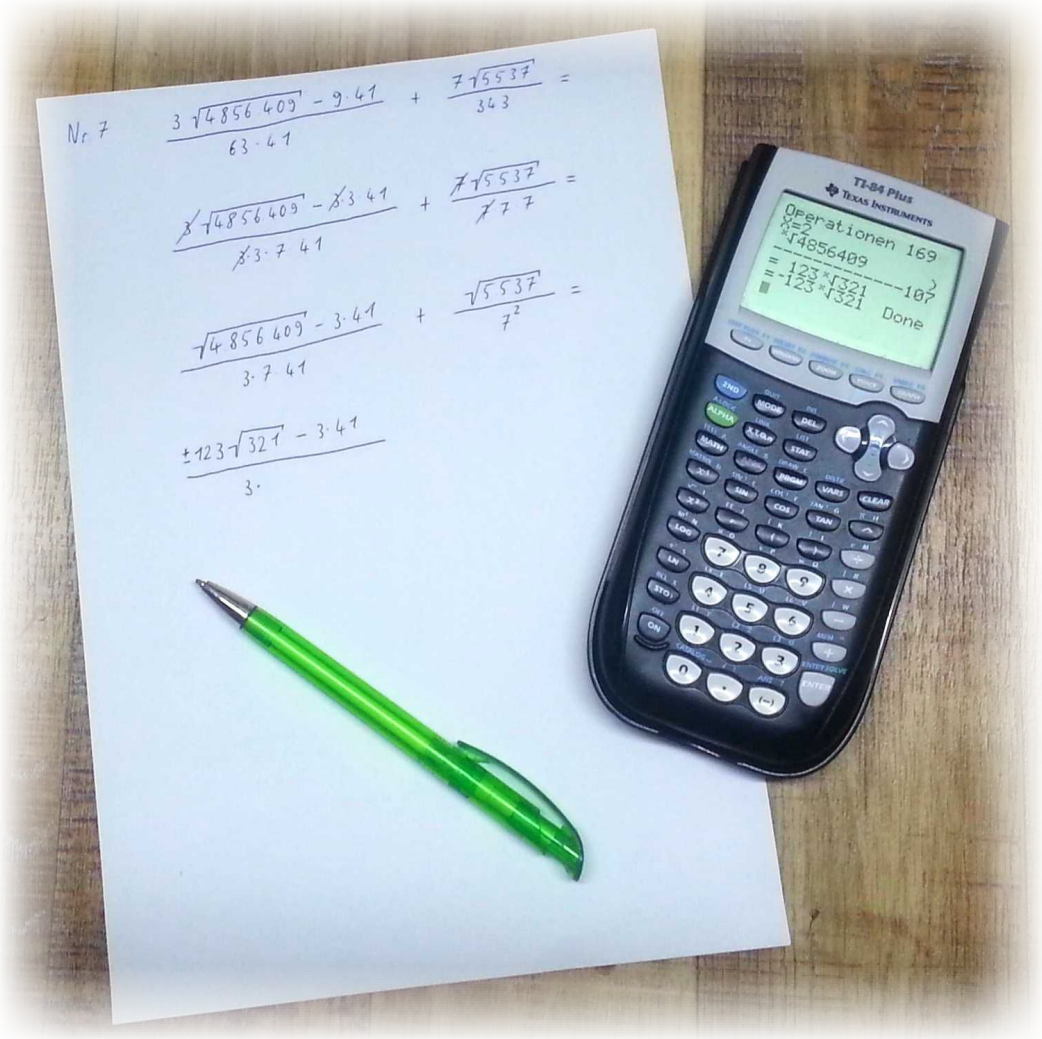
Teilweises Wurzelziehen

Programm für den Taschenrechner Texas Instruments TI-83 / TI-84



Inhaltsverzeichnis

1 Programmvorstellung

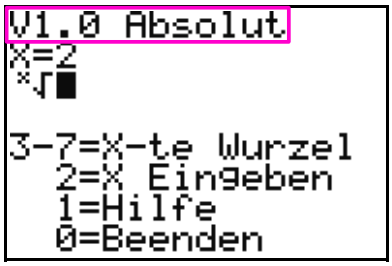


Abbildung 1: Startbild / Bildschirmaufnahme / 23.12.2017

Nachdem Sie das Programm gestartet haben, sehen Sie den Startbild des Programms. In der ersten Zeile wird die Versionsnummer und die Edition des Programms angezeigt (Rosa markiert). Sie können sofort eine beliebige Zahl eingeben, diese wird nach der Bestätigung mit der „Entertaste“ berechnet.

Wenn Sie die Zahl 2 eingeben, können Sie den Wurzelexponent selbst bestimmen. Bei einer Eingabe zwischen 3 und 7, wird dies sofort als Wurzelexponent übernommen.

Dieses Programm zieht immer den größtmöglichen Faktor heraus.
Als Beispiel geben wir 5.537 ein.

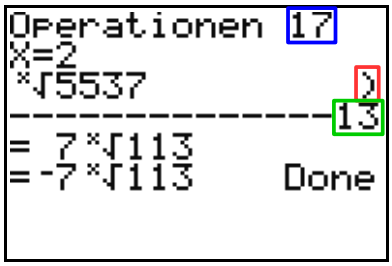


Abbildung 2: Ergebnis / Bildschirmaufnahme / 23.12.2017

Diese hier beschriebene Visualisation wird nur bei der Edition „*TEILWEIA (V1.0 **Absolut**)*“ ausgegeben.
Maximale Rechenoperationen werden hier eingeblendet (Blau markiert).

Die Klammer hinter der eingegebenen Zahl bedeutet das die „Hälfte“ der Rechenzeit / Operationen erreicht wurde (Rot markiert).

Die benötigten Rechenoperationen werden hier angezeigt (Grün markiert).

1.1 Editionen

- **Absolut**. (TEILWEIA)
- **Mini**, alle unnötige Elemente entfernt z.B. schnellere Quadratwurzelberechnung, Operationen anzeige, benötigte Operationen, Hilfe. (TEILWEI**M**)
- Englische Edition **Absolut**. (SIMPLIF**A**)
- Englische Edition **Mini**. (SIMPLIF**M**)

Programme mit den Namen „Locked“ sind nicht editierbare Programme die versehentlich verändert werden können.
z.B.: TEILWEIM (V1.0 Mini, Locked).

1.2 Speicherplatz

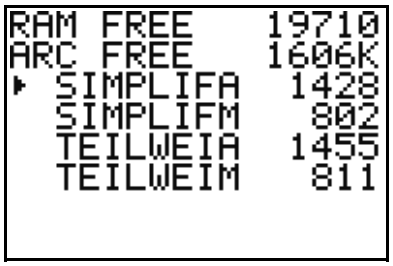


Abbildung 3: Speicherplatzverbrauch / Bildschirmaufnahme / 19.12.2017

2 Installationsbeschreibung

Sie können die Programmeditionen „*Absolut*“ und „*Mini*“, entweder in den Taschenrechner eintippen oder per Kabel „*TI-Graph Link*“ mit dem Taschenrechner verbinden und übertragen.

- Downloaden Sie die aktuellste Software von „*TI Connect*“.
  Internetadresse:
  https://education.ti.com/en/us/software/details/en/14D11109C9F44D55B9BBF65E5A62E7F1/swticonnectsoftwareforwindows
- Installieren Sie die Software.
- Verbinden Sie jetzt den Rechner mit dem Taschenrechner mit Hilfe von dem TI-Graph Link Kabel.

Abbildung 4: TI-Graph Link / Selbsterstellte Unterlage / 08.10.2015

- Öffnen Sie TI Connect.

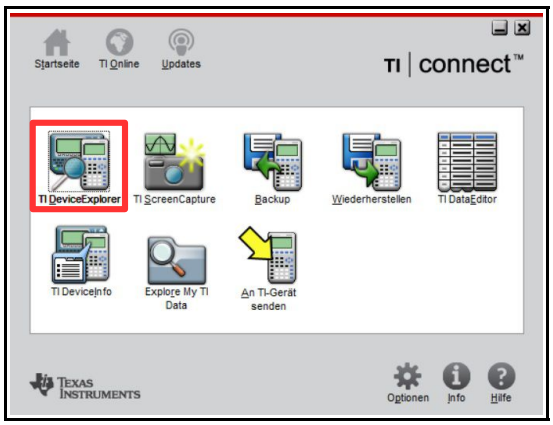
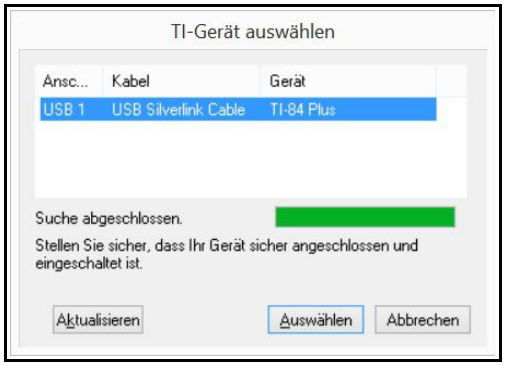


Abbildung 5: TI Connect / Bildschirmaufnahme / 08.10.2015

- Klicken Sie auf TI DeviceExplorer (Rot markiert).



Wählen Sie Ihren USB-Anschluss aus.
Betätigen Sie „*Auswählen*“.

Abbildung 6: Gerät auswählen / Bildschirmaufnahme / 08.10.2015

- Es öffnet sich nun folgendes Fenster.

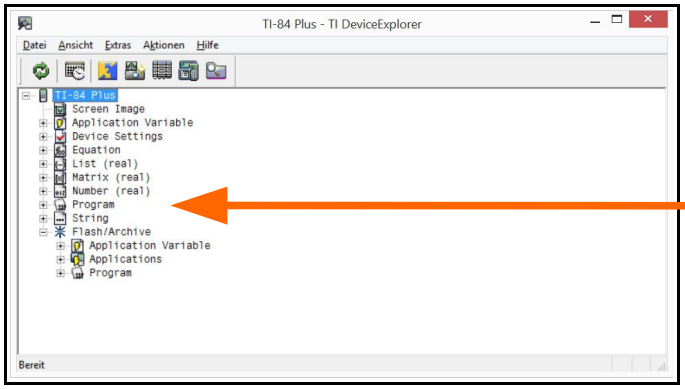


Abbildung 7: TI DeviceExplorer / Bildschirmaufnahme / 08.10.2015

- Aus den entpackten Ordner „*Simplify Roots # Teilweises Wurzelziehen V1.0*“ das Sie gedownloadet haben, wählen Sie eine Sprache und eine Edition aus. z.B.: „TEILWEIM *(V1.0 Mini, Locked)*“

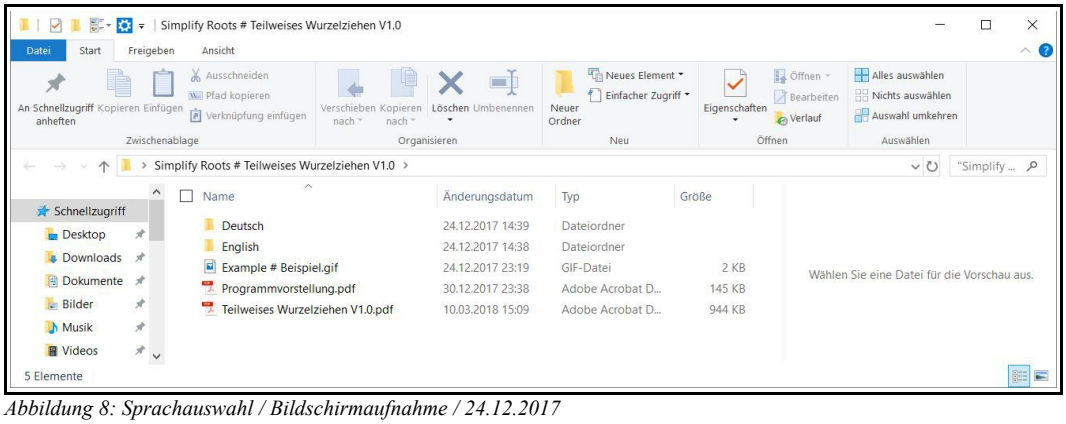


Abbildung 8: Sprachauswahl / Bildschirmaufnahme / 24.12.2017

- Schieben Sie die Datei, die Sie sich ausgewählt haben, in den TI DeviceExplorer (Oranger Pfeil).

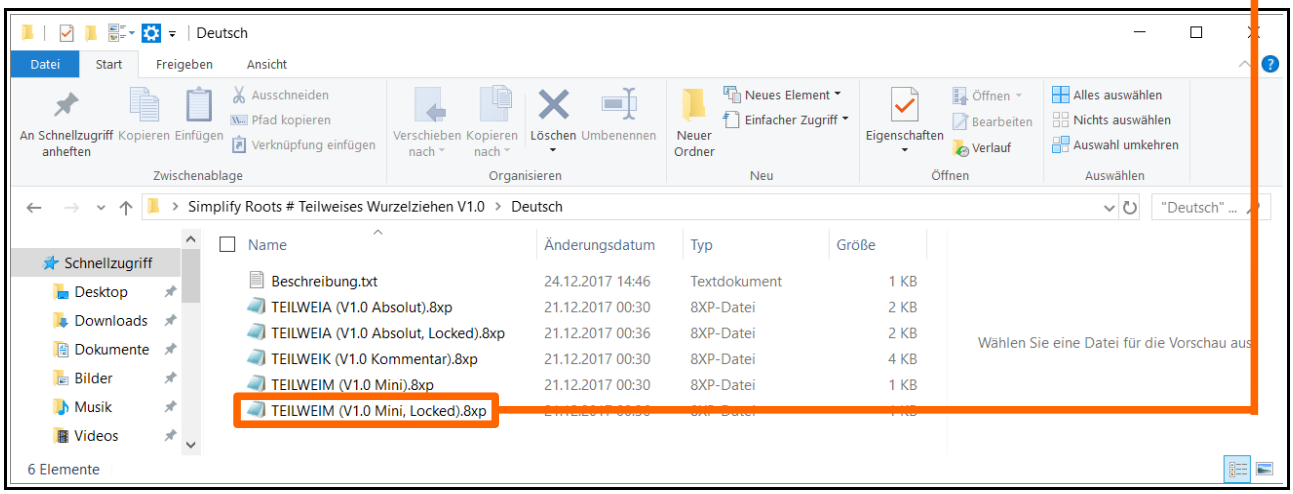


Abbildung 9: Edition / Bildschirmaufnahme / 24.12.2017

- Ab sofort können Sie das Programm im Taschenrechner aufrufen und ausführen.

3 Vorwort

Programmentwickler: Jakob Michel.
Anfertigung der Dokumentation: Jakob Michel.
Ich bin kein Mathematiker, somit sind alle mathematische ausdrücke womöglich nicht korrekt.

Die größtmögliche Zahl die Eingegeben werden kann ist 10.000.000.000 oder kurz 10^{10}.
Da der Taschenrechner rundet kann auch maximal 10.000.000.004 oder kurz $10^{10} + 4$ eingegeben werden, ergibt aber glücklicherweise kein Lösung.

Das Programm „*TEILWEI**K** (V1.0 **Kommentar)***“ ist für diejenigen, die das Programm weiter verbessern, für sich selbst anpassen oder einfach nur lernen wollen wie das Programm funktioniert.

Alle angegebene Rechenzeiten wurden mit einem echten TI-84 Plus Texas Instruments Taschenrechner ermittelt und sind mit einer Toleranz von ± 0,5 Sekunden (500ms) zu versehen. Können sich aber je nach Batteriestand und der Speichernutzung variieren.

4 Aufgabenstellung

Aufgabe ist aus einem Wurzelausdruck einen Faktor herauszufiltern, um den Wurzelausdruck zu verkleinern.

$\sqrt{C} = A \times \sqrt{B}$ mit: $A \in \mathbb{N},\ B \in \mathbb{N},\ C \in \mathbb{N}$
(Die Variablen A, B und C werden je nach Kontext mal Zahl, Faktor oder Variable genannt)

Wenn wir eine der beiden Faktoren finden möchten (A oder B), gibt es verschiedene Methoden dies herauszufinden.

Es gibt zwei Möglichkeiten das vorgestellte Problem zu lösen.
Möglichkeit 1: Man probiert alle möglichen Zahlen aus um eine Lösung zu finden.
Möglichkeit 2: Die Zahl wird in Primfaktoren zerlegt und somit die Lösung berechnet.

Bei der Möglichkeit 1 gibt es folgende Nachteile:
Da keine Primzahltabelle vorliegt dauert die Berechnung bei dieser Methode länger, da jede Zahl ausprobiert wird.

Bei der Möglichkeit 2 gibt es folgende Nachteile:
Da es von 2-100.000 genau 9.592 Primzahlen gibt, müssten im schlimmsten Fall 9.692 Rechenoperationen durchgeführt werden, das ist meiner Meinung nach zuviel und dauert mit dem Taschenrechner zu lange. Zusätzlich brüchten wir noch eine Tabelle, mit all den Primzahlen, oder müssten die berechnen, das sprengt die Leistung des Taschenrechners.
Somit scheidet diese Möglichkeit aus, was an sich die bessere Methode wäre.

Warum bis 100.000?
Denn die maximale Genauigkeitsangabe sind 10 Stellen (9.999.999.999).

$\sqrt{9.999.999.999} \approx 100.000$

Wir nehmen als Beispiel die Zahl: $123^2 \times 321 = 4.856.409$. Die Primfaktoren sind $3 \times 3 \times 3 \times 41 \times 41 \times 107$.

Somit ist die Lösung: $\sqrt{123^2 \times 321} = \sqrt{3 \times 3^2 \times 41^2 \times 107} = 3 \times 41\sqrt{3 \times 107} = 123\sqrt{321} = 2.203{,}726\ldots$

Wir wissen jetzt dass der größte herauszunehmende Faktor nicht größer als 2.203 sein kann.
Nun wissen wir auch schon wieviele Rechenoperationen wir maximal brauchen werden, bis wir möglicherweise den Faktor gefunden haben, nämlich 2.203 Rechenoperationen.

Denn wir rechnen die Variable A runter von 2.203 bis auf 2, damit wir den größtmöglichsten Faktor herausnehmen können.
z.B.:

$$\sqrt{123^2 \times 321} = \sqrt{\frac{123^2 \times 321 \times 2.203^2}{2.203^2}} = \sqrt{\frac{123^2 \times 321}{2.203^2} \times 2.203^2} = \sqrt{1{,}000\,659\ldots \times 2.203^2} = 2.203 \times \sqrt{1{,}000\,659\ldots}$$

$$\sqrt{123^2 \times 321} = \sqrt{\frac{123^2 \times 321 \times 2.202^2}{2.202^2}} = \sqrt{1{,}001\,568\ldots \times 2.202^2} = 2.202 \times \sqrt{1{,}001\,568\ldots}$$

$$\sqrt{123^2 \times 321} = 2.201 \times \sqrt{1{,}002\,478\ldots}$$

$$\sqrt{123^2 \times 321} = 2.200 \times \sqrt{1{,}003\,390\ldots}$$

Ist in der Wurzel eine Zahl mit keine Nachkommastellen, so haben wir den Faktor gefunden.

Programm A wird genau das tun.

Die Rechenoperationen werden für die For-schleife benötigt, zusätzlich kann man auch dadurch die maximale Rechendauer abschätzen.

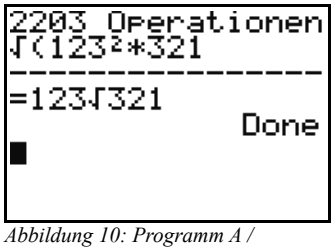
5 Programm A

```
"Variablen A Wurzel aussen, B Wurzel innen, C Eingabe,
D Zaehler, E Rechenoperationen, F Verschieben"
"Lbl 0-2"
Lbl 0
ClrHome
Disp ""
Output(1,1,"V0.1")
Input "sqrt(",C
"Rechenoperationen berechnen"
int(sqrt(C))->E
Output(1,1,E)
"Verschiebung
1->F
While iPart(E/(10^F))!=0
        F+1->F
End
Output(1,1+F," Operationen")
"Aeussere Wurzel"
For(D,int(sqrt(C)),2,~1)
        If fPart(C/D^^2)=0:Goto 1
End
"Kein Faktor gefunden"
Disp "Kein Faktor","gefunden.
Stop

"Aeussere Wurzel Loesung gefunden"
Lbl 1
C/D^^2->B
D->A
Disp "----------------"
Disp ""
Output(4,1,"=")
Output(4,2,A)
If B=1:Goto 2
"Zahl schieben"
1->F
While iPart(A/(10^F))!=0
        F+1->F
End
Output(4,2+F,"sqrt(")
Output(4,3+F,B)
Lbl 2
Stop
```

Tabelle 1: Programm A / Selbsterstellte Unterlage / 16.09.2015

~ (Tilde) ist im TI84 negatives Vorzeichen (nicht Minus).

A^^2	=	A^2
1|E10	=	1E10
sqrt(	=	$\sqrt{}$(
->	=	→



Dauer bis Lösungsfindung: circa 27 Sekunden.

Abbildung 10: Programm A / Bildschirmaufnahme / 16.09.2015

5.1 Warum For-schleife statt While-schleife

Schleife	For	While
Programm	For(A,0,2000) End Disp "Fertig Stop	0->A While A<2000 A+1->A End Disp "Fertig Stop
Benötigte Zeit:	2 Sekunden	9 Sekunden
Nachteil:	Nicht richtig Programmierbar. (Beispiel aus Programm A) For(D,int(sqrt(C)),2,~1) If fPart(C/D^^2)=0:Goto 1 End Lbl 1	Zeit

Tabelle 2: For- Whileschleife / Selbsterstellte Unterlage / 16.09.2015

5.2 Erste Optimierung

Wir rechnen mal die ersten Schritte nach:

$$\sqrt{123^2 \times 321} = 2.203 \times \sqrt{1{,}000659...}$$

$$\sqrt{123^2 \times 321} = 2.202 \times \sqrt{1{,}001568...}$$

$$\sqrt{123^2 \times 321} = 2.201 \times \sqrt{1{,}002478...}$$

$$\sqrt{123^2 \times 321} = 2.200 \times \sqrt{1{,}003390...}$$

Nun, wir merken dass die Zahl in der Wurzel langsam ansteigt.
Die nächste Natürliche Zahl in der Wurzel wäre die 2, alle dazwischen sind unnötige Rechenoperationen.

Somit:

$$\sqrt{123^2 \times 321} = \sqrt{\frac{123^2 \times 321 \times 2}{2}} = \sqrt{\frac{123^2 \times 321}{2} \times 2} = \sqrt{\frac{123^2 \times 321}{2}} \times \sqrt{2} = \sqrt{2.428.204{,}5} \times \sqrt{2} = 1.558{,}269713... \times \sqrt{2}$$

Da jetzt die äußere Zahl A eine Nachkommastellen hat, ist dieses Ergebnis keine Lösung.
Wenn wir jetzt das Programm A ab hier ablaufen lassen würden, brüchten wir noch 1.558 Rechenoperationen.
Machen wir mal weiter.

$$\sqrt{123^2 \times 321} = \sqrt{\frac{123^2 \times 321 \times 3}{3}} = \sqrt{1.618.803} \quad \times \sqrt{3} = 1.272{,}321892... \times \sqrt{3}$$

$$\sqrt{123^2 \times 321} = \sqrt{\frac{123^2 \times 321 \times 4}{4}} = \sqrt{1.214.102{,}25} \times \sqrt{4} = 1.101{,}86308... \times \sqrt{4}$$

Zählen wir in der Wurzel hoch sehen wir, dass die äußere Zahl A rapide fällt.
So haben wir nur mit 3 Rechenoperationen, wenn wir jetzt die Strategie von Programm A anwenden würden, die möglichen Rechenoperationen halbiert.

Die nachfolgenden Schritte wird der Taschenrechner nicht durchführen,
den er würde die Lösung mit dem Faktor 321 gefunden haben.
Folgende Schritte wären die Letzten:

$$\sqrt{123^2 \times 321} = \sqrt{\frac{123^2 \times 321 \times 1.214.101}{1.214.101}} = \sqrt{4{,}0000041...} \times \sqrt{1.214.101} = 2{,}000\,001\,03... \times \sqrt{1.214.101}$$

$$\sqrt{123^2 \times 321} = 2{,}000\,000\,206... \times \sqrt{1.214.102}$$

$$\sqrt{123^2 \times 321} = 1{,}999\,999\,382... \times \sqrt{1.214.103}$$

Würden wir in der Wurzel immer hochzählen, benötigen wir maximal 1.214.103 Rechenoperationen.
Dieses Programm würde erheblich länger dauern als Programm A.
Also folglich zählen wir erst die Variable B in der Wurzel hoch, springen „irgendwann“ heraus und Zählen dann die äußere Variable A runter.

Die Frage ist: Ab welchen Faktor müssen wir aus der Wurzel heraus und die äußere Variable runter Zählen?

Besser gefragt:
Welcher Faktor ist sowohl in der Wurzel (Zahl B) und sowohl auch außen (Zahl A)?

$$C = X \times X \times X = X^3$$

$$\sqrt{C} = \sqrt{X^3} = \sqrt{X^2 \times X} = X\sqrt{X}$$

$$\underline{\sqrt[3]{X^3} = X}$$

$$\sqrt[3]{123^2 \times 321} = 169{,}344755...$$

Probe:

$$\sqrt{\frac{123^2 \times 321}{169{,}344755}} = 169{,}344756$$

Hirsaus folgt das man um die Rechenoperationen zu berechnen man einfach die nächsthöhere Wurzelexponent benutzt und das Ergebnis mit 2 multipliziert (Variable B hochzählen und danach Variable A runterzählen).
Für den Wert: $\sqrt{123^2 \times 321}$ bräuchten wir jetzt nur noch $\sqrt[3]{123^2 \times 321} \times 2 \approx 169 \times 2 = 338$ Rechenoperationen.

$$\frac{(Operationen\ Programm\ A)\ \ 2203}{(Operationen\ Programm\ B)\ \ 338} = 6{,}51... \rightarrow$$ Um das 6,5 fache an Rechenoperationen haben wir damit eingespart.

Die Grüne Schrift bedeutet die Änderungen des neuen Programm gegenüber dem vorigem Programm.

6 Programm B

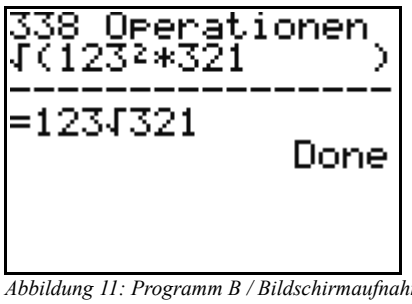
```
"Variablen A Wurzel aussen, B Wurzel innen, C Eingabe,
D Zaehler, E Rechenoperationen, F Verschieben"
"Lbl 0-4"
Lbl 0
ClrHome
Disp ""
Output(1,1,"V0.2")
Input "sqrt(",C
"Rechenoperationen berechnen"
int(cuberoot(C)*2)->E
Output(1,1,E)
"Verschiebung
1->F
While iPart(E/(10^F))!=0
        F+1->F
End
Output(1,1+F," Operationen")
"Innere Wurzel"
For(D,1,int(cuberoot(C)))
        If fPart(sqrt(C/D))=0:Goto 1
End
"Haelfte erreicht"
Output(2,16,")")
"Aeussere Wurzel"
For(D,D,2,~1)
        If fPart(C/D^^2)=0:Goto 2
End
"Kein Faktor gefunden"
Disp "Kein Faktor","gefunden.
Stop
```

```
"Innere Wurzel Loesung gefunden"
Lbl 1
(sqrt(C/D))->A
D->B
Goto 3


"Aeussere Wurzel Loesung gefunden"
Lbl 2
C/D^^2->B
D->A


"Loesung ausgabe"
Lbl 3
Disp "----------------"
Disp ""
Output(4,1,"=")
Output(4,2,A)
If B=1:Goto 4
"Zahl schieben"
1->F
While iPart(A/(10^F))!=0
        F+1->F
End
Output(4,2+F,"sqrt(")
Output(4,3+F,B)
Lbl 4
Stop
```

Tabelle 3: Programm B / Selbsterstellte Unterlage / 16.09.2015

~ (Tilde) ist im TI84 negatives Vorzeichen (nicht Minus).

A^^2	=	A^2
1|E10	=	1E10
sqrt(	=	$\sqrt{}$(
cuberoot(	=	$\sqrt[3]{}$(
->	=	→



Dauer bis Lösungsfindung: circa 4,5 Sekunden.

Abbildung 11: Programm B / Bildschirmaufnahme / 16.09.2015

6.1 Wurzel höheren Grades

Versuchen wir einen Faktorisirung des Wurzels mit einem höheren Grad.
z.B.: $\sqrt[3]{123^2 \times 321}$
Berechnung der Maximalen Rechenoperationen:
$\sqrt[4]{123^2 \times 321 \times 2} \approx 46 \times 2 = 92$

7 Programm C

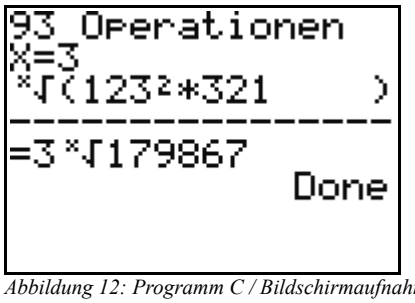
```
"Variablen A Wurzel aussen, B Wurzel innen, C Eingabe, D
Zaehler, E Rechenoperationen, F Verschieben, X Hochzahl"
"Lbl 0-5"
Lbl 0
ClrHome
Disp ""
Disp "X=2"
Output(1,1,"V0.3")
Output(7,1,"2-7=X-te Wurzel")
Output(8,3,"1=X Eingeben")
Input "xroot(",C
"2te-Wurzel"
If C>7
Then
        2->X
        Goto 5
End
"X-Wurzel"
ClrHome
Disp ""
Output(1,1,"V0.3")
"C=1"
If C=1
Then
        Output(3,1,"xroot(")
        Input "X=",X
        Input "xroot(",C

        Else
        "C=1-7"
        C->X
        Disp "X="
        Output(2,3,X)
        Input "xroot(",C
End
Lbl 5
"Rechenoperationen berechnen"
int((X+1)xroot(C)*2)->E
Output(1,1,E)
"Verschiebung
1->F
While iPart(E/(10^F))!=0
        F+1->F
End
Output(1,1+F," Operationen")
```

```
"Innere Wurzel"
For(D,1,int((X+1)xroot(C)))
        If fPart(Xxroot(C/D))=0:Goto 1
End
"Haelfte erreicht"
Output(3,16,")")
"Aeussere Wurzel"
For(D,D,2,~1)
        If fPart(C/D^X)=0:Goto 2
End
"Kein Faktor gefunden"
Disp "Kein Faktor","gefunden.
Stop


"Innere Wurzel Loesung gefunden"
Lbl 1
(Xxroot(C/D))->A
D->B
Goto 3


"Aeussere Wurzel Loesung gefunden"
Lbl 2
C/D^X->B
D->A


"Loesung ausgabe"
Lbl 3
Disp "----------------"
Disp ""
Output(5,1,"=")
Output(5,2,A)
If B=1:Goto 4
"Zahl schieben"
1->F
While iPart(A/(10^F))!=0
        F+1->F
End
Output(5,2+F,"xroot(")
Output(5,4+F,B)
Lbl 4
Stop
```

Tabelle 4: Programm C / Selbsterstellte Unterlage / 17.09.2015

~ (Tilde) ist im TI84 negatives Vorzeichen (nicht Minus).

A^^2	=	A2
1|E10	=	1E10
sqrt(	=	$\sqrt{}$(
xroot(	=	$\sqrt[x]{}$(
->	=	→



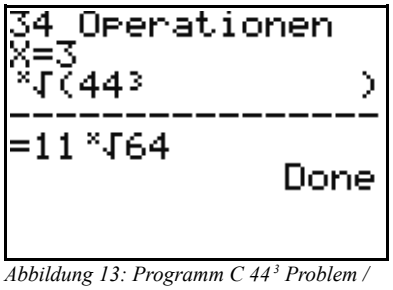
Dauer bis Lösungsfindung: circa 2 Sekunden.

Abbildung 12: Programm C / Bildschirmaufnahme / 17.09.2015

7.1 Probleme mit der Lösungsfindung

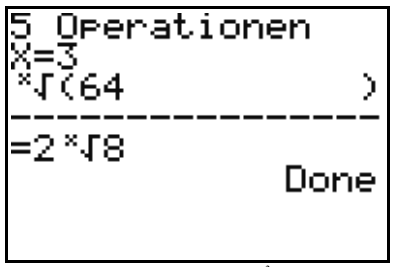
Leider tauchen einige Fehler auf, bei der Zerlegung der Zahl in der Wurzel höheren Grades.

z.B.: bei der Zahl :



$\sqrt[3]{44^3} = 44$

Abbildung 13: Programm C 44^3 Problem / Bildschirmaufnahme / 17.09.2015



$\sqrt[3]{64} = 4$

Abbildung 14: Programm C 4^3 Problem / Bildschirmaufnahme / 17.09.2015

Die Lösung ist nicht falsch aber wir wollen den größtmöglichsten Faktor herausziehen.
Wir müssen jetzt herausfinden woran das liegt.

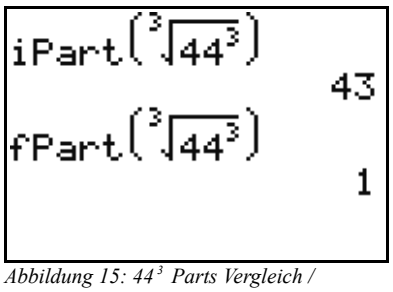


Abbildung 15: 44^3 Parts Vergleich / Bildschirmaufnahme / 18.09.2015

Leider gibt uns der Taschenrechner hier ein Fehler aus.

$\sqrt[3]{44^3}$ gibt eine Zahl mit nachkommastellen aus, obwohl das Ergebnis 44 ist (mit den Operatoren „iPart“ und „fPart“).

fPart Operation kann niemals 1 sein, da es nur irgendetwas mit 0,XXX… ausgeben müsste!

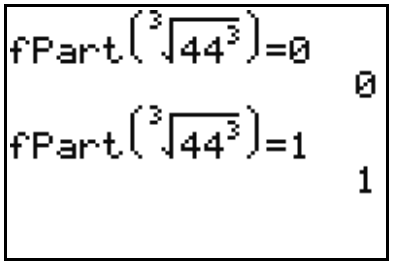


Abbildung 16: fPart gibt falschen Wert aus / Bildschirmaufnahme / 18.09.2015

Hier mit den Vergleichsoperatoren.

Desweiteren läuft der Operator $\sqrt[X]{}$([xroot(] langsamer ab als Operator $\sqrt{}$([sqrt(], wenn für X=2 eingesetzt wird.

Programm B benötigt für die Aufgabe: $\sqrt{123^2 \times 321}$ circa 4,5 Sekunden.
Programm C benötigt circa 7 Sekunden dafür.

8 Programm D (fPart=1)

```
"Variablen A Wurzel aussen, B Wurzel innen, C Eingabe, D Zaehler, E
Rechenoperationen, F Verschieben, X Hochzahl"
"Lbl 0-6"
Lbl 0
ClrHome
Disp ""
Disp "X=2"
Output(1,1,"V0.4")
Output(7,1,"2-7=X-te Wurzel")
Output(8,3,"1=X Eingeben")
Input "xroot(",C
"Initialisieren
1->A
"2te-Wurzel"
If C>7:Goto 5
"X-Wurzel"
ClrHome
Disp ""
Output(1,1,"V0.4")
"C=1"
If C=1
Then
        Output(3,1,"xroot(")
        Input "X=",X
        Input "xroot(",C

        Else
        "C=1-7"
        C->X
        Disp "X="
        Output(2,3,X)
        Input "xroot(",C
End
"Rechenoperationen berechnen"
int((X+1)xroot(C)*3)->E
Output(1,1,E)
"Verschiebung
1->F
While iPart(E/(10^F))!=0
        F+1->F
End
Output(1,1+F," Operationen")
"Innere Wurzel"
For(D,1,int((X+1)xroot(C)))
        If fPart(Xxroot(C/D))=0:Goto 1
        If fPart(Xxroot(C/D))=1:Goto 7
End
"Haelfte erreicht"
Output(3,16,")")
"Aeussere Wurzel"
For(D,D,2,~1)
        If fPart(C/D^X)=0:Goto 2
End
If A!=1:Goto 3
Goto 6


"Innere Wurzel Loesung gefunden"
Lbl 1
(Xxroot(C/D))->A
D->B
Goto 3

"Aeussere Wurzel Loesung gefunden"
Lbl 2
C/D^X->B
D->A


"Loesung ausgabe"
Lbl 3
Disp "----------------"
Disp ""
Output(5,1,"=")
Output(5,2,A)
If B=1:Goto 4
"Zahl schieben"
1->F
While iPart(A/(10^F))!=0
        F+1->F
End
Output(5,2+F,"xroot(")
Output(5,4+F,B)
Lbl 4
Stop


"Quadratwurzel berechnen"
Lbl 5
2->X
"Rechenoperationen berechnen
int(cuberoot(C)*2)->E
Output(1,1,E)
1->F
While iPart(E/(10^F))!=0
        F+1->F
End
Output(1,1+F," Operationen")
"Innere Wurzel
For(D,1,E)
        If fPart(sqrt(C/D))=0:Goto 1
End
"Haelfte erreicht
Output(3,16,")
"Aeussere Wurzel
For(D,D,2,~1)
        If fPart(C/D^^2)=0:Goto 2
End


"Kein Faktor gefunden"
Lbl 6
Disp "Kein Faktor","gefunden.
Stop


"fpart Fehler ausgleich
Lbl 7
(iPart(Xxroot(C/D))+1->A
D->B
Goto 3
```

Tabelle 5: Programm D / Selbsterstellte Unterlage / 11.10.2015

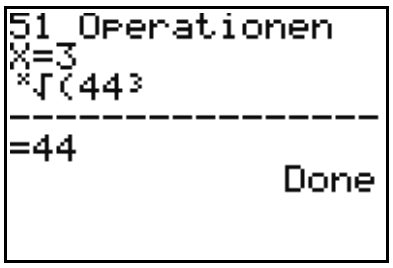


Abbildung 17: Programm D / Bildschirmaufnahme / 11.10.2015

Komischerweise funktioniert der Befehl: „*If fPart(Xxroot(C/D))=0 or 1:Goto 1*“ nicht.
Andersrum zu positionieren hilft auch nicht: „*If 0 or 1=fPart(Xxroot(C/D)):Goto 1*“.
Da diese Programmierung nicht funktioniert, erhöht sich der Rechenaufwand um circa 50%
(Blau markiert auf der Seite 14).

8.1 Zweite Optimierung (Gerade, Ungerade Zahlen)

Wir versuchen das Programm weiter zu beschleunigen indem wir vielleicht versuchen die Zahlen in Gerade (G) und Ungerade (U) einzuteilen. Eine Gerade oder Ungerade Zahl erkennt man nur an der letzten Zahl.

Hierzu die Gesetzmäßigkeiten:

Mengen:	Eigenschaft:
$x, y, z = n \in \mathbb{Z}$	$\{-\infty, \dots, -3, -2, -1, 0, 1, 2, 3, \dots, \infty\}$
$G = 2\mathrm{n}$ *für alle* $n \in \mathbb{Z}$ $G = \{n \in \mathbb{Z} \mid 2\mathrm{m} = n\}$	$\{-\infty, \dots, -6, -4, -2, 0, 2, 4, 6, \dots, \infty\}$ $G = 2\mathrm{x}$
$U = 2\mathrm{n}+1$ *für alle* $n \in \mathbb{Z}$ $U = \{n \in \mathbb{Z} \mid 2\mathrm{m}+1 = n\}$	$\{-\infty, \dots, -7, -5, -3, -1, 1, 3, 5, 7, \dots, \infty\}$ $U = 2\mathrm{x}+1$
$Q \in \mathbb{Q}$	Für unsere Gesetzmäßigkeit / Analyse nicht brauchbar.
$P \in \mathbb{P}$	$\{2, 3, 5, 7, 11, 13, 17, 19, 23, 29, 31, \dots, \infty\}$
$R \in \mathbb{R}$	Für unsere Gesetzmäßigkeit / Analyse nicht brauchbar.

Tabelle 6: Definition der Mengen / Selbsterstellte Unterlage / 29.09.2015

Regel:	Gesetzmäßigkeit:	Beweis:
Addition: (Mengen sind kommutativ $[G+U=U+G]$)	$G_1+G_2=G_3$	$G_1 + G_2 = 2x + 2y = 2\,(x + y) \rightarrow G_3$
	$G+U_1=U_2$ $U_1+G=U_2$	$G + U_1 = 2x + 2y + 1 = 2\,(x + y) + 1 \rightarrow U_2$
	$U_1+U_2=G$	$U_1 + U_2 = 2x + 1 + 2y + 1 = 2x + 2y + 2 = 2\,(x + y + 1) \rightarrow G$
Subtraktion:	$G_1-G_2=G_3$	$G_1 - G_2 = 2x - 2y = 2\,(x - y) \rightarrow G_3$
	$G-U_1=U_2$	$G - U_1 = 2x - (2y + 1) = 2x - 2y - 1 = 2x - 2y - 1 + 2 - 2 =$ $2x - 2y - 2 + 1 = 2\,(x - y - 1) + 1 \rightarrow U_2$
	$U_1-G=U_2$	$U_1 - G = 2x + 1 - 2y = 2\,(x - y) + 1 \rightarrow U_2$
	$U_1-U_2=G$	$U_1 - U_2 = 2x + 1 - (2y + 1) = 2x - 2y + 1 - 1 = 2\,(x + y) \rightarrow G$
Multiplikation: (Mengen sind kommutativ $[G\times U=U\times G]$)	$G_1\times G_2=G_3$	$G_1\times G_2 = 2x\times 2y = 2\,(2xy) \longrightarrow G_3$
	$G_1\times U=G_2$ $U\times G_1=G_2$	$G_1\times U = 2x\times(2y+1) = 4xy+2x = 2\,(2xy+x) \longrightarrow G_2$
	$U_1\times U_2=U_3$	$U_1\times U_2=(2x+1)\times(2y+1)=4xy+2x+2y+1=$ $2\,(2xy+x+y)+1 \longrightarrow U_3$
Division:	$\frac{G_1}{G_2}=Q, G_3, U$	$\frac{G_1}{G_2}=\frac{2x}{2y}=\frac{x}{y} \longrightarrow Q \text{ mit } y\neq 0,\ G_3 \text{ falls } x=(2z)\times y,\ U \text{ falls } x=(2z+1)\times y$
	$\frac{G_1}{U}=Q, G_2$	$\frac{G_1}{U}=\frac{2x}{2y+1}=2\left(\frac{x}{2y+1}\right) \longrightarrow Q,\ G_2 \text{ falls } x=(2y+1)\times z$
	$\frac{U_1}{G_1}=Q_1$	$\frac{U_1}{G_1}=\frac{2x+1}{2y}=\frac{2x}{2y}+\frac{1}{2y}=\frac{x}{y}+\frac{1}{2y} \longrightarrow Q_1 \text{ mit } y\neq 0$ $\frac{x}{y} \longrightarrow Q_2, G_2, U_2 \left(\text{siehe } \frac{G_1}{G_2}\right)$ $\frac{1}{2y} \longrightarrow Q_3 \text{ mit } Q_3 \leq \frac{1}{2} \text{ und } y\neq 0$ $Q_2, G_2, U_2 + Q_3 = Q_1$
	$\frac{U_1}{U_2}=Q, U_3$	$\frac{U_1}{U_2}=\frac{2x+1}{2y+1} \longrightarrow Q,\ U_3 \text{ falls } x=y \text{ oder } y=0$

Tabelle 7: Addition, Subtraktion, Multiplikation, Division / Selbsterstellte Unterlage / 29.09.2015

Regel:	Gesetzmäßigkeit:	Beweis:
Potenz:	$G_1{}^{G_2} = G_3$ $G_1{}^{U} = G_2$ } $G_1{}^{x} = G_2 \; mit \; x \neq 0$ $U_1{}^{G} = U_2$ $U_1{}^{U_2} = U_3$ } $U_1{}^{x} = U_2$	Siehe Multiplikation
Wurzel:	$\sqrt[G_1]{G_2} = R, G_4$ Teilergebnisse: $G_5 \sqrt[G_1]{G_6}$ $G_7 \sqrt[G_1]{U_7}$ $U_8 \sqrt[G_1]{G_8}$	$\sqrt[G_1]{G_2} = \sqrt[G_1]{G_3 \times U_3} = \sqrt[2x]{2y \times (2z+1)} = \sqrt[2x]{2y} \times \sqrt[2x]{2z+1}$ $\longrightarrow R, \; G_4 falls (2y)^{2x} \times (2z+1)^{2x}$ Teilergebnisse: $G_5 \sqrt[G_1]{G_6} \; falls \; (2y)^{2x+1} \; und \; z = 0$ $G_7 \sqrt[G_1]{U_7} \; falls \; (2y)^{2x} \; und \; z \neq 0$ $U_8 \sqrt[G_1]{G_8} \; falls \; (2z+1)^{2x}$ Ansonsten: $\sqrt[G_1]{G_2}$
	$\sqrt[U_1]{G_2} = R, G_4$ Teilergebnisse: $G_5 \sqrt[U_1]{G_6}$ $G_7 \sqrt[U_1]{U_7}$ $U_8 \sqrt[U_1]{G_8}$	$\sqrt[U_1]{G_2} = \sqrt[U_1]{G_3 \times U_3} = \sqrt[2x+1]{2y \times (2z+1)} =$ $\sqrt[2x+1]{2y} \times \sqrt[2x+1]{2z+1} \longrightarrow R, \; G_4 falls (2y)^{2x+1} \times (2z+1)^{2x+1}$ Teilergebnisse: $G_5 \sqrt[U_1]{G_6} \; falls \; (2y)^{2x+1+1} \; und \; z = 0$ $G_7 \sqrt[U_1]{U_7} \; falls \; (2y)^{2x+1} \; und \; z \neq 0$ $U_8 \sqrt[U_1]{G_8} \; falls \; (2z+1)^{2x+1}$ Ansonsten: $\sqrt[U_1]{G_2}$
	$\sqrt[G]{U_2} = R, U_3$ Teilergebnisse: $U_4 \sqrt[G]{U_5}$	$\sqrt[G]{U_2} = \sqrt[2x]{2y+1} \longrightarrow R, U_3 \; falls \; (2y+1)^{2x}$ Teilergebnisse: $U_4 \sqrt[G]{U_5} \; falls \; (2y+1)^{2x+1}$ Ansonsten: $\sqrt[G]{U_2}$
	$\sqrt[U_1]{U_2} = R, U_3$ Teilergebnisse: $U_4 \sqrt[U_1]{U_5}$	$\sqrt[U_1]{U_2} = \sqrt[2x+1]{2y+1} \longrightarrow R, U_3 \; falls \; (2y+1)^{2x+1}$ Teilergebnisse: $U_4 \sqrt[U_1]{U_5} \; falls \; (2y+1)^{2x+1+1}$ Ansonsten: $\sqrt[U_1]{U_2}$

Tabelle 8: Potenz, Wurzel / Selbsterstellte Unterlage / 29.09.2015

Der rot markierte Buchstabe entspricht der eingegebenen Zahl.
Die blauen Buchstaben sind Variablen die hoch oder runter gezählt werden.
In Klammern Gesetzter Term wird vom Taschenrechner ausgerechnet und vor die Wurzel gesetzt.
Grüne Ergebnisse können nicht als eine gefundene Lösung gezeigt werden, denn das Ergebnis wäre immer eine rationale Zahl Q, reelle Zahl R, oder ist nicht weiter zu vereinfachen.
Primzahlen können nie in zwei natürliche Faktoren geteilt werden.

Operation:	Ergebnis:	Beispiel:
Gerade Zahl wurde eingegeben (G).		
$\sqrt[x]{\left(\frac{G}{G}\right)\times G}$	$\sqrt[x]{G}$	Keine Lösung $\sqrt[3]{4}=\sqrt[3]{\left(\frac{2\times2}{2}\right)\times2}=R\ \sqrt[3]{2}$
	G	Keine Lösung $\sqrt[3]{8}=\sqrt[3]{\left(\frac{2\times2\times2}{2}\right)\times2}=R\ \sqrt[3]{2}$
	$G\sqrt[x]{G}$	$\sqrt[3]{16}=\sqrt[3]{\left(\frac{2\times2\times2\times2}{2}\right)\times2}=2\ \sqrt[3]{2}$
	$G\sqrt[x]{U}$	Keine Lösung $\sqrt[3]{24}=\sqrt[3]{\left(\frac{2\times2\times2\times3}{2}\right)\times2}=R\ \sqrt[3]{2}$
	$U\sqrt[x]{G}$	$\sqrt[3]{54}=\sqrt[3]{\left(\frac{2\times3\times3\times3}{2}\right)\times2}=3\ \sqrt[3]{2}$
$\sqrt[x]{\left(\frac{G}{U}\right)\times U}$	$\sqrt[x]{G}$	Keine Lösung $\sqrt[3]{4}=\sqrt[3]{\left(\frac{2\times2}{3}\right)\times3}=R\ \sqrt[3]{3}$
	G	Keine Lösung $\sqrt[3]{8}=\sqrt[3]{\left(\frac{2\times2\times2}{3}\right)\times3}=R\ \sqrt[3]{3}$
	$G\sqrt[x]{G}$	Keine Lösung $\sqrt[3]{16}=\sqrt[3]{\left(\frac{2\times2\times2\times2}{3}\right)\times3}=R\ \sqrt[3]{3}$
	$G\sqrt[x]{U}$	$\sqrt[3]{24}=\sqrt[3]{\left(\frac{2\times2\times2\times3}{3}\right)\times3}=2\ \sqrt[3]{3}$
	$U\sqrt[x]{G}$	Keine Lösung $\sqrt[3]{54}=\sqrt[3]{\left(\frac{2\times3\times3\times3}{3}\right)\times3}=R\ \sqrt[3]{3}$
$\sqrt[x]{\frac{G}{G^x}\times\left(G^x\right)}=G\sqrt[x]{\frac{G}{G^x}}$	$\sqrt[x]{G}$	Keine Lösung $\sqrt[3]{4}=\sqrt[3]{\frac{2\times2}{2^3}\times\left(2^3\right)}=2\ \sqrt[3]{Q}$
	G	$\sqrt[3]{8}=\sqrt[3]{\frac{2\times2\times2}{2^3}\times\left(2^3\right)}=2$
	$G\sqrt[x]{G}$	$\sqrt[3]{16}=\sqrt[3]{\frac{2\times2\times2\times2}{2^3}\times\left(2^3\right)}=2\ \sqrt[3]{2}$
	$G\sqrt[x]{U}$	$\sqrt[3]{24}=\sqrt[3]{\frac{2\times2\times2\times3}{2^3}\times\left(2^3\right)}=2\ \sqrt[3]{3}$
	$U\sqrt[x]{G}$	Keine Lösung $\sqrt[3]{54}=\sqrt[3]{\frac{2\times3\times3\times3}{2^3}\times\left(2^3\right)}=2\ \sqrt[3]{Q}$

Tabelle 9: Mögliche Ergebnisse bei geradene Zahlen / Selbsterstellte Unterlage / 30.09.2015

$\sqrt[x]{\frac{G}{U^x}\times\left(U^x\right)}=U\sqrt[x]{\frac{G}{U^x}}$	$\sqrt[x]{G}$	Keine Lösung $\sqrt[3]{4}=\sqrt[3]{\frac{2\times2}{3^3}\times\left(3^3\right)}=3\sqrt[3]{Q}$
	G	Keine Lösung $\sqrt[3]{8}=\sqrt[3]{\frac{2\times2\times2}{3^3}\times\left(3^3\right)}=3\sqrt[3]{Q}$
	$G\sqrt[x]{G}$	Keine Lösung $\sqrt[3]{16}=\sqrt[3]{\frac{2\times2\times2\times2}{3^3}\times\left(3^3\right)}=3\sqrt[3]{Q}$
	$G\sqrt[x]{U}$	Keine Lösung $\sqrt[3]{24}=\sqrt[3]{\frac{2\times2\times2\times3}{3^3}\times\left(3^3\right)}=3\sqrt[3]{Q}$
	$U\sqrt[x]{G}$	$\sqrt[3]{54}=\sqrt[3]{\frac{2\times3\times3\times3}{3^3}\times\left(3^3\right)}=3\sqrt[3]{2}$
Operation:	Ergebnis:	Beispiel:
Ungerade Zahl wurde eingegeben (U).		
$\sqrt[x]{\left(\frac{U}{G}\right)\times G}$	$Q\sqrt[x]{G}$	Keine Lösung
$\sqrt[x]{\left(\frac{U}{U}\right)\times U}$	$\sqrt[x]{U}$	Keine Lösung $\sqrt[3]{9}=\sqrt[3]{\left(\frac{3\times3}{3}\right)\times3}=R\sqrt[3]{3}$
	U	Keine Lösung $\sqrt[3]{27}=\sqrt[3]{\left(\frac{3\times3\times3}{3}\right)\times3}=R\sqrt[3]{3}$
	$U\sqrt[x]{U}$	$\sqrt[3]{81}=\sqrt[3]{\left(\frac{3\times3\times3\times3}{3}\right)\times3}=3\sqrt[3]{3}$
$\sqrt[x]{\frac{U}{G^x}\times\left(G^x\right)}=G\sqrt[x]{\frac{U}{G^x}}$	$G\sqrt[x]{Q}$	Keine Lösung
$\sqrt[x]{\frac{U}{U^x}\times\left(U^x\right)}=U\sqrt[x]{\frac{U}{U^x}}$	$\sqrt[x]{U}$	Keine Lösung $\sqrt[3]{9}=\sqrt[3]{\frac{3\times3}{3^3}\times\left(3^3\right)}=3\sqrt[3]{Q}$
	U	$\sqrt[3]{27}=\sqrt[3]{\frac{3\times3\times3}{3^3}\times\left(3^3\right)}=3$
	$U\sqrt[x]{U}$	$\sqrt[3]{81}=\sqrt[3]{\frac{3\times3\times3\times3}{3^3}\times\left(3^3\right)}=3\sqrt[3]{3}$
Hieraus folgt, das wenn man eine ungeradene Zahl eingibt, kann nur eine ungeradene Zahl abgezogen werden und/oder auch eine ungeradene Zahl in der Wurzel übrig bleiben. Deshalb können wir auch nur mit ungeradenen Zahlen die Wurzel auflösen.		

Tabelle 10: Mögliche Ergebnisse bei ungeradene Zahlen / Selbsterstellte Unterlage / 30.09.2015

9 Programm E (Gerade Ungerade)

```
"Variablen A Wurzel aussen, B Wurzel innen, C Eingabe, D Zaehler, E
Rechenoperationen, F Verschieben, G Gerade Ungerade, X Hochzahl"
"Lbl 0-6"
Lbl 0
ClrHome
Disp ""
Disp "X=2"
Output(1,1,"V0.5")
Output(7,1,"2-7=X-te Wurzel")
Output(8,3,"1=X Eingeben")
Input "xroot(",C
"Initialisieren
1->A
"2te-Wurzel"
If C>7:Goto 5
"X-Wurzel"
ClrHome
Disp ""
Output(1,1,"V0.5")
"C=1"
If C=1
Then
        Output(3,1,"xroot(")
        Input "X=",X
        Input "xroot(",C

        Else
        "C=1-7"
        C->X
        Disp "X="
        Output(2,3,X)
        Input "xroot(",C
End
"Zahl Ungerade"
1->G
If remainder(C,2)=1:2->G
"Rechenoperationen berechnen"
int((X+1)xroot(C)*3/G)->E
Output(1,1,E)
"Verschiebung
1->F
While iPart(E/(10^F))!=0
        F+1->F
End
Output(1,1+F," Operationen")
"Innere Wurzel"
For(D,1,int((X+1)xroot(C)),G)
        If fPart(Xxroot(C/D))=0:Goto 1
        If fPart(Xxroot(C/D))=1:Goto 7
End
"Haelfte erreicht"
Output(3,16,")")
"Aeussere Wurzel"
For(D,D,2,~G)
        If fPart(C/D^X)=0:Goto 2
End
If A!=1:Goto 3
Goto 6


"Innere Wurzel Loesung gefunden"
Lbl 1
(Xxroot(C/D))->A
D->B
Goto 3


"Aeussere Wurzel Loesung gefunden"
Lbl 2
C/D^X->B
D->A


"Loesung ausgabe"
Lbl 3
Disp "----------------"
Disp ""
Output(5,1,"=")
Output(5,2,A)
If B=1:Goto 4
"Zahl schieben"
1->F
While iPart(A/(10^F))!=0
        F+1->F
End
Output(5,2+F,"xroot(")
Output(5,4+F,B)
Lbl 4
Stop


"Quadratwurzel berechnen"
Lbl 5
2->X
"Zahl Ungerade"
1->G
If remainder(C,2)=1:2->G
"Rechenoperationen berechnen
int(cuberoot(C)*2/G)->E
Output(1,1,E)
1->F
While iPart(E/(10^F))!=0
        F+1->F
End
Output(1,1+F," Operationen")
"Innere Wurzel
For(D,1,E,G)
        If fPart(sqrt(C/D))=0:Goto 1
End
"Haelfte erreicht
Output(3,16,")
"Aeussere Wurzel
For(D,D,2,~G)
        If fPart(C/D^^2)=0:Goto 2
End


"Kein Faktor gefunden"
Lbl 6
Disp "Kein Faktor","gefunden.
Stop


"fpart Fehler ausgleich
Lbl 7
(iPart(Xxroot(C/D))+1->A
D->B
Goto 3
```

Tabelle 11: Programm E / Selbsterstellte Unterlage / 30.09.2015

9.1 Weitere Probleme mit der Lösungsfindung

Leider taucht auch hier ein Fehler auf. Ab $\sqrt[3]{810^3}$ bis $\sqrt[3]{2154^3}$.

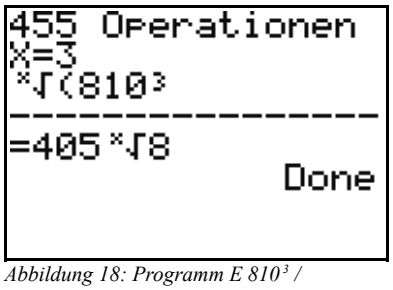


Abbildung 18: Programm E 810^3 / Bildschirmaufnahme / 22.09.2015

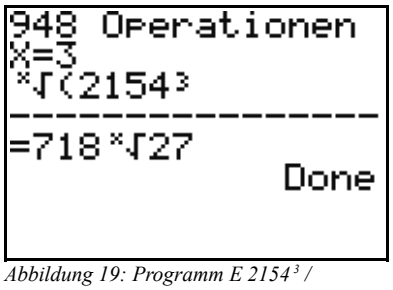


Abbildung 19: Programm E 2154^3 / Bildschirmaufnahme / 22.09.2015

In der Wurzel bleibt eine Zahl B übrig die ja weiter Faktorisiert werden kann. Deshalb wird das Programm einfach nochmal durchlaufen.

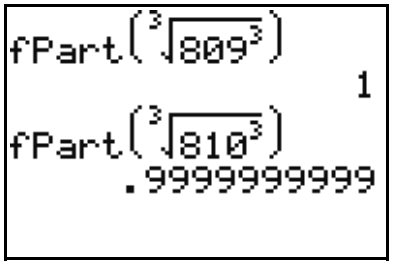


Abbildung 20: Rundungsfehler / Bildschirmaufnahme / 22.09.2015

Das liegt vermutlich daran, dass der Taschenrechner anfängt zu runden.

Mit diesen Eingaben wurde der Fehler entdeckt.

$\sqrt[3]{1709^3 \times 2}$ wird nicht berechnet.

$\sqrt[3]{1708^3 \times 2} = 427\ \sqrt[3]{128}$

$\sqrt[3]{128} = 4\ \sqrt[3]{2}$

10 Programm F (Rundungsfehler)

```
"Variablen A Wurzel aussen, B Wurzel innen, C
Eingabe, D Zaehler, E Rechenoperationen, F
Verschieben, G Gerade Ungerade, H
Wiederholen, X Hochzahl"
"Lbl 0-8"
Lbl 0
ClrHome
Disp ""
Disp "X=2"
Output(1,1,"V0.6")
Output(7,1,"2-7=X-te Wurzel")
Output(8,3,"1=X Eingeben")
Input "xroot(",C
"Initialisieren
1->A
"Wiederholen"
1->H
"2te-Wurzel"
If C>7:Goto 5
"X-Wurzel"
ClrHome
Disp ""
Output(1,1,"V0.6")
"C=1"
If C=1
Then
        Output(3,1,"xroot(")
        Input "X=",X
        Input "xroot(",C

        Else
        "C=1-7"
        C->X
        Disp "X="
        Output(2,3,X)
        Input "xroot(",C
End
2->H
"Zahl Ungerade"
1->G
If remainder(C,2)=1:2->G
"Rechenoperationen berechnen"
int((X+1)xroot(C)*3/G)->E
Output(1,1,E)
"Verschiebung
1->F
While iPart(E/(10^F))!=0
        F+1->F
End
Output(1,1+F," Operationen")
Lbl 8
"Innere Wurzel"
For(D,1,int((X+1)xroot(C)),G)
        If fPart(Xxroot(C/D))=0:Goto 1
        If fPart(Xxroot(C/D))=1:Goto 7
End
"Haelfte erreicht"
Output(3,16,")")
"Aeussere Wurzel"
For(D,D,2,~G)
        If fPart(C/D^X)=0:Goto 2
End
If A!=1:Goto 3
Goto 6


"Innere Wurzel Loesung gefunden"
Lbl 1
(Xxroot(C/D))A->A
D->B
Goto 3


"Aeussere Wurzel Loesung gefunden"
Lbl 2
C/D^X->B
AD->A


"Loesung ausgabe"
Lbl 3
"Wiederholen"
If H>1
Then
        H-1->H
        B->C
        Goto 8
End
Disp "----------------"
Disp ""
Output(5,1,"=")
Output(5,2,A)
If B=1:Goto 4
"Zahl schieben"
1->F
While iPart(A/(10^F))!=0
        F+1->F
End
Output(5,2+F,"xroot(")
Output(5,4+F,B)
Lbl 4
Stop


"Quadratwurzel berechnen"
Lbl 5
2->X
"Zahl Ungerade"
1->G
If remainder(C,2)=1:2->G
"Rechenoperationen berechnen
int(cuberoot(C)*2/G)->E
Output(1,1,E)
1->F
While iPart(E/(10^F))!=0
        F+1->F
End
Output(1,1+F," Operationen")
"Innere Wurzel
For(D,1,E,G)
        If fPart(sqrt(C/D))=0:Goto 1
End
"Haelfte erreicht
Output(3,16,")
"Aeussere Wurzel
For(D,D,2,~G)
        If fPart(C/D^^2)=0:Goto 2
End


"Kein Faktor gefunden"
Lbl 6
Disp "Kein Faktor","gefunden.
Stop


"fpart Fehler ausgleich
Lbl 7
A(iPart(Xxroot(C/D))+1->A
D->B
Goto 3
```

Tabelle 12: Programm F / Selbsterstellte Unterlage / 22.09.2015

~ (Tilde) ist im TI84 negatives Vorzeichen (nicht Minus).

A^^2	=	A2
1|E10	=	1E10
sqrt(	=	$\sqrt{}$(
cuberoot(	=	$\sqrt[3]{}$(
xroot(	=	$\sqrt[x]{}$(
->	=	→

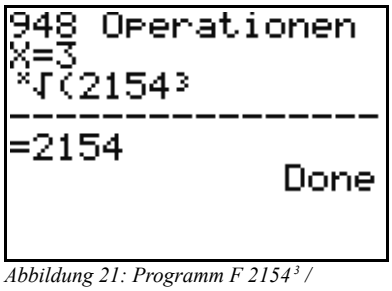


Abbildung 21: Programm F 2154^3 / Bildschirmaufnahme / 22.09.2015

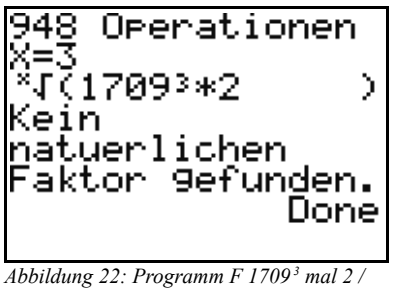


Abbildung 22: Programm F 1709³ mal 2 / Bildschirmaufnahme / 22.09.2015

Trotzallem wird bei diesem Ausdruck keine Lösung gefunden:

$\sqrt[3]{1709^3 \times 2}$

Jetzt wäre die Frage ob der alte Fehler von Programm C („*7.1 Probleme mit der Lösungsfindung auf der Seite 13*“) durch den neuen ersetzt werden kann.

Die Roten Doppelkreuze (#) zeigen das hier eine Zeile/Variable gelöscht wurde.

11 Programm G (fehlschlag)

```
"Variablen A Wurzel aussen, B Wurzel innen, C
Eingabe, D Zaehler, E Rechenoperationen, F
Verschieben, G Gerade Ungerade, H
Wiederholen, X Hochzahl"
"Lbl 0-8"
Lbl 0
ClrHome
Disp ""
Disp "X=2"
Output(1,1,"V0.7")
Output(7,1,"2-7=X-te Wurzel")
Output(8,3,"1=X Eingeben")
Input "xroot(",C
"Initialisieren
1->A
"Wiederholen"
1->H
"2te-Wurzel"
If C>7:Goto 5
"X-Wurzel"
ClrHome
Disp ""
Output(1,1,"V0.7")
"C=1"
If C=1
Then
        Output(3,1,"xroot(")
        Input "X=",X
        Input "xroot(",C

        Else
        "C=1-7"
        C->X
        Disp "X="
        Output(2,3,X)
        Input "xroot(",C
End
2->H
"Zahl Ungerade"
1->G
If remainder(C,2)=1:2->G
"Rechenoperationen berechnen"
int((X+1)xroot(C)*3/G)->E
Output(1,1,E)
```

```
"Verschiebung
1->F
While iPart(E/(10^F))!=0
        F+1->F
End
Output(1,1+F," Operationen")
Lbl 8
"Innere Wurzel"
For(D,1,int((X+1)xroot(C)),G)
        If fPart(Xxroot(C/D))=0:Goto 1
        "#
End
"Haelfte erreicht"
Output(3,16,")")
"Aeussere Wurzel"
For(D,D,2,~G)
        If fPart(C/D^X)=0:Goto 2
End
If A!=1:Goto 3
Goto 6


"Innere Wurzel Loesung gefunden"
Lbl 1
(Xxroot(C/D))A->A
D->B
Goto 3


"Aeussere Wurzel Loesung gefunden"
Lbl 2
C/D^X->B
DA->A


"Loesung ausgabe"
Lbl 3
"Wiederholen"
If H>1
Then
        H-1->H
        B->C
        Goto 8
End
Disp "----------------"
Disp ""
```

```
Output(5,1,"=")
Output(5,2,A)
If B=1:Goto 4
"Zahl schieben"
1->F
While iPart(A/(10^F))!=0
        F+1->F
End
Output(5,2+F,"xroot(")
Output(5,4+F,B)
Lbl 4
Stop


"Quadratwurzel berechnen"
Lbl 5
2->X
"Zahl Ungerade"
1->G
If remainder(C,2)=1:2->G
"Rechenoperationen berechnen
int(cuberoot(C)*2/G)->E
Output(1,1,E)
1->F
While iPart(E/(10^F))!=0
        F+1->F
End
Output(1,1+F," Operationen")
"Innere Wurzel
For(D,1,E,G)
        If fPart(sqrt(C/D))=0:Goto 1
End
"Haelfte erreicht
Output(3,16,")
"Aeussere Wurzel
For(D,D,2,~G)
        If fPart(C/D^^2)=0:Goto 2
End


"Kein Faktor gefunden"
Lbl 6
Disp "Kein Faktor","gefunden.
Stop


"#
```

Tabelle 13: Programm G / Selbsterstellte Unterlage / 22.09.2015

~ (Tilde) ist im TI84 negatives Vorzeichen (nicht Minus).

A^^2	=	A2	sqrt(	=	$\sqrt{}$(	xroot(	=	$\sqrt[X]{}$(
1|E10	=	1E10	cuberoot(	=	$\sqrt[3]{}$(	->	=	→

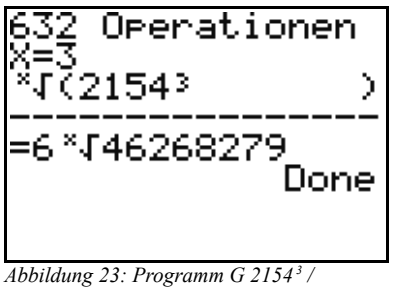


Abbildung 23: Programm G 2154^3 / Bildschirmaufnahme / 22.09.2015

Leider nicht.
Somit wir das Programm F weiterentwickelt.

11.1 Rundungsfehlerkorrektur

Der Rundungsfehler tritt bei den Zahlen $\sqrt[3]{821^3}$ bis $\sqrt[3]{2153^3}$ auf, und zwar nur bei den Primzahlen.
In diesen Bereich gibt es genau 184 Primzahlen (also zwischen 821 und 2153).

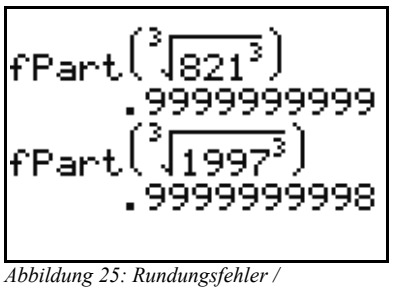


Abbildung 25: Rundungsfehler / Bildschirmaufnahme / 08.10.2017

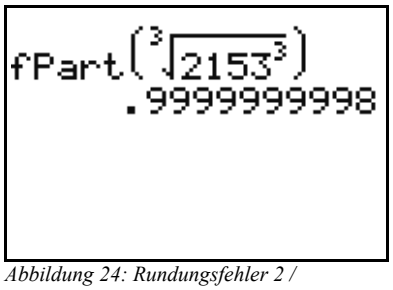


Abbildung 24: Rundungsfehler 2 / Bildschirmaufnahme / 08.10.2017

Durch den Befehl „*If fPart(Jxroot(C^J)*2/2)=1:Goto 6*“ lassen sich die Fehlberechnungen von $\sqrt[3]{811^3}$ bis $\sqrt[3]{997^3}$ vermeiden.

Durch den Befehl „*If fPart(Jxroot(C^J))>=0,9999999999:Goto 6*“ lassen sich die Fehlberechnungen von $\sqrt[3]{821^3}$ bis $\sqrt[3]{1993^3}$ vermeiden.

Durch den Befehl „*If fPart(Jxroot(C^J))>=0,9999999998:Goto 6*“ lassen sich die Fehlberechnungen von $\sqrt[3]{1997^3}$ bis $\sqrt[3]{2928^3}$ vermeiden.

$\sqrt[3]{2154^3}$ = 9.993.948.264 , jede weiterer Faktor überschreitet die 10.000.000.000 Grenze, deshalb könne wir hier aufhören. Änderung des Programms im Programm H (Blau markiert auf der Seite 28).

fpart = 1 und der Rundungsfehler haben den selben Ursprung. Der Taschenrechner rundet bei so hohen Zahlen.

Wurzel-exponent	Radikand mit höchster Potenz für diesen Wurzelexponent	Höchstwert	Höchstwert -1	Nachkommaanteil	Ergebnis
z.B.: 3	2154	2154^3	2154^3-1	$\sqrt[3]{2154^3-1} = \sqrt[3]{9.993.948.263} = 0,999\,999\,928\,2$	Kein Faktor gefunden
2	100.000	10.000.000.000	9.999.999.999	0,999995	$3\times\sqrt[3]{1.111.111.111}$
3	2.154	9.993.948.264	9.993.948.263	0,9999999282	Kein Faktor gefunden
4	316	9.971.220.736	9.971.220.735	0,9999999921	Kein Faktor gefunden
5	100	10.000.000.000	9.999.999.999	0,999999998	Kein Faktor gefunden
6	46	9.474.296.896	9.474.296.895	0,9999999992	Kein Faktor gefunden
7	26	8.031.810.176	8.031.810.175	0,9999999995	Kein Faktor gefunden
8	17	6.975.757.441	6.975.757.440	0,9999999997	Kein Faktor gefunden
9	12	5.159.780.352	5.159.780.351	0,9999999997	Kein Faktor gefunden
10	10	10.000.000.000	9.999.999.999	0,9999999999	Konnte nicht berechnet werden
11	8	8.589.934.592	8.589.934.591	0,9999999999	Konnte nicht berechnet werden
12	6	2.176.782.336	2.176.782.335	0,9999999998	Konnte nicht berechnet werden
13	5	1.220.703.125	1.220.703.124	0,9999999997	Kein Faktor gefunden
14	5	6.103.515.625	6.103.515.624	0,9999999999	Kein Faktor gefunden
15	4	1.073.741.824	1.073.741.823	0,9999999998	Konnte nicht berechnet werden
16	4	4.294.967.296	4.294.967.295	0,9999999999	Konnte nicht berechnet werden
17	3	129.140.163	129.140.162	0,9999999986	Kein Faktor gefunden
18	3	387.420.489	387.420.488	0,9999999996	Kein Faktor gefunden
19	3	1.162.261.467	1.162.261.466	0,9999999999	Kein Faktor gefunden
20	3	3.486.784.401	3.486.784.400	1	Kein Faktor gefunden
21	2	2.097.152	2.097.151	0,9999999546	Kein Faktor gefunden
22	2	4.194.304	4.194.303	0,9999999783	Kein Faktor gefunden
23	2	8.388.608	8.388.607	0,9999999896	Kein Faktor gefunden
24	2	16.777.216	16.777.215	0,999999995	Kein Faktor gefunden
25	2	33.554.432	33.554.431	0,9999999976	Kein Faktor gefunden
26	2	67.108.864	67.108.863	0,9999999989	Kein Faktor gefunden
27	2	134.217.728	134.217.727	0,9999999994	Kein Faktor gefunden
28	2	268.435.456	268.435.455	0,9999999997	Kein Faktor gefunden
29	2	536.870.912	536.870.911	0,9999999999	Konnte nicht berechnet werden
30	2	1.073.741.824	1.073.741.823	0,9999999999	Konnte nicht berechnet werden
31	2	2.147.483.648	2.147.483.647	1	Konnte nicht berechnet werden
32	2	4.294.967.296	4.294.967.295	1	Konnte nicht berechnet werden
33	2	8.589.934.592	8.589.934.591	1	Konnte nicht berechnet werden
34	1	1	0	0	

Tabelle 14: Rundungsfehler / Selbsterstellte Unterlage / 18.10.2017

Die Rot Markierten Felder werden durch das aufrunden des Rundungsfehlers (0,999 999 999 8), falsch berechnet. Diese Fehlberechnung wird durch folgende Programmierung überprüft:

```
"Berechnung pruefen
expr(Str1->C
If abs(C)!=A^IB
Then
        Disp "Konnte nicht
        Disp "Berechnet werden
        Stop
End
```

(Rosa markiert auf der Seite 28).

Mir ist aufgefallen das man die Primzahl 2 herausfiltern kann und zwar solange bis nur noch eine ungerade Zahl übrigbleibt, durch die Erkenntnis von „*8.1 Zweite Optimierung (Gerade, Ungerade Zahlen)*„ auf der Seite 15 kann man die Rechenoperationen weiter reduzieren.
Damit werden die Rechenoperationen mit der Formel: $\sqrt[X+1]{C}$ berechnet.

In unserem Beispiel $\sqrt[3]{123^2 \times 321} \approx 169$

Die Variable G (Gerade Ungerade) kann dafür ersetzt werden, da wir jetzt immer eine ungeradene Zahl aus der Wurzel ziehen.

Frage: wie oft (Variable G) kann man aus einer eingegebenen Zahl C die Primzahl 2 herausnehmen? (Theoretisch)

$$\begin{aligned} 2^G &= C \quad | \ \ln() \\ G \times \ln(2) &= \ln(C) \end{aligned}$$

$$G = \frac{\ln(C)}{\ln(2)}$$

Bei einer maximalen Eingabe von 10.000.000.000 ergibt dies: $\frac{\ln(10.000.000.000)}{\ln(2)} \approx 33$. Diese Berechnungen werden nicht in die Rechenoperationen miteinfliessen.

Wie in Kapitel „8 *Programm D (fPart=1) auf der Seite 14*“ beschrieben steigt die Rechenoperationen um 50%.
Dies lässt sich mit der folgende Programmierung vermeiden:

```
"Innere Wurzel
For(D,1,E,2)
        fPart(Ixroot(C/D))->J
        If J=0:Goto 1
        If J>=0.9999999998:Goto 6
End
```

Änderung des Programms im Programm H (Blau markiert auf der Seite 28).

Es werden auch alle unnötigen Zeichen entfernt, sowie einige Schutzfunktionen integriert.
Eine kurze Hilfestellung wird auch hinzugefügt, wie das Programm funktioniert.
Das Menü wird noch optimiert (Platzsparender programmiert).
Wir erweitern das Programm F soweit, dass er auch bei ungeradene Wurzelexponenten und negative Eingabe die Lösung zeigt, da es vermutlich schon das fertige Programm wird.

12 Programm H

```
"Variablen A Wurzel aussen, B Wurzel innen, C
Eingabe, D Zaehler, E Rechenoperationen, F
Verschieben, G Primzahl 2, H Negativ, I
Hochzahl, J Ergebnisberechnung beschleunigen
"Lbl 0-11
"String 1-3
"Komprimier Strings
"0=Beenden->Str2
" Operationen->Str3
2->I
Lbl 0
ClrHome
Disp "
Output(1,1,"V0.8
If I>0 and I<=33
Then
        Disp "
        Output(2,1,"X=
        Output(2,3,I
        Else
        Output(2,6,":Maximal 33
        Output(8,3,Str2
        Input "X=",I
        If I<0:Goto 9
        If I=0:Stop
        If I>33:Goto 8
End
Output(5,1,"3-7=X-te Wurzel
Output(6,3,"2=X Eingeben
Output(7,3,"1=Hilfe
Output(8,3,Str2
Input "xroot",Str1
expr(Str1->C
If C=0:Stop
If C=1:Goto 11
If C=2
Then
        34->I
        Goto 0
End
If C<8 and C>0
Then
        C->I
        Goto 0
End
If C>1|E10:Goto 8
"Negativabfrage
0->H
If C<0
Then
        If fPart(I)>0:Goto 9
        If fPart(I/2)!=.5:Goto 9
        1->H
        ~C->C
End
"Kommazahl eingegeben
If fPart(C)>0:Goto 10
"Initialisieren
1->A
1->B
#
#
```

```
"Primzahl 2 herausfiltern
For(G,1,iPart(ln(C)/ln(2))+1)
        If fPart(C/2^G)>0:Goto 7
End
Lbl 7
G-1->G
If G>0:C/2^G->C
"Rechenoperationen berechnen
int((I+1)xroot(C->E
"Rechenoperationen auf Display
Output(1,1,E
1->F
While iPart(E/(10^F))!=0
        F+1->F
End
Output(1,1+F,Str3
"Quadratwurzel berechnen
If I=2:Goto 4


"X-Wurzel berechnen
#
"Innere Wurzel
For(D,1,E,2)
        fPart(Ixroot(C/D))->J
        If J=0:Goto 1
        If J>=0.9999999998:Goto 6
End
"Haelfte erreicht
Output(3,16,")
"Aeussere Wurzel
For(D,D,2,~2)
        If fPart(C/D^I)=0:Goto 2
End
"2^^2*3
C->B
"Primzahl noch uebrig
If G>=I:Goto 3
Goto 5


"Innere Wurzel Loesung gefunden
Lbl 1
#(Ixroot(C/D->A
D->B
Goto 3


"Aeussere Wurzel Loesung gefunden
Lbl 2
#D->A
C/D^I->B


"Loesung ausgabe
Lbl 3
#
#
#
#
#
#
"Falls X Wurzel nur aus Primzahl 2 und unter
2^X
If G<I and A=1:Goto 5
```

```
"Maximale Rechenoperationen
ClrHome
Disp "
Output(1,1,E
Output(1,1+F,Str3
"X=
Disp "X=
Output(2,3,I
"Wurzel=
Disp "xroot
Output(3,3,Str1
"benoetigte Rechenoperationen
If A<B
Then
        E-A/2->E
        Else
        B/2->E
End
"Verschieben
1->F
While iPart(E/(10^F))!=0
        F+1->F
End
"Primzahl zurueck rechnen
While G>=I
        2A->A
        G-I->G
End
B2^G->B
"1.5xroot(44
If fPart(B)>0:Goto 5
"Berechnung pruefen
expr(Str1->C
If abs(C)!=A^IB
Then
        Disp "Konnte nicht
        Disp "Berechnet werden
        Stop
End
"Negativ richtig drehen
If H=1
Then
        If B=1
        Then
                ~A->A
                Else
                ~B->B
        End
End
"Ergebnis
Disp "----------
Output(4,17-F,int(E
Disp "
Output(5,1,"=
Output(5,3,A
"Negative Loesung auch richtig
If fPart(I/2)=0
Then
        Output(6,1,"=~
        Output(6,3,A
End
If B=1:Stop
```

```
"Zahl B schieben
1->F
While iPart(A/(10^F))!=0
        F+1->F
End
Output(5,3+F,"xroot
Output(5,5+F,B
"Negative Loesung auch richtig
If fPart(I/2)=0
Then
        Output(6,3+F,"xroot
        Output(6,5+F,B
End
Stop


"Quadratwurzel berechnen
Lbl 4
"Innere Wurzel
For(D,1,E,2)
        If fPart(sqrt(C/D))=0:Goto 1
End
"Haelfte erreicht
Output(3,16,")
"Aeussere Wurzel
For(D,D,2,~2)
        If fPart(C/D^^2)=0:Goto 2
End
"2^^2*3
C->B
"Primzahl noch uebrig
If G>=I:Goto 3
```

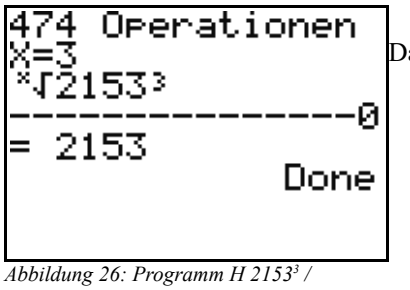
```
"Kein Faktor gefunden
Lbl 5
ClrHome
"Rechenoperationen
Disp "
Output(1,1,E
Output(1,1+F,Str3
"X=
Disp "X=
Output(2,3,I
"Wurzel=
Disp "xroot
Output(3,3,Str1
Disp "Kein Faktor
Disp "gefunden.
Stop


"fpart Fehler ausgleich
Lbl 6
#(iPart(Ixroot(C/D))+1->A
D->B
Goto 3


"Zahl zu gross
Lbl 8
Pause "Zahl zu gross.
Goto 0
```

```
"Negative Zahl
Lbl 9
Disp "Keine negative
Pause "Zahl eingeben.
Goto 0


"Keine Nachkommastellen
Lbl 10
Disp "Keine Nachkomma-
Pause "stellen eingeben
Goto 0


"Hilfe
Lbl 11
ClrHome
Disp "Dieses Programm
Disp "zieht die groes-
Disp "ste natuerliche
Disp "Zahl aus der
Disp "Wurzel z.B.:
Pause "cuberoot(192)=4*cuberoot(3)
Goto 0
```

Tabelle 15: Programm H / Selbsterstellte Unterlage / 18.10.2017

~ (Tilde) ist im TI84 negatives Vorzeichen (nicht Minus).

A^^2	=	A^2
1|E10	=	1E10
sqrt(	=	$\sqrt{}$(
xroot(	=	$\sqrt[x]{}$(
->	=	→



Dauer bis Lösungsfindung: circa 1 Sekunden.

Abbildung 26: Programm H 2153³ / Bildschirmaufnahme / 08.10.2017

Die höchste einzugeben Zahl die mit keiner Wurzelexponent gelöst werden kann ist die: 9.999.999.997.
Den seine Primfaktoren sind: 13 x 769.230.769.
Die Berechnungsdauer ist hier am höchsten, das ist die maximale Wartezeit für eine Antwort, die nicht überschritten werden kann.

Bei einer Quadratwurzelberechnung dauert es ca.33 Sekunden und
bei der Kubikwurzelberechnung beträgt die Wartezeit ca. 9 Sekunden.

13 Programm Test / Anforderung

Um dieses Programm mit anderen zu vergleichen, können folgende Aufgaben gestellt werden:

Für benötigte Rechenoperationen-überprüfung.
$\sqrt[2]{2^2 \times 3} = 2\sqrt[2]{3}$ / 0 Rechenoperation von 1 Rechenoperationen
$\sqrt[3]{251^3 \times 47} = 251\sqrt[3]{47}$ / 23 Rechenoperationen von 165 Rechenoperationen
$\sqrt[3]{251 \times 47^3} = 47\sqrt[3]{251}$ / 47 Rechenoperationen von 71 Rechenoperationen

Für fpart=1 Überprüfung.
$\sqrt[3]{44^3} = 44$

Für Rundungsfehler-überprüfung.
$\sqrt[3]{2153^3} = 2153$

Maximale Berechnungsdauer.
$\sqrt[2]{9.999.999.997}$ = keine Lösung, denn 9.999.999.997 in seine Primfaktoren zerlegt: 13 x 769.230.769.
Berechnungsdauer ca. 33 Sekunden.

$\sqrt[3]{9.999.999.997}$ = keine Lösung.
Berechnungsdauer ca. 9 Sekunden.

Speicherplatz.

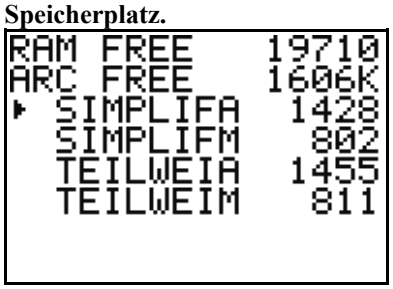


Abbildung 27: Speicherplatzverbrauch / Bildschirmaufnahme / 19.12.2017

14 Konkurrenzprogramme

Aufgabe: / Programmname:	$\sqrt[2]{123^2 \times 321} = 123\sqrt[2]{321}$	$\sqrt[2]{9.999.999.997} = \sqrt[2]{9.999.999.997}$	$\sqrt[3]{123^3 \times 321} = 123\sqrt[3]{321}$	$\sqrt[3]{44^3} = 44$	$\sqrt[3]{2.153^3} = 2.153$	$\sqrt[3]{9.999.999.997} = \sqrt[3]{9.999.999.997}$	Eingabe des höchsten Wurzelexponent	RAM [Speicherplatz]	Internetadresse:
TEILWEIA V1.0	3 Sekunden	33 Sekunden	4 Sekunden	1 Sekunde	1 Sekunde	9 Sekunden	33	1.455	http://www.ticalc.org/pub/83plus/basic/math/algebra/
TEILWEIM V1.0	4 Sekunden	52 Sekunden	4 Sekunden	1 Sekunde	1 Sekunde	9 Sekunden	33	811	
asqrt.zip	2 Sekunden	*ERR: MEMORY* 31 Sekunden	–	–	–	–	2	210	
equsolve3.zip	$3\sqrt[2]{539.601}$ 1 Sekunde	$\sqrt[2]{10.000.000.000}$ 1 Sekunde	–	–	–	–	2	609	
mathtoolsalgebra1.zip	*ERR: MEMORY* 42 Sekunden	*ERR: MEMORY* 47 Sekunden	–	–	–	–	2	9.415	
p_radical.zip	$3\sqrt[2]{539.601}$ 1 Sekunde	1 Sekunde	–	–	–	–	2	233	
radicalsimplify.zip	27 Sekunden	über 4 Minuten → abgebrochen	10 Sekunden	$22\sqrt[3]{8}$ 1 Sekunde	$\sqrt[3]{9.980.035.577}$ 28 Sekunden	28 Sekunden	3	676	
radicalv.2.zip	1 Minute	über 4 Minuten → abgebrochen	–	–	–	–	2	532	
sqrt.zip	21 Sekunden	über 4 Minuten → abgebrochen	21 Sekunden	1 Sekunde	2 Minuten und 6 Sekunden	über 4 Minuten → abgebrochen	332	797	
sqsimp.zip	über 4 Minuten → abgebrochen	über 4 Minuten → abgebrochen	–	–	–	–	2	384	
square_simplifyer.zip	$3\sqrt[2]{539.601}$ 1 Sekunde	$2\sqrt[2]{2.499.999.999}$ 1 Sekunde	–	–	–	–	2	410	
radicale.zip	1 Minuten und 2 Sekunden	über 4 Minuten → abgebrochen	24 Sekunden	1 Sekunde	1 Sekunde	$1\sqrt[3]{999.999.997}$ 72 Sekunden	9.999.999.999	546	http://www.ticalc.org/pub/83/basic/math/algebra/
REDRAT	$3\sqrt[2]{539.601}$ 1 Sekunde	1 Sekunde	–	–	–	–	2	104	https://www.youtube.com/watch?v=pydfgczpQbg
SQRT.zip (Primzahlliste)	4 Sekunden	*ERR: SYNTAX* 16 Sekunden	–	–	–	–	2	589 (1.516)	https://www.cemetech.net/programs/index.php?mode=folder&path=/83plus/basic/math/

Hintergrundfarbe
Bester / Schnellster in dieser Kategorie / Aufgabe
Ok
Dauer über 33 Sekunden
Teilberechnung oder Zeitüberschreitung

Tabelle 16: Berechnungsdauer der Konkurrenzprogramme / Selbsterstellte Unterlage / 18.10.2017

Das Programm „radicale.zip“ hat als Vorteil, bis zum 9.999.999.999 Wurzelexponent Aufzulösen. Wenn man aber bedenkt das $2^{33} = 8.589.934.592$ und $2^{34} = 17.179.869.184$, übersteigt 2^{34} die Genauigkeit des Taschenrechners. Die 2 ist der kleinste Faktor der gezogen werden kann. Es macht also kein Sinn höher als die 33 Wurzelpotenz zu gehen, denn wir haben nur 10 Stellen zur Verfügung.

15 Editionen

Desweiteren erzeugen wir verschieden Programme, für die jeweiligen Ansprüche.
z.B.:

- **Absolut,** (TEILWEI**A**)
- **Mini**, alle unnötige Elemente entfernt z.B. schnellere Quadratwurzelberechnung, Operationen anzeige, benötigte Operationen, Hilfe. (TEILWEI**M**)
- Englische Edition **Absolute**. (SIMPLIF**A**)
- Englische Edition **Mini**. (SIMPLIF**M**)

Die hier gezeigten Edition wurden im Gegensatz zu Programm H platzsparend programmiert, da mir hinterher immer wieder neue Ideen kamen, wie ich das Programm kleiner schreiben konnte.

Die Kommentare sind mit der roten Schrift gegenzeichnet.
Die grüne Schrift zeigt die Schutzfunktionen des Programms (Falscheingabe durch den Anwender).

Kommentar

```
"Variablen A Wurzel aussen, B Wurzel innen, C Eingabe, D Zaehler, E Rechenoperationen, F Verschieben, G Primzahl 2, H Negativ, I Hochzahl, J Ergebnisberechnung beschleunigen
"Lbl 0-11
"String 1-3
"Komprimier Strings
"0=Beenden->Str2
"Operationen_->Str3
2->I
Lbl 0
ClrHome
Disp "V1.0 Kommentar
If I>0 and I<=33
Then
        Disp "X=
        Output(2,3,I
Else
        Output(2,6,":Maximal 33
        Output(8,3,Str2
        Input "X=",I
        If I<0:Goto 9
        If I=0:Stop
        If I>33:Goto 8
End
Output(5,1,"3-7=X-te Wurzel
Output(6,3,"2=X Eingeben
Output(7,3,"1=Hilfe
Output(8,3,Str2
Input "xroot",Str1
expr(Str1->C
If C=0:Stop
If C=1:Goto 11

If C=2
Then
        34->I
        Goto 0
End

If C<8 and C>0
Then
        C->I
        Goto 0
End

"Negativabfrage
0->H
```

Absolut

```
"0=Beenden->Str1

2->I
Lbl 0
ClrHome
Disp "V1.0 Absolut
If I>0 and I<=33
Then
        Disp "X=
        Output(2,3,I
Else
        Output(2,6,":Maximal 33
        Output(8,3,Str1
        Input "X=",I
        If I<0:Goto 9
        If I=0:Stop
        If I>33:Goto 8
End
Output(5,1,"3-7=X-te Wurzel
Output(6,3,"2=X Eingeben
Output(7,3,"1=Hilfe
Output(8,3,Str1
Input "xroot",C
C->K
If C=0:Stop
If C=1
Then
        ClrHome
        Disp "Dieses Programm
        Disp "zieht die groes-
        Disp "ste natuerliche
        Disp "Zahl aus der
        Disp "Wurzel z.B.:
        Pause "cuberoot(192)=4*cuberoot(3)
        Goto 0
End

If C=2
Then
        34->I
        Goto 0
End

If C<8 and C>0
Then
        C->I
        Goto 0
End

0->H
```

Mini

```
2->I
Lbl 0
ClrHome
Disp "V1.0 Mini

Disp "X=
Output(2,3,I

Output(7,1,"2-33=X-te Wurzel

Output(8,4,"0=Beenden
Input "xroot",C
C->K
If C=0:Stop

If C<34 and C>0
Then
        C->I
        Goto 0
End

0->H
```

```
If C<0
Then
	If fPart(I)>0:Goto 9
	If fPart(I/2)!=.5:Goto 9
	1->H
	~C->C
End
If C>1|E10:Goto 8
"Kommazahl eingegeben
If fPart(C)>0:Goto 10




"Initialisieren
1->A
1->B
"Primzahl 2 herausfiltern
For(G,1,iPart(ln(C)/ln(2)))
	If fPart(C/2^G)>0:Goto 7
End
Lbl 7
G-1->G
If G>0:C/2^G->C


"Rechenoperationen berechnen
int((I+1)xroot(C->E
Output(1,1,Str3
"Rechenoperationen auf Display
Output(1,13,E
"Quadratwurzel berechnen
If I=2:Goto 4
"X-Wurzel berechnen
"Innere Wurzel
For(D,1,E,2)
	fPart(Ixroot(C/D))->J
	If J=0:Goto 1
	If J>=0.9999999998:Goto 6
End




"Haelfte erreicht
Output(3,16,")
"Aeussere Wurzel
For(D,D,2,~2)
	If fPart(C/D^I)=0:Goto 2
End

"2^^2*3
C->B
"Primzahl noch uebrig
If G>=I:Goto 3
Goto 5


"Innere Wurzel Loesung gefunden
Lbl 1
(Ixroot(C/D->A
D->B
Goto 3


"Aeussere Wurzel Loesung gefunden
Lbl 2
D->A
C/D^I->B


"Loesung ausgabe
Lbl 3
"Falls X Wurzel nur aus Primzahl 2 und unter 2^X
If G<I and A=1:Goto 5
"Maximale Rechenoperationen
ClrHome
Disp "
Output(1,1,Str3
Output(1,13,E
"X=
Disp "X=
Output(2,3,I
"Wurzel=
Disp "xroot
Output(3,3,Str1
"benoetigte Rechenoperationen
If A<B
Then
	E-A/2->E
Else
	B/2->E
End
"Verschieben
1->F
```

```
If C<0
Then
	If fPart(I)>0 or fPart(I/2)!=.5:Goto 9

	1->H
	~C->C
End
If C>10^(10:Goto 8

If fPart(C)>0
Then
	Disp "Keine Nachkomma-
	Pause "stellen eingeben
	Goto 0
End

1->A
1->B

For(G,1,iPart(ln(C)/ln(2)))
	If fPart(C/2^G)>0:Goto 7
End
Lbl 7
G-1->G
If G>0:C/2^G->C


int((I+1)xroot(C->E
Output(1,1,"Operationen_

Output(1,13,E

If I=2:Goto 4


For(D,1,E,2)
	fPart(Ixroot(C/D))->J
	If J=0:Goto 1
	If J>=0.9999999998
	Then
		(iPart(Ixroot(C/D))+1->A
		D->B
		Goto 3
	End
End

Output(3,16,")

For(D,D,2,~2)
	If fPart(C/D^I)=0:Goto 2
End


C->B

If G>=I:Goto 3
Goto 5


Lbl 1
(Ixroot(C/D->A
D->B
Goto 3


Lbl 2
D->A
C/D^I->B


Lbl 3

If G<I and A=1:Goto 5




If A<B
Then
	E-A/2->E
Else
	B/2->E
End

1->F
```

```
If C<0
Then
	If fPart(I)>0 or fPart(I/2)!=.5:Goto 0

	1->H
	~C->C
End
If C>10^(10
Then
	Pause "Zahl zu gross.
	Goto 0
End
If fPart(C)>0:Goto 0


1->A
1->B

For(G,1,iPart(ln(C)/ln(2)))
	If fPart(C/2^G)>0:Goto 7
End
Lbl 7
G-1->G
If G>0:C/2^G->C


int((I+1)xroot(C->E




For(D,1,E,2)
	fPart(Ixroot(C/D))->J
	If J=0:Goto 1
	If J>=0.9999999998
	Then
		(iPart(Ixroot(C/D))+1->A
		D->B
		Goto 3
	End
End


For(D,D,2,~2)
	If fPart(C/D^I)=0:Goto 2
End


C->B

If G>=I:Goto 3
Goto 5


Lbl 1
(Ixroot(C/D->A
D->B
Goto 3


Lbl 2
D->A
C/D^I->B


Lbl 3

If G<I and A=1 :Goto 5
```

```
While iPart(E/(10^F))!=0
	F+1->F
End


"Primzahl zurueck rechnen
While G>=I
	2A->A
	G-I->G
End
B2^G->B
"1.5xroot(44
If fPart(B)>0:Goto 5
"Negativ richtig drehen
If H=1
Then
	If B=1
	Then
		~A->A
	Else
		~B->B
	End
End



"Berechnung pruefen
If expr(Str1)!=A^IB
Then
	Disp "Konnte nicht
	Disp "Berechnet werden
	Stop
End



"Ergebnis
Disp "----------------
Output(4,17-F,int(E
Disp "
Output(5,1,"=
Output(5,3,A
"Negative Loesung auch richtig
If fPart(I/2)=0
Then
	Output(6,1,"=~
	Output(6,3,A
End
If B=1:Stop
"Zahl B schieben
1->F
While iPart(A/(10^F))!=0
	F+1->F
End
Output(5,3+F,"xroot
Output(5,5+F,B



"Negative Loesung auch richtig
If fPart(I/2)=0
Then
	Output(6,3+F,"xroot
	Output(6,5+F,B
End
Stop



"Quadratwurzel berechnen
Lbl 4
"Innere Wurzel
For(D,1,E,2)
	If fPart(sqrt(C/D))=0:Goto 1
End
"Haelfte erreicht
Output(3,16,")
"Aeussere Wurzel
For(D,D,2,~2)
	If fPart(C/D^^2)=0:Goto 2
End
"2^^2*3
C->B
"Primzahl noch uebrig
If G>=I:Goto 3
"Kein Faktor gefunden
Lbl 5
ClrHome
"Rechenoperationen
Disp "
Output(1,1,Str3
Output(1,13,E
"X=
Disp "X=
Output(2,3,I
"Wurzel=
Disp "xroot
```

```
While iPart(E/(10^F))!=0
	F+1->F
End


While G>=I
	2A->A
	G-I->G
End
B2^G->B

If fPart(B)>0:Goto 5

If H=1
Then
	If B=1
	Then
		~A->A
	Else
		~B->B
	End
End



If K!=A^IB
Then
	Disp "Konnte nicht
	Disp "Berechnet werden
	Stop
End



Disp "----------------
Output(4,17-F,int(E
Disp "
Output(5,1,"=
Output(5,3,A

If fPart(I/2)=0
Then
	Output(6,1,"=~
	Output(6,3,A
End
If B=1:Stop

1->F
While iPart(A/(10^F))!=0
	F+1->F
End
Output(5,3+F,"xroot
Output(5,5+F,B



If fPart(I/2)=0
Then
	Output(6,3+F,"xroot
	Output(6,5+F,B
End
Stop



Lbl 4

For(D,1,E,2)
	If fPart(sqrt(C/D))=0:Goto 1
End

Output(3,16,")

For(D,D,2,~2)
	If fPart(C/D^^2)=0:Goto 2
End

C->B

If G>=I:Goto 3

Lbl 5
```

```
While G>=I
	2A->A
	G-I->G
End
B2^G->B

If fPart(B)>0:Goto 5

If H=1
Then
	If B=1
	Then
		~A->A
	Else
		~B->B
	End
End

If K!=A^IB
Then
	Disp "Konnte nicht
	Disp "Berechnet werden
	Stop
End

Disp "----------------

Disp "
Output(5,1,"=
Output(5,3,A

If fPart(I/2)=0
Then
	Output(6,1,"=~
	Output(6,3,A
End
If B=1:Stop

1->F
While iPart(A/(10^F))!=0
	F+1->F
End
Output(5,3+F,"xroot
Output(5,5+F,B

If fPart(I/2)=0
Then
	Output(6,3+F,"xroot
	Output(6,5+F,B
End
Stop

Lbl 5
```

Output(3,3,Str1 Disp "Kein Faktor Disp "gefunden. Stop	Disp "Kein Faktor Disp "gefunden. Stop	Disp "Kein Faktor Disp "gefunden. Stop
"fpart Fehler ausgleich Lbl 6 (iPart(Ixroot(C/D))+1->A D->B Goto 3		
"Zahl zu gross Lbl 8 Pause "Zahl zu gross. Goto 0	Lbl 8 Pause "Zahl zu gross. Goto 0	
"Negative Zahl Lbl 9 Disp "Keine negative Pause "Zahl eingeben. Goto 0	Lbl 9 Disp "Keine negative Pause "Zahl eingeben. Goto 0	
"Keine Nachkommastellen Lbl 10 Disp "Keine Nachkomma- Pause "stellen eingeben Goto 0		
"Hilfe Lbl 11 ClrHome Disp "Dieses Programm Disp "zieht die groes- Disp "ste natuerliche Disp "Zahl aus der Disp "Wurzel z.B.: Pause "cuberoot(192)=4*cuberoot(3) Goto 0		

Tabelle 17: Programm Editionen / Selbsterstellte Unterlage / 18.10.2017

~ (Tilde) ist im TI84 negatives Vorzeichen (nicht Minus).

A^^2	=	A2	sqrt(	=	$\sqrt{}$(	xroot(	=	$\sqrt[X]{}$(
1|E10	=	1E10	cuberoot(	=	$\sqrt[3]{}$(	->	=	→
_	=	Leerzeichen	10^(	=	$10^{}$(			

16 Abbildungsverzeichnis

17 Tabellenverzeichnis