Pell's ligning

Christian Marcussen

I 1940'erne lykkedes det at konstruere de første effektive elektroniske regnemaskiner. I det 10-år, der fulgte, 1950'erne, blev forskerne klar over, at det var muligt at kaste lys over mange problemer, naturvidenskabelige og matematiske, ved at lade datamaterne arbejde numerisk med disse problemer. Det viste sig i 1950'erne, at det var muligt at tage opgaver op, som man før havde måttet lade ligge, da de krævede for omfattende numeriske regninger. I 1950'erne blev man også klar over, at de forskellige områder af naturvidenskab og matematik krævede numeriske metoder af varierende art, når de skulle behandles effektivt af datamaterne. De hele tals teori har op gennem tiderne fascineret matematisk interesserede mennesker, både professionelle og amatører. De hele tals system har en meget simpel opbygning, som de fleste mennesker har let ved at forstå.

Da matematikerne i 1600-tallet forsøgte at trænge dybere ind i de hele tals natur viste det sig hurtigt, at det var svært at afdække tallenes hemmeligheder. Man stødte på store teoretiske vanskeligheder; og mange forhold vedrørende de hele tals natur er stadig uafklarede. Disse problemer ser undertiden simple ud ved en overfladisk betragtning; og meningen med dem kan forstås af mennesker med kun ringe matematisk viden. Det er derfor forståeligt, at mange er kommet til at interessere sig for de hele tals problemer.

Vi vil i dette indlæg beskæftige os med nogle simple metoder til numerisk behandling af problemer i de hele tals teori. Ved de almindelige kommercielle og naturvidenskabelige beregninger bruges floating-point talrepræsentation i datamaterne. De hele tal kræver fixed-point talrepræsentation; derfor kan datamaterne i almindelighed kun regne med hele tal af en meget begrænset størrelse. For en mikrocomputer ligger denne grænse i mange tilfælde ved tallet $2^{15}-1=32767$. Indgår der i en regning et helt tal, som overstiger denne grænse, kommer der den fejludskrift, som man betegner "overflow"; og maskinen standser udførelsen af vores regneprogram.

Arbejder man numerisk med talteoretiske problemer, finder man hurtigt ud af, at man ikke kan affinde sig med denne øvre grænse for størrelsen af hele tal. Man forsøger at omgå dette problem ved at anvende metoder af egen opfindelse ved programmeringen af de talteoretiske problemer. Anvendelsen af disse metoder gør, at man kan regne numerisk med væsentlig større hele tal end det ellers er muligt med vores computer.

Vi vil i dette indlæg komme ind på rutiner, som kan udføre de almindelige regningsarter med "vilkårligt store" hele tal. "Vilkårligt store" hele tal skal her forstås på den måde, at der er en naturlig grænse for tallenes størrelse. Denne grænse afhænger af vor computers størrelse og regnehastighed. En mikrocomputers centrallager har en begrænset størrelse. Skal man udføre regninger, hvori der indgår "store" hele tal, må disses størrelse begrænses af, at man skal kunne udføre de nødvendige operationer med dem indenfor det område af maskinens centrallager, som vort program har til rådighed. Det andet problem, at man skal tage hensyn til computerens regnehastighed, bliver man også hurtigt klar over, når man udfører regninger med "store" hele tal. Det viser sig nemlig, at operationer med disse tal kan tage ret lang tid for en mikrocomputer. Multiplikation af 2 hele tal hver med 100 cifre kræver eksempelvis over 20000 enkeltregninger; og dette kræver selv i en forholdsvis hurtig computer en tid, som man må tage i betragtning.

Man må altså ved udvælgelsen af de opgaver, man tager op, og ved tilrettelæggelsen af de indgående regninger være opmærksom på, om det er en opgave, som ens computer har kapacitet til og kan udføre på en rimelig tid. Ved arbejdet med hele tal kan man komme ud for at skulle udføre de 6 grundlæggende regningsarter med dem. Vil man kunne tage de problemer op, som forekommer, må man have rutiner til enhver af disse 6 regningsarter. Det drejer sig om:

1) Addition.
   En additionsrutine vil som regel kunne beregne summen af et vilkårligt antal addender.
2) Subtraktion.
   Af en rutine for subtraktion må man forlange, at den skal kunne finde den numeriske differens mellem en minuend og en subtrahend med et vilkårligt antal cifre og angive, om differensen er positiv, 0 eller negativ.
3) Multiplikation.
   En rutine skal her kunne multiplicere 2 hele tal med et vilkårligt antal cifre.

Det skal bemærkes, at disse tre rutiner er ganske simple.

4) Division.
   En rutine til denne regningsart skal kunne bestemme kvotient og principalrest, hvis dividend og divisor er vilkårligt store hele tal.
5) Potensopløftning.
   En rutine er her forholdsvis simpel, da en potensopløftning er en udførelse af et vist antal multiplikationer efter hinanden. Ved mine egne regninger har jeg aldrig været ude for tilfælde, hvor eksponentens værdi ligger uden for computerens heltalsområde.
6) Roduddragning.
   Hvis man f.eks. vil løse opgaven: uddrag 7. rod af et 41-cifret tal, skal man have en rutine som kan bestemme rod og rest. Ved rest forstås her den mindste rest ≥ 0.

Routinerne for division og roduddragning er væsentlig mere komplicerede end de andre. Disse rutiner vil ikke blive gennemgået i dette indlæg.

I resten af denne artikel vil vi gennemgå et forholdsvis simpelt program, som kan anvendes til løsning af de såkaldte PELL'ske ligninger. Når denne opgave er valgt som eksempel, skyldes det særligt to forhold.

1) Mange problemer af aritmetisk og geometrisk natur kræver løsning af en sådan ligning.
2) En forholdsvis simpel programtekst kan anvendes til løsning af de fleste af de PELL'ske ligninger, man bliver stillet overfor ved anvendelserne.

Det teoretiske grundlag for løsningsmetoden vil her kun blive skitsemæssigt gennemgået.

Ved en PELL'sk ligning forstås en ligning af formen:

$$x^2-Dy^2 = 1$$

hvor konstanten D er hel og $D\geq 2$.

At løse ligningen vil sige at finde talpar (x,y), hvor x og y er hele tal, som stemmer i ligningen. Er D et kvadrattal, har ligningen kun løsningen $(x,y)=(1,0)$. Vi vil derfor gå ud fra, at D ikke er kvadratet på et helt tal. Har man først fundet en løsning (p,q) i en PELL'sk ligning, kan man finde en hel serie af løsninger ved simple regninger. Lad (p,q) være en løsning i ligningen $x^2-Dy^2=1$. Vi har så:

$$x^2-Dy^2 = p^2-Dq^2 = 1;$$

da $1^n=1$, har man

$$x^2-Dy^2 = (p^2-Dq^2)^n = 1;$$

For n=2 får vi:

$$\begin{aligned} x^2-Dy^2 &= (p^2-Dq^2)^2 = 1;\\ (p^2-Dq^2)^2 &= p^4+D^2q^4-2Dp^2q^2\\ &= p^4+2Dp^2q^2+D^2q^4-4D^2q^2\\ &= (p^2+Dq^2)^2-(2pq)^2D = 1. \end{aligned}$$

Lad os betegne de 2 nu kendte værdisæt (p_1,q_1) og (p_2,q_2). Vi har altså:

$$(p_1,q_1) = (p,q);\quad (p_2,q_2) = (p^2+Dq^2,2pq)$$

Vi vil nu finde et 3. værdisæt (p_3,q_3):

$$\begin{aligned} x^2-Dy^2 &= 1 = (p^2-Dq^2)^3\\ (p^2-Dq^2)^3 &= (p^2-Dq^2)^2(p^2-Dq^2)\\ &= [(p^2+Dq^2)^2-D(2pq)^2](p^2-Dq^2)\\ &= (p_2^2-Dq_2^2)(p_1^2-Dq_1^2)\\ &= (p_1^2p_2^2+D^2q_1^2q_2^2)-D(p_1^2q_2^2+q_1^2p_2^2) = 1\\ &= (p_1^2p_2^2+2p_1p_2q_1q_2D+D^2q_1^2q_2^2)\\ &\quad -D(p_1^2q_1^2+2p_1p_2q_1q_2+q_1^2p_2^2)\\ &= (p_1p_2+Dq_1q_2)^2-D(p_1q_2+p_2q_1)^2 = 1 \end{aligned}$$

Vi har altså fundet:

$$(p_3,q_3) = (p_1p_2+Dq_1q_2,p_1q_2+p_2q_1)$$

Ved at fortsætte dette ræsonnement kan man komme til formlen:

$$(p_{n+1},q_{n+1}) = (p_1p_n+Dq_1q_n,p_1q_n+p_nq_1)$$

Vi ser altså, at hvis man kan finde én løsning og helst den mindste løsning i ligningen

$$x^2-Dy^2 = 1,$$

kan man bestemme en hel serie af løsninger.

Vi mangler nu at finde den mindste løsning i ligningen $x^2-Dy^2=1$. Hertil anvender vi teorien for kædebrøker. En størrelse af formen:

$$a_1+\cfrac{1}{a_2+\cfrac{1}{a_3+\cfrac{1}{a_4+\cdots}}}$$

kaldes en kædebrøk.

Findes der i en kædebrøk kun et endeligt antal af konstanter a_k, kaldes kædebrøken endelig eller sluttet. En sluttet kædebrøk kan omskrives til et rationalt tal (en brøk). Eksempel:

$$\frac{r}{s} = a_1+\cfrac{1}{a_2+\cfrac{1}{a_3+\cfrac{1}{a_4}}}$$

Tager man kun en del af kædebrøkens led med, får man en tilnærmelse af brøken r/s. De forskellige tilnærmelser kaldes brøkens konvergenter (se næste side).

Vi vil nu fremhæve 3 ting:

1) Konvergenterne approksimerer brøken r/s

$$\frac{r_1}{s_1} < \frac{r}{s},\ \frac{r_2}{s_2} > \frac{r}{s},\ \frac{r_3}{s_3} < \frac{r}{s},\ \frac{r_4}{s_4} = \frac{r}{s}.$$

2) Har vi en kædebrøk med n konstanter $a_1,a_2,\ldots,a_n$ kan man ved at de ovenstående regninger nå til formlen:

$$\frac{r_{k+1}}{s_{k+1}} = \frac{a_{k+1}r_k+r_{k-1}}{a_{k+1}s_k+s_{k-1}}$$

3) Ved hjælp af denne formel kan man successivt bestemme konvergenterne og sluttelig brøken r/s ud fra talsættet $(a_1,a_2,\ldots,a_n)$ i en sluttet kædebrøk.

Skal man finde den mindste løsning i ligningen $x^2-Dy^2=1$, benytter man kædebrøksfremstillingen for $\sqrt{D}$. Det viser sig, at en kvadratrod kan repræsenteres ved en uendelig kædebrøk. Denne kædebrøk er ''periodisk''; d.v.s. at konstanterne a_j gentager sig med en vis periode.

Metoden med successivt at bestemme konvergenterne kan også bruges ved kvadratrødder, da denne formels udledning ikke forudsætter at kædebrøken er sluttet. Skal man finde konvergenterne for $\sqrt{D}$ er man i den situation, at man i starten ikke kender kædebrøkens konstanter $a_1,a_2,a_3,\ldots$

Vi vil begynde med at angive formlerne for konvergenterne:

Konvergenter til r/s

$$\frac{r_1}{s_1} = \frac{a_1}{1}$$

$$\frac{r_2}{s_2} = a_1 + \frac{1}{a_2} = \frac{a_1a_2+1}{a_2} = \frac{r_1a_2+1}{s_1a_2}$$

$$\frac{r_3}{s_3} = a_1 + \frac{1}{a_2 + \frac{1}{a_3}} = a_1 + \frac{1}{\frac{a_2a_3+1}{a_3}} = a_1 + \frac{a_3}{a_2a_3+1} = \frac{a_1(a_2a_3+1)+a_3}{a_2a_3+1}$$

$$= \frac{a_1a_2a_3+a_1+a_3}{a_2a_3+1} = \frac{a_3(a_1a_2+1)a_1}{a_2a_3+1} = \frac{a_3r_2+r_1}{a_3s_2+s_1}$$

$$\frac{r_4}{s_4} = \frac{r}{s} = a_1 + \frac{1}{a_2 + \frac{1}{a_3 + \frac{1}{a_4}}} = a_1 + \frac{1}{a_2 + \frac{1}{\frac{a_3a_4+1}{a_4}}} = a_1 + \frac{1}{a_2 + \frac{a_4}{a_3a_4+1}}$$

$$= a_1 + \frac{1}{\frac{a_2a_3a_4+a_2+a_4}{a_3a_4+1}} = a_1 + \frac{a_3a_4+1}{a_2a_3a_4+a_2+a_4} = \frac{a_1(a_2a_3a_4+a_2+a_4)+a_3a_4+1}{a_2a_3a_4+a_2+a_4}$$

$$= \frac{a_4(a_1a_2a_3+a_1+a_3)+(a_1a_2+1)}{a_4(a_2a_3+1)+a_2} = \frac{a_4r_3+r_2}{a_4s_3+s_2}$$

$P_1=0$; $Q_1=1$; $a_1=$ hele del af $\sqrt{D}$; $p_1=a_1$; $q_1=1$

$P_2=a_1$; $Q_2=D-a_1^2$; $a_2=$ hele del af $\dfrac{2a_1}{Q_2}$;

$p_2=a_2a_1+1$; $q_2=a_2$

$P_n=a_{n-1}Q_{n-1}-P_{n-1}$; $Q_n=\dfrac{D-P_n^2}{Q_{n-1}}$

$a_n=$ hele del af $\dfrac{a_1+P_n}{Q_n}$

$p_n=a_np_{n-1}+p_{n-2}$; $q_n=a_nq_{n-1}+q_{n-2}$

Ved hjælp af disse formler kan vi forholdsvis let beregne konvergenterne p_n/q_n af kvadratroden, hvis D er et lille tal.

Beregning af konvergenter for $\sqrt{7}$

I tabellen herunder er konvergenterne for $\sqrt{7}$ beregnet ved hjælp af formlerne op til n=13. For at antyde et ''bevis'' for denne fremgangsmåde, beregnes kædebrøksfremstillingen af $\sqrt{7}$ øverst på næste side, og konvergenterne beskrives nederst på næste side.

Værdier for konvergenter af $\sqrt{7}$

n	1	2	3	4	5	6	7	8	9	10	11	12	13
P	0	2	1	1	2	2	1	1	2	2	1	2	2
Q	1	3	2	3	1	3	2	3	1	3	2	3	1
a	2	1	1	1	4	1	1	1	4	1	1	1	4
p	2	3	5	8	37	45	82	127	590	717	1307	2024	9403
q	1	1	2	3	14	17	31	48	223	271	494	765	3554

Vi ser, at udtrykket $\dfrac{1}{\sqrt{7}-2}$, som forekommer til sidst ved P_6 også forekom i andet led ved P_2. Regningerne vil altså nu gentage sig på samme måde igen; vi har fundet kædebrøkens periode. Perioden for P, Q og a indeholder altså 4 led. Dette ser man let ud fra det beregnede skema.

Vi ser, at der gælder det samme m.h.t. konvergenterne her som ved den sluttede kædebrøk, de er skiftevis større end og mindre end det tal, de skal approksimere. Vi ser endelig, at der gælder formlen:

$$p_k^2 - 7q_k^2 = (-1)^kQ_{k+1}$$

Skal man løse ligningen $x^2-Dy^2=1$, skal man i talrækken for Q'erne finde et ulige n, hvor Q=1. Så har man:

$$p_{n-1}^2 - Dq_{n-1}^2 = 1$$

Som før bemærket: der kan siges 2 ting om det program, der er medtaget i denne artikel.

1) Problemet med løsning af PELL'ske ligninger er valgt, fordi det kan gennemføres med en forholdsvis simpel programtekst.
2) Det er tilstræbt at gøre programteksten så enkel som muligt. Det er så overladt til den enkelte læser at gøre den mere ''avanceret'' på sin egen måde.

Maskinen er en SPERRY mikrocomputer. For at størrelser skal opfattes som fixed-point tal i denne, skal deres identifikator (navn) slutte med tegnet %; dette gælder for simple variable og arrays. Man kan godt undgå %-tegnene ved at indføre deklarationssætninger; men det er ikke anvendt i dette program.

Vi vil forsøge at give en kort beskrivelse af regningernes forløb. Forkortelse: ℓ=linie. Konstanten D i den PELL'ske ligning betegnes `D1%` (`ℓ 100`). Er computerens øvre grænse for heltalsregning $2^{15}-1=32767$, vil programmet kunne klare ligninger med $D<32768$.

Man kan ved programmets hjælp lave en tabel over løsninger til ligningerne op til denne værdi af D. Selv har jeg kun regnet eksempler på ligninger med $D<10000$. For at løse ligningen får vi brug for de 2 størrelser

1) den hele rod af $\sqrt{D}$
2) resten.

Eksempel: D=13; hele rod =3; rest =4.

Kædebrøksfremstilling af $\sqrt{7}$

$$\sqrt{7} = 2+(\sqrt{7}-2) = 2+\cfrac{1}{\cfrac{1}{\sqrt{7}-2}} = 2+\cfrac{1}{\cfrac{\sqrt{7}+2}{3}} = 2+\cfrac{1}{1+\cfrac{\sqrt{7}-1}{3}} = 2+\cfrac{1}{1+\cfrac{1}{\cfrac{3}{\sqrt{7}-1}}} = 2+\cfrac{1}{1+\cfrac{1}{\cfrac{\sqrt{7}+1}{2}}}$$

$$= 2+\cfrac{1}{1+\cfrac{1}{1+\cfrac{\sqrt{7}-1}{2}}} = 2+\cfrac{1}{1+\cfrac{1}{1+\cfrac{2}{\sqrt{7}-1}}} = 2+\cfrac{1}{1+\cfrac{1}{1+\cfrac{1}{\cfrac{\sqrt{7}+1}{3}}}} = 2+\cfrac{1}{1+\cfrac{1}{1+\cfrac{1}{1+\cfrac{\sqrt{7}-2}{3}}}}$$

$$= 2+\cfrac{1}{1+\cfrac{1}{1+\cfrac{1}{1+\cfrac{1}{\cfrac{3}{\sqrt{7}-2}}}}} = 2+\cfrac{1}{1+\cfrac{1}{1+\cfrac{1}{1+\cfrac{1}{\cfrac{\sqrt{7}+2}{1}}}}} = 2+\cfrac{1}{1+\cfrac{1}{1+\cfrac{1}{1+\cfrac{1}{4+\cfrac{\sqrt{7}-2}{1}}}}} = 2+\cfrac{1}{1+\cfrac{1}{1+\cfrac{1}{1+\cfrac{1}{4+\cfrac{1}{\sqrt{7}-2}}}}}$$

Ved resten forstår man altså den mindste positive (principale) rest ved kvadratrodsuddragningen. Rest = 0 forekommer ikke, da programmet ikke kan løse ligninger, hvor D er kvadratet på et helt tal. En sådan ligning har som før nævnt kun løsningen $(p,q)=(1,0)$.

For at finde den hele rod og resten anvender programmet (`ℓ 140--230`) den enklest mulige metode. Man forsøger først med rod = 1,2,3,... o.s.v. til man finder den rigtige. Resultat: `K5%` = rod; `K4%` = rest.

For at finde en løsning på ligningen skal man, som før bemærket, beregne søjlerne i skemaet skridtvis (søjle 1,2,3,...,n) indtil vi finder en søjle med ulige n, hvor Q_n=1. Beregningen af de 2 første søjler adskiller sig fra beregningen af de følgende. I de første 2 søjler holder tallene sig indenfor computerens heltalsområde, så der er det ikke nødvendigt at bruge vore egne rutiner. `N10%` (`ℓ 240`) angiver søjle nr.

1. søjle: `ℓ 240-320`; der beregnes (P,Q,a_1,p_1,q_1);
resultat = (`P1%, Q1%, A1%, R1%, S1%`).
2. søjle: `ℓ 330-410`; der beregnes (P_2,Q_2,a_2,p_2,q_2);
resultat = (`P2%, Q2%, A2%, R2%, S2%`).

Vi går nu over til at beskrive beregningen af en vilkårlig søjle (søjle n). I dette program er der gået ud fra, at de 3 første rækker i søjlerne for alle n ligger indenfor computerens heltalsområde; for at denne betingelse skal være opfyldt er det nødvendigt at forlange, at $D <$ øvre grænse for dette område.

`ℓ 420-530` beregner P_n,Q_n og a_n.
Resultat: `P3%`=P_n; `Q3%`=Q_n; `A3%`=a_n.

For at finde de 2 størrelser p_n og q_n bruges formlerne:

$$p_n = a_n p_{n-1}+p_{n-2}$$
$$q_n = a_n q_{n-1}+q_{n-2}$$

Man kan komme ud for, at allerede p_3 og q_3 ligger udenfor vor computers heltalsområde.

For at beregne p og q for n>2 er det altså nødvendigt at anvende særlige rutiner, som vi selv konstruerer. Den simpleste måde at regne med disse "store hele tal" på er ved at lagre 1 ciffer i hver af vor computers celler. Lad os først se på n=3:

$$p_3 = a_3 p_2+p_1$$
$$q_3 = a_3 q_2+q_1$$

Vi ser, at der i formlerne for p_3 og q_3 indgår de 5 størrelser p_1,p_2,q_1,q_2 og a_3.

For at anvende vore "hjemmelavede" rutiner er det nødvendigt at spalte disse 5 tal op, så der ligger 1 ciffer i hver celle.

a_3 lagres i array `IAA%`

Konvergenter i udvikling af $\sqrt{7}$

$$\frac{p_1}{q_1} = \frac{2}{1}; \left(\frac{p_1}{q_1}\right)^2 = \frac{4}{1}<7;\ p_1^2-7q_1^2 = 4-7 = -3 = -Q_2$$

$$\frac{p_2}{q_2} = 2+\frac{1}{1}=\frac{3}{1}; \left(\frac{p_2}{q_2}\right)^2 = \frac{9}{1}>7;\ p_2^2-7q_2^2 = 2 = +Q_3$$

$$\frac{p_3}{q_3} = 2\cfrac{1}{1+\cfrac{1}{1}} = 2+\frac{1}{2} = \frac{5}{2}; \left(\frac{p_3}{q_3}\right)^2 = \frac{25}{4}<7;\ p_3^2-7q_3^2 = 25-28 = -3 = -Q_4$$

$$\frac{p_4}{q_4} = 2+\cfrac{1}{1+\cfrac{1}{1+\cfrac{1}{1}}} = 2+\cfrac{1}{1+\frac{1}{2}} = 2+\frac{2}{3} = \frac{8}{3}; \left(\frac{p_4}{q_4}\right)^2 = \frac{64}{9}>7;\ p_4^2-7q_4^2 = 64-63 = 1 = +Q_5$$

$$\frac{p_5}{q_5} = 2+\cfrac{1}{1+\cfrac{1}{1+\cfrac{1}{1+\cfrac{1}{4}}}} = 2+\cfrac{1}{1+\cfrac{1}{1+\cfrac{1}{1\frac{1}{4}}}} = 2+\cfrac{1}{1+\cfrac{1}{1\frac{4}{5}}} = 2+\cfrac{1}{1\frac{5}{9}} = 2+\frac{9}{14} = \frac{37}{14};$$

$$\left(\frac{p_5}{q_5}\right)^2 = \frac{1369}{196}<7;\ p_5^2-7q_5^2 = 1369-7\times196 = -3 = -Q_6$$

p_1 og p_2 i de 2 arrays IAR1% og IAR2%

q_1 og q_2 i de 2 arrays IAS1% og IAS2%

Disse beregninger udføres i ℓ 490–980; tallene spaltes her op i cifre og antallet af cifre i hvert tal bestemmes.

At problemet med løsning af PELL'ske ligninger er så simpelt skyldes, at der kun skal anvendes addition og multiplikation for at bestemme p_n og q_n.

Vi ser først på addition. Vi udfører den på computeren på samme måde som vi selv adderer på papir: Først lægges enerne sammen, menten overføres og dernæst lægges tierne sammen osv.

ℓ 1210–1290 er udført additionen i ligningen for p.

K3% er mente; denne er 0 fra start.

Vi har de 2 addender i de 2 arrays IAR3% og IAR1%; summen lagres i IAR3%.

ℓ 1250: man adderer mente + 1 ciffer fra hver addend.

ℓ 1260–1270: opspaltning i sum og ny mente.

ℓ 1220: Ved at addere 1 gang mere end antal cifre i noget af tallene undgår man beskrivelse af, at menten skal med som ciffer, når man adderer sidste gang.

Vi ser nu på multiplikation. Her anvender man også samme princip på computeren som når man regner "i hånden"; men man udfører multiplikation og addition samtidigt for at spare plads i maskinen.

ℓ 990–1200 er udført multiplikationen i p-ligningen. K9% er additionsmenten; K7% er multiplikationsmenten. a_3 er som før sagt lagret i array IAA%; denne anvendes som multiplikator; p er lagret i IAR2%; denne bruges som multiplikand. Produktet lagres i IAR3%, dette array er "nulstillet" før starten på multiplikationen.

ℓ 1080 og ℓ 1120: disse udtryk udfører forskydningen mod venstre.

ℓ 1030: Både ved multiplikation og addition går man 1 skridt længere end antallet af cifre i multiplikanden for at undgå en beskrivelse af, at menten skal med i produkt og sum sidste gang.

Efter multiplikationen er udført findes antal cifre i produktet (ℓ 1150–1200).

Slutningen af programmet ℓ 1940–2140 behandler skift til næste søjle. Her skal kun bemærkes en enkelt ting. Man skelner mellem søjle 3 og de følgende. Før man kan beregne p_3 og q_3 må man som før sagt spalte a_3,p_1,p_2,q_1 og q_2 op i cifre og bestemme cifferantal. Efter at p_3 og q_3 er beregnet er de allerede spaltet op i cifre; og man bestemmer deres cifferantal umiddelbart efter. ℓ 550–920 kan derfor overspringes for n>3. a_n skal spaltes op i cifre for alle n, da denne størrelse bliver beregnet ved hjælp af computerens almindelige rutine til fixed-point regninger.



Christian Marcussen er cand. scient. med astronomi som hovedfag. Arbejder ved Observatoriet for Rumforskning i Rude Skov ved Birkerød. Har gennem flere år lejlighedsvis som hobby arbejdet med talteoretiske problemer på computer.

```
10 T1%=0
20 C0%=100
30 DIM IAA%(C0%)
40 DIM IAR1%(C0%)
50 DIM IAS1%(C0%)
60 DIM IAR2%(C0%)
70 DIM IAS2%(C0%)
80 DIM IAR3%(C0%)
90 DIM IAS3%(C0%)
100 D1%=13
110 PRINT
120 PRINT "D=";D1%
130 PRINT
140 N9%=0
150 N9%=N9%+1
160 K1%=N9%^2
170 K2%=D1%-K1%
180 IF K2%=0 GOTO 2160
190 IF K2%<0 GOTO 240
200 K5%=N9%
210 K3%=K1%
220 K4%=K2%
230 GOTO 150
240 N10%=1
250 P1%=0
260 Q1%=1
270 A1%=K5%
280 R1%=A1%
290 S1%=1
300 PRINT
310 PRINT "N=";N10%
320 PRINT "P=";P1%;", Q=";
    Q1%;", a=";A1%;", p=";
    R1%;", q=";S1%
330 N10%=N10%+1
340 P2%=A1%
350 Q2%=K4%
360 A2%=(A1%+P2%)\Q2%
370 R2%=A1%*A2%+1
380 S2%=A2%
390 PRINT
400 PRINT "N=";N10%
410 PRINT "P=";P2%;", Q=";
    Q2%;", a=";A2%;", p=";
    R2%;", q=";S2%
420 N10%=N10%+1
430 P3%=A2%*Q2%-P2%
440 Q6%=D1%-P3%^2
450 Q3%=Q6%\Q2%
460 IF Q3%>1 GOTO 480
470 IF N10% MOD 2=1 THEN T1%=1
480 A3%=(A1%+P3%)\Q3%
490 IAA%(1)=A3% MOD 10
500 K1%=A3%\10
510 IAA%(2)=K1% MOD 10
520 K1%=K1%\10
530 IAA%(3)=K1%
540 IF N10%>3 GOTO 930
550 FOR N1%=1 TO 5
560 K1%=R1% MOD 10
570 R1%=R1%\10
580 IAR1%(N1%)=K1%
590 K1%=R2% MOD 10
600 R2%=R2%\10
610 IAR2%(N1%)=K1%
620 K1%=S1% MOD 10
630 S1%=S1%\10
640 IAS1%(N1%)=K1%
650 K1%=S2% MOD 10
660 S2%=S2%\10
670 IAS2%(N1%)=K1%
680 NEXT N1%
690 K10%=6
700 K10%=K10%-1
710 K1%=IAR1%(K10%)
720 IF K1%=0 GOTO 700
730 C10%=K10%
740 C11%=C10%+1
750 K10%=6
760 K10%=K10%-1
770 K1%=IAR2%(K10%)
780 IF K1%=0 GOTO 760
790 C12%=K10%
800 C13%=C12%+1
810 K10%=6
820 K10%=K10%-1
830 K1%=IAS1%(K10%)
840 IF K1%=0 GOTO 820
850 C14%=K10%
860 C15%=C14%+1
870 K10%=6
880 K10%=K10%-1
890 K1%=IAS2%(K10%)
900 IF K1%=0 GOTO 880
910 C16%=K10%
920 C17%=C16%+1
930 K10%=6
940 K10%=K10%-1
950 K1%=IAA%(K10%)
960 IF K1%=0 GOTO 940
970 C8%=K10%
980 C9%=K10%+1
990 FOR N2%=1 TO C9%
1000 K9%=0
1010 K7%=0
1020 K8%=IAA%(N2%)
1030 FOR N3%=1 TO C13%
1040 K1%=IAR2%(N3%)
1050 K2%=K8%*K1%+K7%
1060 K3%=K2% MOD 10
1070 K7%=K2%\10
1080 K5%=IAR3%(N2%+N3%-1)
1090 K6%=K3%+K5%+K9%
1100 K1%=K6% MOD 10
1110 K9%=K6%\10
1120 IAR3%(N2%+N3%-1)=K1%
1130 NEXT N3%
1140 NEXT N2%
1150 K10%=C13%+4
1160 K10%=K10%-1
1170 K1%=IAR3%(K10%)
1180 IF K1%=0 GOTO 1160
1190 C18%=K10%
1200 C19%=K10%+1
1210 K3%=0
1220 FOR N1%=1 TO C19%
1230 K1%=IAR3%(N1%)
1240 K2%=IAR1%(N1%)
1250 K4%=K1%+K2%+K3%
1260 K1%=K4% MOD 10
1270 K3%=K4%\10
1280 IAR3%(N1%)=K1%
1290 NEXT N1%
1300 K10%=C19%+2
1310 K10%=K10%-1
1320 K1%=IAR3%(K10%)
1330 IF K1%=0 GOTO 1310
1340 C18%=K10%
1350 C19%=K10%+1
1360 FOR N2%=1 TO C9%
1370 K9%=0
1380 K7%=0
1390 K8%=IAA%(N2%)
1400 FOR N3%=1 TO C17%
1410 K1%=IAS2%(N3%)
1420 K2%=K8%*K1%+K7%
1430 K3%=K2% MOD 10
1440 K7%=K2%\10
1450 K5%=IAS3%(N2%+N3%-1)
1460 K6%=K3%+K5%+K9%
1470 K1%=K6% MOD 10
1480 K9%=K6%\10
1490 IAS3%(N2%+N3%-1)=K1%
1500 NEXT N3%
1510 NEXT N2%
1520 K10%=C17%+4
1530 K10%=K10%-1
1540 K1%=IAS3%(K10%)
1550 IF K1%=0 GOTO 1530
1560 C20%=K10%
1570 C21%=C20%+1
1580 K3%=0
1590 FOR N1%=1 TO C21%
1600 K1%=IAS3%(N1%)
1610 K2%=IAS1%(N1%)
1620 K4%=K1%+K2%+K3%
1630 K1%=K4% MOD 10
1640 K3%=K4%\10
1650 IAS3%(N1%)=K1%
1660 NEXT N1%
1670 K10%=C21%+2
1680 K10%=K10%-1
1690 K1%=IAS3%(K10%)
1700 IF K1%=0 GOTO 1680
1710 C20%=K10%
1720 C21%=K10%+1
1730 PRINT
1740 PRINT "N=";N10%
1750 PRINT "P=";P3%;", Q=";
     Q3%;", a=";A3%
1760 K6%=0
1770 PRINT "p=";
1780 FOR N1%=1 TO C18%
1790 K6%=K6%+1
1800 K1%=IAR3%(C18%-N1%+1)
1810 PRINT K1%;
1820 IF K6% MOD 25=23
     THEN PRINT
1830 NEXT N1%
1840 PRINT
1850 PRINT "q=";
1860 K6%=0
1870 FOR N1%=1 TO C20%
1880 K6%=K6%+1
1890 K1%=IAS3%(C20%-N1%+1)
1900 PRINT K1%;
1910 IF K6% MOD 25=23
     THEN PRINT
1920 NEXT N1%
1930 PRINT
1940 P2%=P3%
1950 Q2%=Q3%
1960 A2%=A3%
1970 FOR N1%=1 TO C19%
1980 IAR1%(N1%)=IAR2%(N1%)
1990 IAR2%(N1%)=IAR3%(N1%)
2000 IAR3%(N1%)=0
2010 NEXT N1%
2020 FOR N1%=1 TO C21%
2030 IAS1%(N1%)=IAS2%(N1%)
2040 IAS2%(N1%)=IAS3%(N1%)
2050 IAS3%(N1%)=0
2060 NEXT N1%
2070 C10%=C12%
2080 C11%=C13%
2090 C14%=C16%
2100 C15%=C17%
2110 C12%=C18%
2120 C13%=C19%
2130 C16%=C20%
2140 C17%=C21%
2150 IF T1%=0 GOTO 420
2160 END
```