275, [897, 908]
3	87, 61, 170, 503, 275, [512, 897, 908]
4	87, 61, 170, 275, [503, 512, 897, 908]

Vid första genomgången kommer 503 att visa sig vara större än 87 och dessa byter plats. 503 stoppas av 512, som är större och fortsätter tills 908 påträffas. 908 är det största talet i listan, det kommer således att "bubbla" upp och lägga sig överst i listan. Talen inom klammrarna är den säkert sorterade listan, för varje genomgång tillkommer minst ett tal till den.

Algoritm B1

1. Sätt B ← N (N är antalet element i listan, B är den högsta adress dit listan behöver sorteras).
2. Sätt t ← 0. Utför steg 3 för j = 1, 2,...,B - 1. Gå sedan till steg 4.
3. Om $R_j > K_{j+1}$ byt $R_j \leftrightarrow R_{j+1}$ och sätt t = j.
4. Om t = 0 avsluta algoritmen (inga byten har skett den senaste genomgången). I annat fall sätt B ← t och gå till steg 2. (Om inga ändringar gjorts vid högre adresser än j, innebär det att den delen av listan är rätt sorterad. Tid behöver då inte ödslas på att jämföra talen i den delen av listan).

Vi kommer nu till en mycket avancerad metod, som troligen den effektivaste sorteringsmetoden, då listan har ett stort antal element. Metoden kallas allmänt "Quick sort". Metoden består i att dela upp ursprungslistan i två dellistor; den ena med tal större än och den andra med tal mindre än ett givet element, vanligtvis det första i listan. Denna uppdelning görs med hjälp av två pekare, som arbetar från varsin ända av listan och byter alla tal som ligger på fel sida om det givna elementet (det första). Låt oss på nytt ta ett exempel med vår lista:

a = jämförelsetal; i = pekare 1; j = pekare 2

```
a↓  i↓                        j↓
503, 87, 512, 61, 908, 170, 897, 275
a↓      i↓                    j↓
503, 87, 275, 61, 908, 170, 897, 512
a↓             i↓   j↓
503, 87, 275, 61, 170, 908, 897, 512
                  j↓   i↓
[170, 87, 275, 61], 503, [908, 897, 512]
```

Från start sätts i (pekare 1) på andra elementet och j (pekare 2) på det sista. i ökas sedan tills ett element hittas som ska placeras till höger om jäm-

förelsetalet, dvs. är större än 503. Pekaren stannar på 512 som då byts mot det element som j pekar på, i vårt fall 275. j stegas alltså neråt och elementen jämförs med 503, fast tvärtom, dvs. om de är mindre än 503. 275 var alltså det första tal som påträffades av j, som uppfyller det kravet. i fortsätter nu att öka och j att minska tills de korsar varandra, då utbyts R_j med jämförelsetalet (503). Vi har nu fått två listor, en med tal större än 503 och en med mindre. Dessa listor kan nu var och en sorteras med denna metod, tills hela listan är sorterad.

Med ingående matematiska analyser har man kommit fram till hur många osorterade element som behövs för att denna metod ska vara lönsam. Antalet nödvändiga element har visat sig vara 9. Det lönar sig således inte att sortera vår lilla lista med "Quicksort". I fall då antalet element är stort kommer det att bli många dellistor i dellistorna; när alla dessa dellistor har 9 eller färre element används algoritm A1 för att avsluta sorteringen. "Quicksort" är alltså inte en helt äkta "utbytes rutin", utan den använder även en "insticks rutin" som avslutning.

Algoritm B2

1. Om N <= 9 gå till steg 9. Om inte töm stacken och sätt l ← 1, r ← N. (Stacken är ett antal register som ska hålla reda på adresserna till dellistor som väntar på att bli sorterade. Adressen består av l = lägsta adressen och r = = högsta adressen.)
2. [Sortering av listan $R_1, \ldots, R_r$]. Sätt i ← l, j ← r +1, $R_0 \leftarrow R_l$ (R_0 = jämförelsetal).
3. [Jämför R_0 och R_i]. Sätt i ← i + 1. Om $R_i < R_0$ upprepa detta steg.
4. [Jämför R_0 och R_j]. Sätt j ← j - 1. Om $R_0 < R_j$ upprepa detta steg.
5. [Jämför i och j]. Om j <= i byt $R_l \leftrightarrow R_j$ och gå till steg 7.
6. Byt $R_i \leftrightarrow R_j$ och gå till steg 3.
7. Om r - j >= j - l > 9 lagra (j +1, r) i stacken, sätt r ← j - 1 och gå till steg 2. (Adresserna l, r till den största delen av de två dellistorna lagras i stacken för senare sortering, medan den minsta delen blir sorterad). Om j - l > r - j > 9 lagra (l, j -1) i stacken, sätt l ← j + 1 och gå till steg 2. Om r - j > 9 >= j - l sätt l ← j + 1 och gå till steg 2 (den större delen av listan blir sorterad medan den mindre nu är klar för slutsortering i steg 2 (den större delen av listan blir sorterad medan den mindre nu är klar för slutsortering i steg 9). Om j - l > 9 >= r - j sätt r ← j - 1 och gå till steg 2.
8. Om stacken inte är tom, sätt $l \leftarrow l_s$, $r \leftarrow r_s$ och gå till steg 2. (l_s och r_s är adresser till osorterade dellistor, som ligger lagrade i stacken).
9. Sortera hela listan enligt Algoritm A1.

OBS! Stacken kan behöva bestå av mer än ett register, då ibland flera dellistors adresser kan behöva ligga lagrade samtidigt.
Om något fel finns i steg 1 - 8 kan detta komma att bli dolt, eftersom en fullständig sortering sker i steg 9. Den enda indikationen på ett fel blir då en onödigt lång sorteringstid.

C. "Urvals sortering"

Vi har nu kommit till den sista av de tre grundläggande sorteringsmetoderna. Grunderna i "urvals sortering" är följande:

I Leta upp det största elementet och placera det först i listan.

II Lägg det första elementet utanför listan och upprepa steg I. Dvs. det näst största elementet letas upp och sätts som element 2.

III Fortsätt att upprepa steg I tills N element valts ut (N = antal element i listan).

Vi tittar nu på vad som händer med vår lista vid en sådan sortering.

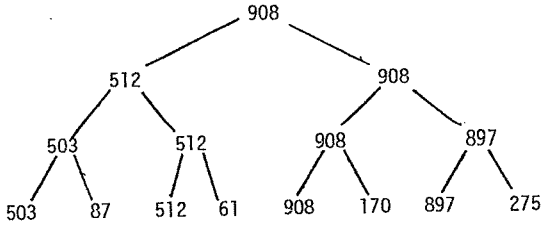
[503, 87, 512, 61, 908, 170, 897, 275]
[503, 87, 512, 61, 170, 897, 275],[908
[503, 87, 512, 61, 170, 275], 897, 908
[503, 87, 61, 170, 275], 512, 897, 908
⋮
[61], 87, 170, 275, 503, 512, 897 908

Den lista som behöver sorteras är den inom klammrar och det största talet är understruket. Från början är 908 störst, det flyttas utanför listan, där nu 897 är störst och flyttas ut osv. Algoritm C1 arbetar direkt på de grunder som har beskrivits.

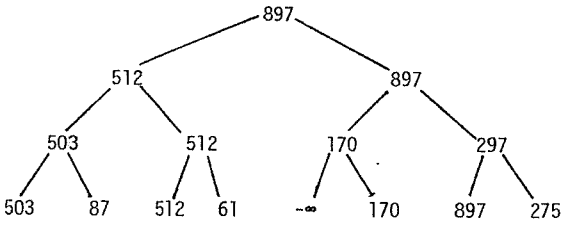
Algoritm C1

1. Utför steg 2 - 5 för j = N, N - 1, N - 2, ... och avsluta därefter körningen.
2. Sätt i ← j - 1, t ← j
3. [Leta upp max]. Om $R_t < R_i$, sätt t ← i
4. Om i > 1, sätt i ← i - 1 och gå till steg 3.
5. Byt $R_t \leftrightarrow R_j$ (R_t är det största elementet i listan och placeras nu utanför listan, nästa lista startar med R_{j-1}).

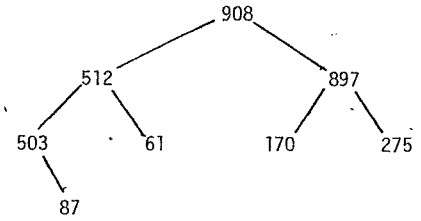
Vi ska även i samband med urvalsmetoden titta på en av de mer avancerade algoritmerna. Den använder sig av principen med binära träd. Vår lista uppsatt som ett binärt träd ser ut såhär:



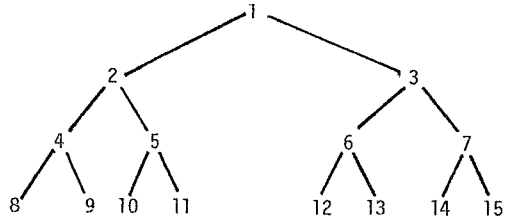
Trädet kan jämföras med en utslagsturnering i någon sport, där det vinnande laget går vidare till en ny match. Här är det istället de olika talen som jämförs och det största går vidare. Det största talet kommer då att ta sig ända till toppen. Om vi nu, i botten av trädet, ersätter det största talet med ett tal som garanterat är mindre än alla andra tal i listan, kommer det näst största talet attta sig till toppen.



908 har ersatts av -∞ som garanterat är mindre än de andra talen i listan. Detta medför att 897, det näst största talet i listan, går upp i toppen.
Vid sortering på detta vis behövs extra minnesutrymme för de extra platser som bildas i trädet. Trädet kan emellertid ändras så att det endast har de 8 platser som listan innehåller.



Om vi nu låter varje tal gå så långt som möjligt, ersätter det största talet med -∞ i botten på trädet utan att ta bort talet i toppen. Vi gör nu om jämförelserna och de två största talen på vardera halvan kommer att gå till sina positioner, dessa ersätts nu med -∞ i botten osv. Detta ger oss ett träd med bara 8 platser. Genom att numrera platserna (vilka hädanefter kommer att kallas noder) kan vi beskriva förhållandet mellan elementen matematiskt.



$R_{(j/2)} >= R_j$ för $1 <= (j/2) < j <= N$, där R_j är element vid nod j. ((j/2) betecknar heltalsdelen av j/2. Exempel: talet vid nod 4 är större än talen vid noderna (2 * 4) och (2 * 4 + 1). (2 * 4 + 1 = 9, 9 som nod sammankopplas också med nod 4 enligt (j/2), (9/2) = 4 (9/2 = 4.5, INT 4.5 = 4).

Om vi nu kan få vår lista att uppfylla dessa krav, kan vi plocka ut talen ett och ett i ordning från toppen av trädet. När listan ska ordnas på detta vis kan vi börja med den övre halvan av listan, eftersom inga tal ska uppfylla några villkor gentemot varandra ((j/2) ska uppfylla ett villkor gentemot j, alltså har (j/2) + 1 till j inga villkor att uppfylla inbördes, om j sätts till max och stegas nedåt dvs. j = N från starten). Sedan plockas talen från andra halvan in ett och ett, och listan justeras så att villkoren uppfylls. När hela listan uppfyller villkoren flyttas R_1 till högsta platsen i listan (R_N) (R_1 = nod 1, som är listans största tal). Listan som nu består av elementen R_1 till R_{N-1} justeras åter så att villkoren uppfylls (När R_1 = max försvann ändrades listan så att villkoren inte längre stämmer överallt, därför måste listan justeras på nytt). Efter justering kommer R_1 att innehålla det näst största talet, som flyttas till R_{N-1} osv.
Vi ser åter på hur vår lista ändras vid sortering.

R_0	R_1	R_2	R_3	R_4	R_5	R_6	R_7	R_8
	503	87	512	61	[908	170	897	275]
61	503	87	512	[61	908	170	897	275]
512	503	87	[512	275	908	170	897	61]
87	503	[87	897	275	908	170	512	61]
503	[503	908	897	275	87	170	512	61]
61	[908	503	897	275	87	170	512]	908
61	[897	503	512	275	87	170]	897	908
61	[512	503	170	275	87]	512	897	908
87	[503	275	170	61]	503	512	897	908
87	[275	61	170]	275	503	512	897	908
87	[170	61]	170	275	503	512	897	908
	[61	87]	170	275	503	512	897	908

Den del av listan som står inom klammrar ska uppfylla villkoren. Ett tal åt gången plockas in, först 61 som har villkor att uppfylla gentemot 275 ($R_{8/2} >= R_8$). R_4 och R_8 byts alltså. R_3 plockas nu in i listan. R_3 ska vara större än R_6 och R_7, R_7 är dock större än R_3, dessa byter då plats och villkoren är uppfyllda. Då hela listan uppfyller villkoren plockas R_1 ut och placeras i R_8, elementet som fanns i R_8 lagras i R_0 men jämförs i listan som om det ersatte 908, vilket medför att det flyttas mot slutet av listan. Denna procedur upprepas tills alla elementen plockats ut. De tal som i figuren här ovan är understrukna, är de mellan vilka villkoren är ställda. Om det inplockade elementet trillar neråt i listan har detta nya tal att uppfylla villkor gentemot. Dessa är inte understrukna.

Detta är en mycket avancerad metod som det kan vara svårt att få kläm på direkt. De många justeringarna verkar också vara besvärliga, men som synes i den algoritmen är justeringsdelen förbluffande kurt och snabb.

KOMPLEXA HYPERBOLISKA FUNKTIONER KOMPLEX A HYPERBOLISKA FUNKTIONER KOMPLEXA HYPER BOLISKA FUNKTIONER KOMP[illegible]ERBOLISKA FUNKTIONER KOMPLE[illegible]SKA FUNKTIO NER KOMPLE[illegible]KA FUNKTIONER KOM PLE[illegible]A FUNKTIONER KOMPLEXA H YPER[illegible]A FUNKTIONER KOMPLEXA HYPERBOL ISKA FUNKTIONER KOMPLEXA HYPERBOLISKA FU NKTIONER KOMPLEXA HYPERBOLISKA FUNKTIONE

Det här är ett program som kan beräkna komplexa hyperboliska funktioner, samt deras inverser. Programmet använder formlerna:

$\sinh x = \sinh a \cos b \pm i \cosh a \sin b$

$\cosh x = \cosh a \cos b \pm i \sinh a \sin b$

$\tanh x = \sinh x / \cosh x$

$\sinh^{-1} x = \ln (x + \sqrt{x^2 + 1})$

$\cosh^{-1} x = \ln (x + \sqrt{x^2 - 1})$

$\tanh^{-1} x = 1/2 \ln (x + 1 / 1 - x)$

Genom flitigt användande av Standardmodulens program 6 har programmet blivit kompakt, vilket säkerligen uppskattas av TI 58 användare.

INSTRUKTIONER

Mata in den reella delen av x (= a + bi)	a	A
Mata in den imaginära delen av x	b	A
Beräkna nu någon av följande funktioner:		
sinh x		B
$\sinh^{-1}$		B'
cosh x		C
$\cosh^{-1}$ x		C'
tanh x		D
$\tanh^{-1}$ x		D'

I samtliga fall hämtas den reela delen ur displayen, den imaginära delen återfinns i t-registret.
Det framräknade talet blir automatiskt nya x.

```
000  76 LBL    031  12  B     061  76 LBL    091  19 D'
001  11  A     032  16 A'     062  17 B'     092  10 E'
002  69 OP     033  43 RCL    063  10 E'     093  69 OP
003  04  04    034  02  02    064  36 PGM    094  21  21
004  11  A     035  32 X:T    065  04  04    095  01  1
005  92 RTN    036  43 RCL    066  13  C     096  94 +/-
006  76 LBL    037  01  01    067  69 OP     097  49 PRD
007  16 A'     038  92 RTN    068  21  21    098  03  03
008  43 RCL    039  76 LBL    069  36 PGM    099  49 PRD
009  01  01    040  13  C     070  05  05    100  04  04
010  48 EXC    041  16 A'     071  14  D     101  69 OP
011  02  02    042  36 PGM    072  36 PGM    102  23  23
012  42 STO    043  06  06    073  04  04    103  36 PGM
013  01  01    044  13  C     074  12  B     104  04  04
014  92 RTN    045  01  1     075  36 PGM    105  18 C'
015  76 LBL    046  94 +/-    076  05  05    106  36 PGM
016  10 E'     047  49 PRD    077  16 A'     107  05  05
017  43 RCL    048  02  02    078  92 RTN    108  16 A'
018  01  01    049  61 GTO    079  76 LBL    109  93  .
019  42 STO    050  00  00    080  18 C'     110  05  5
020  03  03    051  33  33    081  10 E'     111  49 PRD
021  43 RCL    052  76 LBL    082  36 PGM    112  01  01
022  02  02    053  14  D     083  04  04    113  49 PRD
023  42 STO    054  16 A'     084  13  C     114  02  02
024  04  04    055  36 PGM    085  69 OP     115  43 RCL
025  92 RTN    056  06  06    086  31  31    116  02  02
026  76 LBL    057  14  D     087  61 GTO    117  32 X:T
027  12  B     058  61 GTO    088  00  00    118  43 RCL
028  16 A'     059  00  00    089  69  69    119  01  01
029  36 PGM    060  32  32    090  76 LBL    120  92 RTN
030  06  06
```

■

Robert Prins har kommit på ett sätt att lista spärrade moduler. Tyvärr är det ett arbete som tar lång tid, mellan 30 och 40 timmar. Robert är beredd att göra detta om han får tillräckligt många intresserade. Skriv till Robert under adress: Alfred Nobellaan 112, 3731 DX De Bilt, Holland. ■

Algoritm C2

1. Sätt l ← (N/2) + 1, r ← N (l är lägsta och r högsta adressen för den del av listan som uppfyller villkoren, jämför klammrarna.
2. Om l > 1 sätt l ← l - 1, $R_0 \leftarrow R_l$ i annat fall sätt $R_0 \leftarrow R_r$, $R_r \leftarrow R_1$ och r ← r -1 om detta gör r = 1 sätt $R_1 \leftarrow R_0$ och avbryt körningen.
3. Sätt j ← 1
4. Sätt i ← j, j ← 2 · j. Om j > r gå till steg 8 om j = r gå till steg 6. (Vi har nu i = (j/2)).
5. Om $R_j < R_{j+1}$ sätt j ← j +1 (här tas det största av de två tal som enligt villkoret ska vara mindre än R_0 ut).
6. Om $R_0 >= R_j$ gå till steg 8.
7. Sätt $R_i \leftarrow R_j$ och gå till steg 4 (villkoret som inte blev uppfyllt i steg 6 blir här tillrättat, genom att lägga det största talet överst. Detta medför att en ny plats måste hittas åt R_0 som inte med säkerhet kan inta $R_{j:s}$ plats).
8. Sätt $R_i \leftarrow R_0$ och gå till steg 2 (villkoren är uppfyllda och R_0 placeras på rätt plats).

Det finns ett stort antal sorteringsmetoder kvar att behandla, de som har tagits upp här är bara de grundläggande samt en av de effektivare inom varje grupp. De algoritmer som behandlas är alla av den typ som sorterar inom listans eget minnesutrymme, skälet till detta är det lilla minne som TI 58/59 har. För den som vill läsa mer om sortering rekommenderar jag varmt "The art of computer programming" volym 3, av D. E. Knuth. Den innehåller nära 400 sidor om sortering, varur jag själv har hämtat både kunskaper och algoritmer. ■

Här nedan följer program som bygger på några av de sorteringsalgoritmer som presenteras på sidorna 16 - 20. Shell sort är stulet direkt från matematik-modulen, de övriga programmen är skrivna av Björn Gustavsson.

KÖRINSTRUKTIONER

För Bubble sort och Straight insertion gäller att talen placeras i register 01 och uppåt. För att starta sorteringen matar du först in hur många tal som ska sorteras, följt av A. Vad gäller Straight insertion programmet kan B-rutin uteslutas, dess uppgift är att fylla register med slumptal, förutsatt att ett frö finns i displayen och registrens startadress finns i R00.

Shell sort

000	76	LBL	039	03	3	078	00	00	105	22	INV
001	15	E	040	22	INV	079	82	HIR	106	77	GE
002	00	0	041	77	GE	080	14	14	107	00	00
003	42	STO	042	00	00	081	42	STO	108	31	31
004	00	00	043	53	53	082	00	00	109	32	X:T
005	92	RTN	044	04	4	083	32	X:T	110	61	GTO
006	76	LBL	045	22	INV	084	72	ST*	111	00	00
007	11	A	046	77	GE	085	00	00	112	60	60
008	69	DP	047	00	00	086	82	HIR	113	00	0
009	20	20	048	53	53	087	17	17	114	92	RTN
010	72	ST*	049	01	1	088	82	HIR	115	76	LBL
011	00	00	050	67	EQ	089	54	54	116	13	C
012	92	RTN	051	01	01	090	01	1	117	01	1
013	61	GTO	052	13	13	091	32	X:T	118	22	INV
014	11	A	053	82	HIR	092	82	HIR	119	90	LST
015	76	LBL	054	07	07	093	14	14	120	42	STO
016	12	B	055	82	HIR	094	77	GE	121	00	00
017	43	RCL	056	55	55	095	00	00	122	76	LBL
018	00	00	057	01	1	096	62	62	123	14	D
019	53	(	058	82	HIR	097	01	1	124	73	RC*
020	82	HIR	059	06	06	098	82	HIR	125	00	00
021	08	08	060	82	HIR	099	36	36	126	69	DP
022	82	HIR	061	04	04	100	82	HIR	127	20	20
023	05	05	062	42	STO	101	16	16	128	92	RTN
024	55	÷	063	00	00	102	32	X:T	129	61	GTO
025	02	2	064	73	RC*	103	82	HIR	130	01	01
026	54	)	065	00	00	104	15	15	131	24	24
027	59	INT	066	32	X:T
028	61	GTO	067	82	HIR
029	00	00	068	17	17
030	53	53	069	44	SUM
031	82	HIR	070	00	00
032	18	18	071	73	RC*
033	82	HIR	072	00	00
034	05	05	073	77	GE
035	82	HIR	074	00	00
036	17	17	075	97	97
037	32	X:T	076	32	X:T
038	01	1	077	72	ST*

CHECKSUM:

BANK 1
4289409702.

PROG
4289409702.

Shell sort har samma instruktioner som ovanstående program, med vissa tillägg. Rutin C listar register. Rutin D tar fram ett register i sänder, och visar det i displayen. A underlättar inmatning av talen. E står för initiering; samtliga dessa rutiner kan uteslutas. Sorteringen startas i detta fall genom ett tryck på B, om antal tal finns i R00.

Quick sort slutligen, använder själv register 00 - 12. Talen får därför placeras ovanför R12. För att starta sorteringen: mata in i vilket register det sista talet finns, tryck A.

Straight insertion

000	76	LBL	022	32	X:T	044	22	INV	053	00	0
001	11	A	023	72	ST*	045	28	LOG	054	95	=
002	82	HIR	024	00	00	046	33	X²	055	59	INT
003	04	04	025	97	DSZ	047	22	INV	056	72	ST*
004	01	1	026	00	00	048	59	INT	057	00	00
005	42	STO	027	00	00	049	65	×	058	69	DP
006	00	00	028	07	07	050	32	X:T	059	20	20
007	73	RC*	029	82	HIR	051	01	1	060	32	X:T
008	00	00	030	14	14	052	00	0	061	61	GTO
009	69	DP	031	32	X:T				062	12	B
010	20	20	032	01	1
011	32	X:T	033	85	+
012	73	RC*	034	42	STO
013	00	00	035	00	00
014	77	GE	036	22	INV
015	00	00	037	77	GE
016	29	29	038	00	00
017	32	X:T	039	07	07
018	72	ST*	040	25	CLR
019	00	00	041	92	RTN
020	69	DP	042	76	LBL
021	30	30	043	12	B

CHECKSUM:

BANK 1
2966452502.

PROG
2966452502.

Bubble sort

000	76	LBL	019	34	34	038	14	14	044	82	HIR
001	11	A	020	32	X:T	039	22	INV	045	13	13
002	82	HIR	021	72	ST*	040	67	EQ	046	22	INV
003	04	04	022	00	00	041	00	00	047	67	EQ
004	00	0	023	69	DP	042	10	10	048	11	A
005	82	HIR	024	30	30	043	29	CP	049	92	RTN
006	03	03	025	32	X:T
007	01	1	026	72	ST*
008	42	STO	027	00	00
009	00	00	028	69	DP
010	73	RC*	029	20	20
011	00	00	030	43	RCL
012	69	DP	031	00	00
013	20	20	032	82	HIR
014	32	X:T	033	03	03
015	73	RC*	034	43	RCL
016	00	00	035	00	00
017	77	GE	036	32	X:T
018	00	00	037	82	HIR

CHECKSUM:

BANK 1
5906849000.

PROG
5906849000.

Quick sort

000	76	LBL	029	75	-	058	22	INV	087	75	-	116	29	29	145	63	63	174	00	00	189	68	68
001	11	A	030	43	RCL	059	77	GE	088	43	RCL	117	43	RCL	146	69	DP	175	77	GE	190	82	HIR
002	82	HIR	031	10	10	060	00	00	089	09	09	118	09	09	147	30	30	176	01	01	191	14	14
003	04	04	032	42	STO	061	52	52	090	95	=	119	48	EXC	148	75	-	177	90	90	192	32	X:T
004	42	STO	033	08	08	062	43	RCL	091	32	X:T	120	10	10	149	59	INT	178	32	X:T	193	01	1
005	11	11	034	95	=	063	09	09	092	43	RCL	121	85	+	150	42	STO	179	72	ST*	194	85	+
006	42	STO	035	32	X:T	064	32	X:T	093	09	09	122	69	DP	151	10	10	180	00	00	195	42	STO
007	09	09	036	09	9	065	43	RCL	094	75	-	123	39	39	152	95	=	181	69	DP	196	00	00
008	69	DP	037	77	GE	066	08	08	095	43	RCL	124	69	DP	153	65	×	182	30	30	197	22	INV
009	29	29	038	01	01	067	77	GE	096	10	10	125	39	39	154	01	1	183	32	X:T	198	77	GE
010	01	1	039	40	40	068	00	00	097	95	=	126	43	RCL	155	00	0	184	72	ST*	199	01	01
011	03	3	040	73	RC*	069	79	79	098	77	GE	127	09	09	156	00	0	185	00	00	200	68	68
012	42	STO	041	10	10	070	73	RC*	099	01	01	128	55	÷	157	95	=	186	97	DSZ	201	25	CLR
013	10	10	042	32	X:T	071	08	08	100	15	15	129	01	1	158	42	STO	187	00	00	202	92	RTN
014	25	CLR	043	69	DP	072	63	EX*	101	43	RCL	130	00	0	159	11	11	188	01	01
015	42	STO	044	28	28	073	09	09	102	09	09	131	00	0	160	61	GTO
016	00	00	045	73	RC*	074	72	ST*	103	85	+	132	95	=	161	00	00
017	23	LNX	046	08	08	075	08	08	104	01	1	133	69	DP	162	23	23
018	42	STO	047	22	INV	076	61	GTO	105	85	+	134	20	20	163	01	1
019	12	12	048	77	GE	077	00	00	106	69	DP	135	72	ST*	164	03	3
020	94	+/-	049	00	00	078	40	40	107	39	39	136	00	00	165	85	+
021	72	ST*	050	43	43	079	73	RC*	108	43	RCL	137	61	GTO	166	42	STO
022	09	09	051	32	X:T	080	09	09	109	09	09	138	00	00	167	00	00
023	43	RCL	052	32	X:T	081	63	EX*	110	48	EXC	139	23	23	168	73	RC*
024	11	11	053	69	DP	082	10	10	111	11	11	140	73	RC*	169	00	00
025	42	STO	054	39	39	083	72	ST*	112	61	GTO	141	00	00	170	69	DP
026	09	09	055	73	RC*	084	09	09	113	01	01	142	29	CP	171	20	20
027	69	DP	056	09	09	085	43	RCL	114	28	28	143	67	EQ	172	32	X:T
028	29	29	057	32	X:T	086	11	11	115	69	DP	144	01	01	173	73	RC*

CHECKSUM:

BANK 1
7074342004.

PROG
7074342004.

Benchmarks -2

BO NORDLIN

I förra numrets Benchmark artikel lovade jag att återkomma med CC-40:s tider:

1	2	3	4	5	6	7	8
5.2	15.2	36.7	37.1	41.4	72.7	116.4	530.1

Vilket ger ett ovägt medelvärde på 106.85. Därmed skulle CC-40 placera sig under TI-99 men över Casio PB-100 i vår tidigare publicerade lista. HP-75 och Epson HX-20, som också är batteridrivna, portabla BASIC datorer, om än i högre prisklass, är alltså snabbare än CC-40.
Om man analyserar värdena vidare finner man att CC-40 är relativt snabb på Benchmarks 1 - 7. Men att den sackar betänkligt när logaritmiska och trigonometriska funktioner blandas in. Sammantaget är den inte radikalt snabbare än sina närmaste konkurrenter.

Från Lars Winter och Lars Nilsson har jag fått en hel del berättigad kritik för mina HP-41 program. Detta är inte annat än naturligt då jag inte behärskar HP:n i samma utsträckning som TI 58/59. Jag återger här nedan Lars Nilssons program med tider och hans kommentarer.

```
BM 1          96 s
1 E 3
LBL 00
DSE X
GTO 00
END
```

Slingkörning i stacken är snabbare än med minnesregister. Jag tog med rutinen bara som en kommentar då tidsskillnaden är så liten.

```
BM 2          98 s
1 E 3
ENTER
  1
ACOS
LBL 00
LAST X
  +
X <Y?
GTO 00
END
```

Ett typiskt exempel på en rutin där ISG skulle ha använts i ett praktiskt HP-program: Förvånansvärt nog är denna rutin nästan lika snabb som en ISG-rutin för samma ändamål. 1 ACOS lägger en etta i Last X registret och nollställer X-registret.

```
BM 3          360 s
1 E 3         LAST X
ENTER           +
CLX           LAST X
LBL 00          -
ISG X         STO 01
CLA            X<>L
ENTER          X <Y?
  /           GTO 00
LAST X        END
  .
```

Räkning i stacken går snabbare för ISG, som här fungerar som Dsz nn Nop på TI 58/59, ISG räknar dock upp och inte ned som Dsz. CLA efter ISG fungerar som en enbytes Nop.

```
BM 4          570 s
1 E 3           .
ENTER           4
CLX             +
LBL 00          5
ISG X           -
CLA           STO 06
ENTER         RDN
ENTER         X<Y?
  2           GTO 00
  /           END
  3
```

Även i denna rutin bevaras testvärdet i stacken, i vilken även uppräckning sker. Den största orsaken till att rutinerna snabbats upp är att inmatningen av testvärdet inte behöver göras varje varv. Utan det ligger konstant i stacken.

```
BM 5          630 s
1 E 3         CLA           3             -           GTO 00
ENTER         ENTER         .           STO 06        STOP
CLX           ENTER         4           XEQ 01        LBL 01
LBL 00          2           +           RDN           RTN
ISG X           /           5           X<Y?          END
```

```
BM 6          1010 s
1 E 3           2         STO 06    DSE-loopen som väl
ENTER           /         LBL 02    enligt BASICrutinen
CLX             3         DSE L     skulle ha varit en
LBL 00          .         GTO 02    ISG-loop, kan smidigt
ISG X           4         RDN       göras då 5 redan lig-
CLA             +         X<Y?      ger i Last X och ned-
ENTER           5         GTO 00    räkningen kan göras
ENTER           -         END       direkt däri.
```

```
BM 7          1230 s
1 E 3         ENTER         4           LBL 02        GTO 00
ENTER         ENTER         +         STO IND L       STOP
CLX             2           5         DSE L         LBL 01
LBL 00          /           -         GTO 02          RTN
ISG X           3         STO 06      RDN             END
CLA             .         XEQ 01      X<Y?
```

```
BM 8          1050 s    STO 02      X<Y?
1 E 3         CLA       LAST X      GTO 00
ENTER         X↑2       SIN         END
CLX           STO 01    STO 03
LBL 00        LAST X    R↑
ISG X         LOG       LAST X
```

6▶ UTMANINGEN!

Gemensam början på A- och B-programmen.

000	25	CLR	040	55	÷	080	09	9	120	65	×				
001	01	1	041	82	HIR	081	42	STO	121	82	HIR				
002	00	0	042	18	18	082	00	00	122	16	16				
003	69	DP	043	54	)	083	25	CLR	123	95	=				
004	17	17	044	74	SM*	084	85	+	124	65	×				
005	47	CMS	045	00	00	085	73	RC*	125	82	HIR				
006	01	1	046	65	×	086	00	00	126	18	18				
007	32	X:T	047	82	HIR	087	59	INT	127	22	INV				
008	09	9	048	17	17	088	75	-	128	64	PD*				
009	09	9	049	65	×	089	53	(	129	00	00				
010	42	STO	050	82	HIR	090	24	CE	130	82	HIR				
011	00	00	051	18	18	091	55	÷	131	15	15				
012	82	HIR	052	95	=	092	82	HIR	132	74	SM*				
013	18	18	053	65	×	093	16	16	133	00	00				
014	75	-	054	82	HIR	094	54	)	134	82	HIR				
015	53	(	055	18	18	095	59	INT	135	18	18				
016	24	CE	056	97	DSZ	096	82	HIR	136	97	DSZ				
017	55	÷	057	00	00	097	05	05	137	00	00				
018	82	HIR	058	00	00	098	65	×	138	00	00				
019	17	17	059	14	14	099	82	HIR	139	84	84				
020	54	)	060	25	CLR	100	16	16	140	25	CLR				
021	59	INT	061	82	HIR	101	85	+	141	22	INV				
022	74	SM*	062	17	17	102	73	RC*	142	87	IFF				
023	00	00	063	50	I×I	103	00	00	143	01	01				
024	65	×	064	67	EQ	104	22	INV	144	00	00				
025	82	HIR	065	01	01	105	59	INT	145	08	08				
026	17	17	066	47	47	106	95	=	146	91	R/S				
027	95	=	067	75	-	107	65	×	147	82	HIR				
028	65	×	068	02	2	108	82	HIR	148	16	16				
029	82	HIR	069	95	=	109	18	18	149	34	FX				
030	18	18	070	65	×	110	75	-	150	82	HIR				
031	75	-	071	82	HIR	111	53	(	151	06	06				
032	53	(	072	17	17	112	24	CE	152	86	STF				
033	53	(	073	69	DP	113	55	÷	153	01	01				
034	24	CE	074	10	10	114	82	HIR	154	61	GTO				
035	55	÷	075	94	+/-	115	16	16	155	00	00				
036	82	HIR	076	95	=	116	54	)	156	79	79				
037	17	17	077	82	HIR	117	59	INT							
038	54	)	078	07	07	118	72	ST*							
039	59	INT	079	09	9	119	00	00							

Slut A-programmet. Slut B-programmet. C-programmet.

202	76	LBL	212	82	HIR	222	08	08	232	95	=				
203	11	A	213	07	07	223	68	NOP	233	22	INV				
204	25	CLR	214	02	2	224	25	CLR	234	52	EE				
205	09	9	215	05	5	225	02	2	235	58	FIX				
206	69	DP	216	82	HIR	226	52	EE	236	00	00				
207	17	17	217	06	06	227	01	1	237	60	DEG				
208	01	1	218	01	1	228	02	2	238	86	STF				
209	07	7	219	52	EE	229	94	+/-	239	40	IND				
210	00	0	220	06	6	230	85	+							
211	05	5	221	82	HIR	231	01	1							
202	76	LBL	212	07	07	222	82	HIR	232	95	=				
203	12	B	213	02	2	223	08	08	233	22	INV				
204	25	CLR	214	03	3	224	25	CLR	234	52	EE				
205	09	9	215	09	9	225	02	2	235	58	FIX				
206	69	DP	216	33	X²	226	52	EE	236	00	00				
207	17	17	217	82	HIR	227	01	1	237	60	DEG				
208	05	5	218	06	06	228	02	2	238	86	STF				
209	00	0	219	01	1	229	94	+/-	239	40	IND				
210	01	1	220	52	EE	230	85	+							
211	82	HIR	221	06	6	231	01	1							
000	76	LBL	051	01	1	102	72	ST*	153	00	00				
001	10	E'	052	00	0	103	01	01	154	75	75				
002	25	CLR	053	42	STO	104	43	RCL	155	43	RCL				
003	09	9	054	07	07	105	08	08	156	07	07				
004	69	DP	055	03	3	106	59	INT	157	42	STO				
005	17	17	056	00	0	107	65	×	158	00	00				
006	03	3	057	42	STO	108	01	1	159	85	+				
007	94	+/-	058	06	06	109	06	6	160	43	RCL				
008	91	R/S	059	76	LBL	110	75	-	161	06	06				
009	02	2	060	19	D'	111	73	RC*	162	75	-				
010	94	+/-	061	43	RCL	112	02	02	163	01	1				
011	91	R/S	062	07	07	113	59	INT	164	95	=				
012	92	RTN	063	42	STO	114	65	×	165	42	STO				
013	76	LBL	064	00	00	115	04	4	166	01	01				
014	11	A	065	43	RCL	116	85	+	167	73	RC*				
015	10	E'	066	06	06	117	82	HIR	168	01	01				
016	29	CP	067	42	STO	118	17	17	169	59	INT				
017	01	1	068	01	01	119	95	=	170	99	PRT				
018	52	EE	069	85	+	120	42	STO	171	73	RC*				
019	06	6	070	03	3	121	08	08	172	01	01				
020	82	HIR	071	00	0	122	22	INV	173	22	INV				
021	08	08	072	95	=	123	77	GE	174	59	INT				
022	25	CLR	073	42	STO	124	01	01	175	65	×				
023	82	HIR	074	02	02	125	89	89	176	82	HIR				
024	07	07	075	73	RC*	126	25	CLR	177	18	18				
025	02	2	076	01	01	127	82	HIR	178	95	=				
026	09	9	077	42	STO	128	07	07	179	99	PRT				
027	42	STO	078	08	08	129	43	RCL	180	69	DP				
028	07	07	079	22	INV	130	08	08	181	31	31				
029	03	3	080	59	INT	131	75	-	182	97	DSZ				
030	01	1	081	65	×	132	53	(	183	00	00				
031	42	STO	082	01	1	133	24	CE	184	01	01				
032	06	06	083	06	6	134	55	÷	185	67	67				
033	61	GTO	084	75	-	135	82	HIR	186	25	CLR				
034	19	D'	085	73	RC*	136	18	18	187	35	1/X				
035	76	LBL	086	02	02	137	54	)	188	91	R/S				
036	12	B	087	22	INV	138	59	INT	189	02	2				
037	76	LBL	088	59	INT	139	82	HIR	190	00	0				
038	13	C	089	65	×	140	37	37	191	94	+/-				
039	10	E'	090	04	4	141	65	×	192	82	HIR				
040	03	3	091	85	+	142	82	HIR	193	07	07				
041	00	0	092	82	HIR	143	18	18	194	65	×				
042	42	STO	093	17	17	144	95	=	195	82	HIR				
043	07	07	094	55	÷	145	74	SM*	196	18	18				
044	42	STO	095	82	HIR	146	01	01	197	95	=				
045	06	06	096	18	18	147	69	DP	198	22	INV				
046	61	GTO	097	75	-	148	21	21	199	44	SUM				
047	19	D'	098	59	INT	149	69	DP	200	08	08				
048	76	LBL	099	82	HIR	150	22	22	201	61	GTO				
049	14	D	100	07	07	151	97	DSZ	202	01	01				
050	10	E'	101	95	=	152	00	00	203	29	29				

CHECKSUM: Hela A: 5835204702; hela B: 2734048302; C: 2350783506.

Början på resultatet:

```
141592.
653589.
793238.
462643.
383279.
502884.
197169.
399375.
105820.
974944.
592307.
816406.
286208.
998628.
 34825.
342117.
 67982.
148086.
513282.
306647.
 93844.
609550.
582231.
725359.
408128.
481117.
450284.
102701.
938521.
105559.
```

■

MEDLEMS ANNONSER

SÄLJES

Texas TI 59 med skrivare PC 100, Leisure modulen och ett stort antal program säljes för 2400:-.
TI 58 med statistikmodulen 595:-, obs! så gott som oanvänd.
Elektronik-, matematik- och spelmodulen pris 160:-/st.
Klas Mangerud, Rågvägen 13, 175 43 JÄRFÄLLA. Tel. 0758-393 08.

SÄLJES

Matematikmodul 200:-. Ring Lars Nilsson tel. 0418-172 51 efter kl. 16.

▶ Detta ger ett ovägt medelvärde på 630.5, visar att 41:an är något snabbare än 58/9 i Fast Mode (på detta test, är säkrast att tillägga). Lars Nilsson påpekar att det är möjligt att höja HP:ns hastighet till det dubbla, genom ett ganska enkelt ingrepp i räknaren. Nog om HP!

Även från Gösta Blume har jag fått kommentarer om Benchmark artikeln. Han påpekar att jag har använt mig av x^2, där BASIC datorerna tvingas använda funktionen upphöjt till 2, vilket är långsammare. Dessutom gjorde jag en dundertabbe! I BASIC programmen står det log, vilket enligt MicroSoft standard ska uttydas naturlig logaritm. Detta hade Gösta påpekat för mig innan jag hade gjort mina Benchmarks, trots detta använde jag mig av 10-logaritm i mina program. Detta fel har tyvärr upprepats i Lars Nilssons HP-program. Effekten av detta torde inte ha blivit stor, det rör sig 2 - 3 sekunder på det ovägda genomsnittet.
Gösta har kört Benchmarks-testet både på sin 58:a och 59:a. Resultaten skiljer faktiskt avsevärt från varandra: 15%!

Till sist måste jag erkänna att en hel del "tryckfel" smög sig in i min föregående text. I BASIC-programmen 1 - 7 saknas ett "<" mellan K och 1000. Dessutom ska rad 140 i BM 8 uttydas A=K↑2. ■

Nu är den här, kompaktdatorn från Texas Instruments!

CC-40

För nybörjare, specialister, studerande, forskare, affärsmän...
CC-40 är den första modellen i en ny familj av kompaktdatorer från Texas Instruments. Programmeras i en utökad version av BASIC. Färdiga program finns i "Solid State Software"-moduler och i små kassettband, "Wafertape". Hela systemet är batteridrivet och ryms ledigt i en portfölj. Upp till 200 timmars drift på 4 st alkaliska batterier. 6 K RAM-minne och 34 K inbyggt ROM-minne i grundutförandet.

TEXAS INSTRUMENTS

Skrivare/Plotter HX-1000
X–Y grafik i 4 färger. För tex cirkeldiagram, histogram och regressionskurvor. Skriver ut programlistor och resultat. Upp till 36 tecken per rad. Skriver på 57 mm vanligt papper.

Wafertape HX-2000
Digital mikrokassettenhet som ger 48 K lagringsutrymme i varje kassett. Snabb tillgång till program och data (8000 baud). Styrs helt från datorn.

RS-232 parallell-interface HX-3000
För anslutning av CC-40 till matrisskrivare, modem eller annan dator.

Programvara
18 programpaket finns redan klara. 4 "Solid State Software"-moduler (upp till 128 K) och 14 Wafertapekassetter. Ledtexter på engelska, tyska eller franska.