

Programmieren in der 4. Klasse

Themenbereich	
Geometrie	
Inhalte	Ziele
<ul style="list-style-type: none"> • Oberfläche und Volumen verschiedener Körper; • Arbeiten mit Formeln; • Einfache Programme mit dem Ti-92; 	<ul style="list-style-type: none"> • Zahlen in Formeln einsetzen können • Die Größe einer Formel berechnen können, wenn die anderen Größen bekannt sind. • Einfache Programm zur Berechnung von Volumen und Oberfläche verschiedener Körper erstellen können.
<p>Adressaten:</p> <p>Die folgenden Seiten dienen als Vorschlag, wie in Klassen, in denen der Ti-92 bereits in der 3. bzw. 4. Klasse zum Einsatz kommt, das Programmieren des Ti-92 eingeführt und erklärt werden kann. Diese Sequenz orientiert sich am Skriptum "Funktionen und Programme am Ti-92" von Dr. Thomas Himmelbauer anlässlich eines T³-Seminars in St. Pölten am 23. April 1998.</p> <p>Es handelt sich bei den folgenden Seiten um Anleitungen für den Lehrer, wie man ausgehend von sehr einfachen Funktionen und Prozeduren, den Umfang und damit die Benutzerfreundlichkeit der Programme langsam ausbauen kann.</p>	

Voraussetzungen

Mathematische Voraussetzungen

Die Schüler sollten bereits die Formeln zur Berechnung von Volumen und Oberfläche von Drehkegel, Drehzylinder, Prisma und Pyramide kennen und Zahlen in diese Formeln einsetzen können.

TI-92 Voraussetzungen

Die Schüler sollten bereits folgende Grundkenntnisse im Rechnen mit dem TI-92 besitzen: Grundrechnungsarten (inklusive Klammern), Brüche, Rechnen mit Variablen, Abspeichern von Zahlen als Variable mittels Sto.

Ti-92 Erweiterung

Im Laufe der Unterrichtssequenz erlernen die Schüler das Erstellen einfacher Programme. Die nötigen Befehle werden an den entsprechenden Stellen erläutert.

Unterrichtsmethode

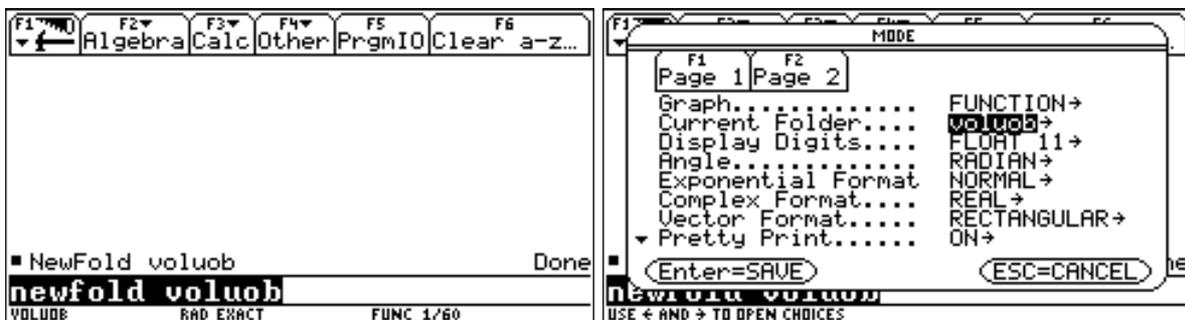
Die Vorgehensweise im Unterricht ist dem Lehrer überlassen. Die folgenden Seiten sind nur als Anleitung für den Lehrer gedacht, nicht aber als Arbeitsblätter für die Schüler. Eventuell benötigte Arbeitsblätter sind selbst zu erstellen, Zusammenfassungen der wichtigsten Befehle können an der Tafel aufgeschrieben bzw. ins Heft diktiert werden.

Volums- und Oberflächenberechnung des Drehkegels

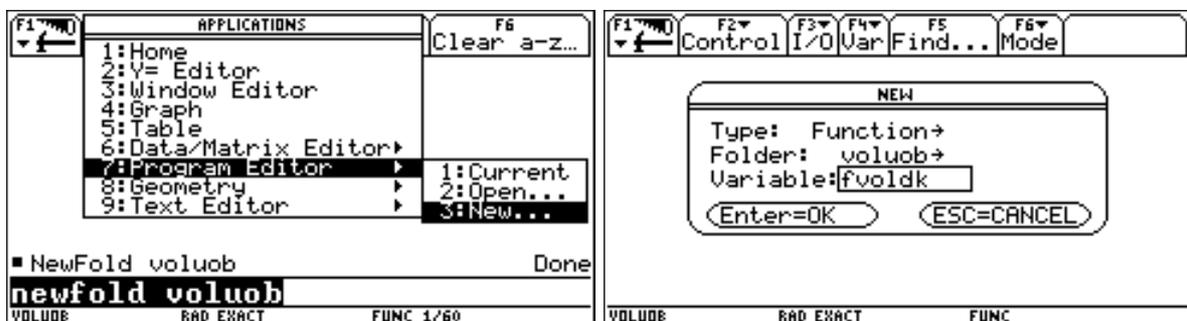
Funktion zur Berechnung des Volumens

Zunächst soll eine Funktion erstellt werden, die das Volumen eines Drehkegels, von dem Radius und Höhe gegeben sind, ausgibt.

Sollen alle Volums- und Oberflächenberechnungen in einem gemeinsamen Folder (z.B. voluob für VOLumen Und OBERfläche) gespeichert werden, so muss dieser Folder zunächst mit newfold erstellt und dann im Mode-Menü zum aktuellen Folder gemacht werden. Sollen bei den Ergebnissen ebenso wie bei den folgenden Seiten immer Dezimalzahlen ausgegeben werden, so sollte auf der zweiten Seite des Mode-Menüs Approximate eingestellt werden.



Nun wählt man im Programmeditor New. Dann stellt man den Typ auf Function um, achtet auf die Einstellung, des richtigen Folders und gibt der Function einen Namen.

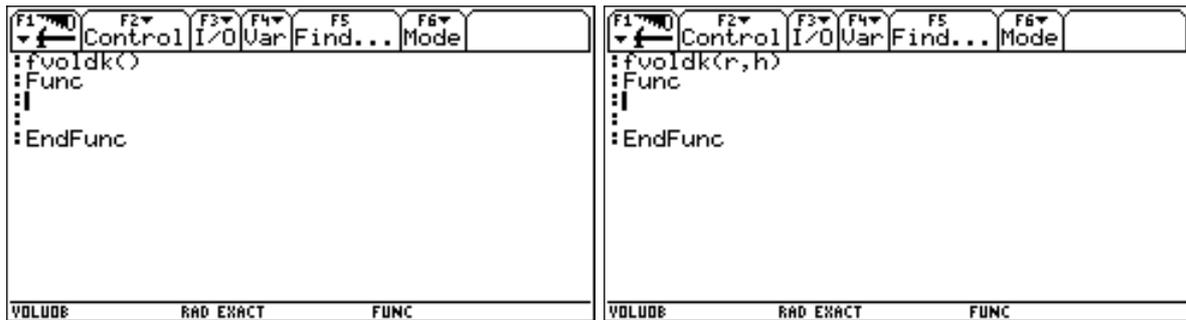


Es erscheint das Fenster zum Bearbeiten von Funktionen bzw. Programmen. Ganz oben steht der Name der Function fvoldk(), dann der Funktionsanfang Func und das Funktionsende EndFunc.

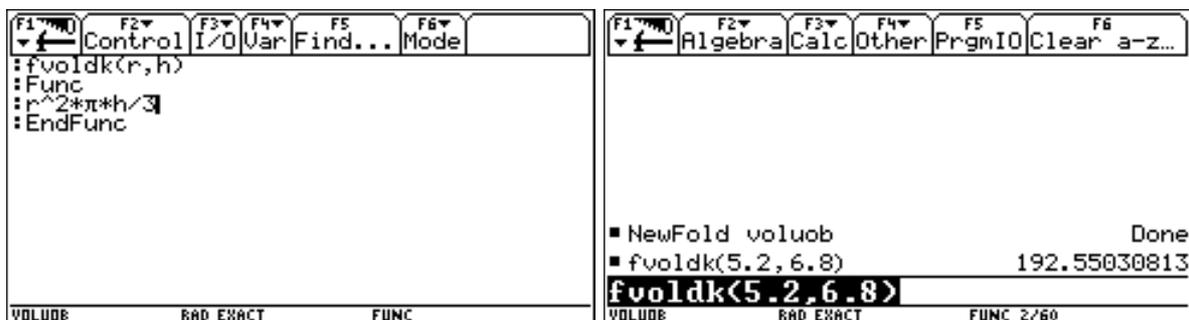
Im Bereich zwischen Func und EndFunc müssen die von der Function durchzuführenden Befehle eingetragen werden. Jeder Befehl muss mit einem Doppelpunkt beginnen.

Die einzige Eingabemöglichkeit für Funktionen sind Parameter, die innerhalb der runden Klammern am Ende des Funktionsnamen angeführt werden. Sie werden durch ihre Reihenfolge voneinander unterschieden. Wir wählen als Eingabevariable die beiden gegebenen Größen r und h . Beim Aufruf der Funktion muss die gewählte Reihenfolge der Variablen eingehalten werden.

z.B. führt der Aufruf $fvoidk(5.2,6.8)$ zur Berechnung des Volumens eines Drehkegels mit $r=5,2$ LE und $h=6,8$ LE.



Nun ist nur mehr der Befehl zur Berechnung einzugeben. Um die Function zu testen, wechselt man in den Homebildschirm und ruft die Function auf. Als Antwort erhält man das Ergebnis der letzten Zeile der Function. (In unserem Fall besteht die Function eigentlich nur aus einer Zeile)



Funktion zur Berechnung der Oberfläche

Auf ähnliche Weise kann nun eine neue Function zur Berechnung der Oberfläche des Drehkegels erstellt werden:

Soll das Ergebnis nicht in einem Rechenschritt ermittelt werden, so muss zunächst s berechnet und abgespeichert werden. Weil eine Function nur mit lokalen Variablen arbeiten kann, muss s zunächst als local bekanntgegeben werden.

```

F1 Control F2 I/O F3 Var F4 Find... F5 Mode F6
:fobdk(r,h)
:Func
:Local s
:√(r^2+h^2)→s
:r^2*π+r*π*s
:EndFunc
VOLUOB RAD EXACT FUNC

F1 Algebra F2 Calc F3 Other F4 PrgmIO F5 Clear a-z... F6
■ NewFold voluob Done
■ fvoidk(5.2, 6.8) 192.55030813
■ fobdk(5.2, 6.8) 224.79334444
fobdk(5.2, 6.8)
VOLUOB RAD EXACT FUNC 3/60

```

Programm zur Berechnung von Oberfläche und Volumen

Nun soll das gleiche Problem mit Hilfe eines Programms gelöst werden. Das hat den Vorteil, dass eine Vielzahl von Anweisungen zur Verfügung stehen, die praktisch alle Bereiche des Rechners steuern können. So wird es uns möglich sein, in einem Programm sowohl Volumen als auch Oberfläche zu berechnen und mit Text versehen auszugeben. Der Nachteil von Programmen liegt aber darin, dass die Ausgabe nicht im Homebildschirm erfolgt, sondern im Input-Output-Schirm (I/O-Schirm).

Wir wählen im Programmreditor New, lassen den Typ Program und legen einen Programmnamen fest. Es erscheint eine zu den Funktionen analoge Struktur.

```

F1 Control F2 I/O F3 Var F4 Find... F5 Mode F6
NEW
Type: Program
Folder: voluob
Variable: pvolobdk
Enter=OK ESC=CANCEL
VOLUOB RAD EXACT FUNC

F1 Control F2 I/O F3 Var F4 Find... F5 Mode F6
:pvolobdk()
:Prgm
:
:
:EndPrgm
VOLUOB RAD EXACT FUNC

```

Wir geben wieder die beiden Parameter r und h an und fügen die Zeilen zur Berechnung von s , O und V ein. Da Programme auch mit globalen Variablen arbeiten, müssten diese Variablen nicht als local definiert werden. Der Vorteil der Verwendung von lokalen Variablen liegt aber darin, dass diese Variablen nach Beendigung des Programms nicht belegt sind - was ja bei manchen Spielen leider der Fall ist.

Bei Programmen erfolgt die Ausgabe aber nicht automatisch, sondern es müssen dem Programm die entsprechenden Befehle erteilt werden. Die einfachste Art der Ausgabe ist der Befehl disp. Wenn früher ausgegebene Inhalte nicht erwünscht sind, muss der I/O-Schirm mit ClrIO gelöscht werden.

F1	F2	F3	F4	F5	F6
Control	I/O	Var	Find...	Mode	
<pre> :pvlobdk(r,h) :Prgm :local s,o,v :√(r^2+h^2)→s :r^2*π+r*π*s→o :r^2*π*h/3→v : :EndPrgm </pre>					
VOLUDB		RAD EXACT		FUNC	

F1	F2	F3	F4	F5	F6
Control	I/O	Var	Find...	Mode	
<pre> :pvlobdk(r,h) :Prgm :Local s,o,v :√(r^2+h^2)→s :r^2*π+r*π*s→o :r^2*π*h/3→v :ClrIO :Disp "Oberfläche: " :Disp o :Disp "Volumen:" :Disp v :EndPrgm </pre>					
VOLUDB		RAD APPROX		FUNC	

Nun können wir unser Programm testen. Wir wechseln in den Homebildschirm und rufen das Programm mit den beiden Parametern auf. Der Rechner wechselt in den I/O-Schirm und zeigt die Lösungen an. Über F5 kommt man wieder in den Homebildschirm zurück.

F1	F2	F3	F4	F5	F6
Algebra	Calc	Other	PrgmIO	Clear	a-z...
<pre> Oberfläche: 224.79334444 Volumen: 192.55030813 </pre>					
pvlobdk(5.2,6.8)					
VOLUDB		RAD APPROX		FUNC 0/60	

F1	F2	F3	F4	F5	F6
Algebra	Calc	Other	PrgmIO	Clear	a-z...
<pre> Oberfläche: 224.79334444 Volumen: 192.55030813 </pre>					
VOLUDB		RAD APPROX		FUNC 1/60	

Nun soll das Programm noch etwas bedienerfreundlicher gemacht werden. Die Eingabe der Variablen über Parameter ist insofern kompliziert, da man sich die Reihenfolge der Parameter merken muss. Einfacher ist es, wenn die Variablen erst, während das Programm läuft, abgefragt werden. Dazu steht der Befehl Prompt zur Verfügung.

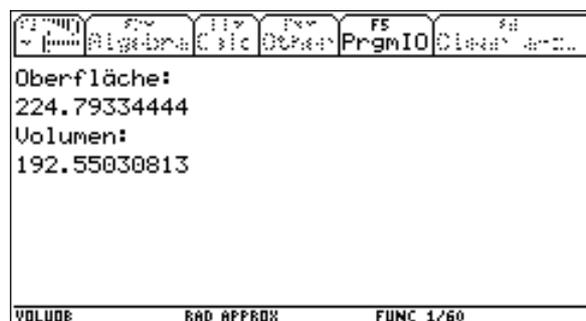
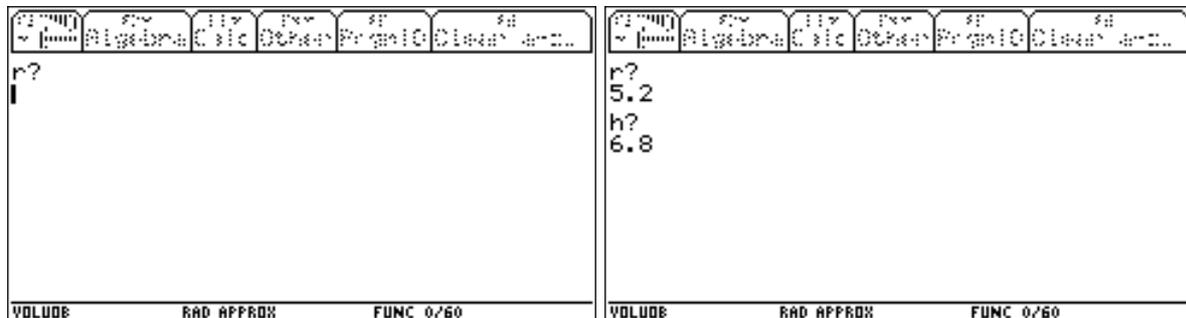
Wir löschen also die Eingabeparameter. Die runden Klammern müssen aber erhalten bleiben. Nach einem Löschen des I/O-Schirmes rufen wir über Prompt die einzugebenden Variablen auf. Wurden diese vorher als local festgelegt, so handelt es sich um lokale Variable, andernfalls um globale.

Im Homebildschirm rufen wir nun das Programm mit runden Klammer auf.

F1	F2	F3	F4	F5	F6
Control	I/O	Var	Find...	Mode	
<pre> :pvlobdk() :Prgm :Local s,o,v,r,h :ClrIO :Prompt r,h :√(r^2+h^2)→s :r^2*π+r*π*s→o :r^2*π*h/3→v :ClrIO :Disp "Oberfläche: " :Disp o :Disp "Volumen:" </pre>					
VOLUDB		RAD APPROX		FUNC	

F1	F2	F3	F4	F5	F6
Algebra	Calc	Other	PrgmIO	Clear	a-z...
<pre> </pre>					
pvlobdk()					
VOLUDB		RAD APPROX		FUNC 0/60	

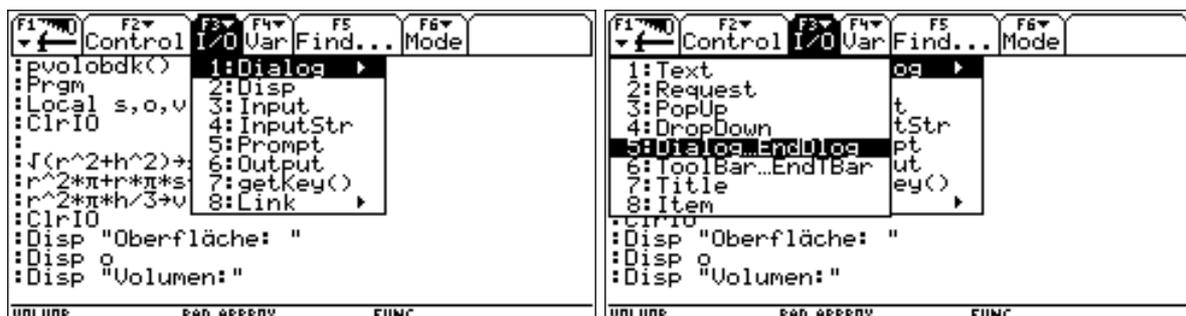
Es meldet sich der I/O-Schirm mit der im Promptbefehl zuerst angeführten Variablen r und einem Fragezeichen dahinter. Der Cursor steht eine Zeile tiefer. Nach Eingabe eines Wertes für r und bestätigen mit Enter verlangt der Rechner die Eingabe von h . Anschließend wird das Ergebnis wie schon bekannt ausgegeben.



Damit ist dieses Programm eigentlich funktionsfähig. In ähnlicher Weise können auch Programme für weitere Körper, aber auch Programme zur Berechnung von Flächeninhalt und eventuell Umfang verschiedener Vierecke (die bereits in der 3. Klasse behandelt wurden) erstellt werden. Wie man diese Programme zu einem eigenen Menü zusammenfassen kann wird weiter hinten erläutert.

Zuvor sollen aber noch einige Erweiterungsmöglichkeiten, des obigen Programms gezeigt werden. Diese sind im Unterricht zwar nicht unbedingt nötig, könnten in einer Supplierstunde aber als Anregung für die Schüler dienen, ihre eigenen Programme noch "schöner" und benutzerfreundlicher zu gestalten.

Zunächst einmal soll die Eingabe der beiden Variablen nicht über den Befehl Prompt erfolgen, sondern mittels Dialog. Wir entfernen den Befehl Prompt r,h wieder und fügen an dieser Stelle die Dialogbox für die Eingabe ein. Die entsprechenden Befehle befinden sich unter F3/Dialog.



```

F1 Control F2 I/O F3 Var F4 Find... F5 Mode F6
:pvlobdk()
:Prgm
:Local s,o,v,r,h
:ClrIO
:Dialog
:
:EndDlog
:√(r^2+h^2)→s
:r^2*π+r*π*s→o
:r^2*π*h/3→v
:ClrIO
:Disp "Oberfläche: "
VOLUDB RAD APPROX FUNC

```

Danach fügen wir den Befehl Title ein, um die Überschrift der Dialogbox festzulegen.

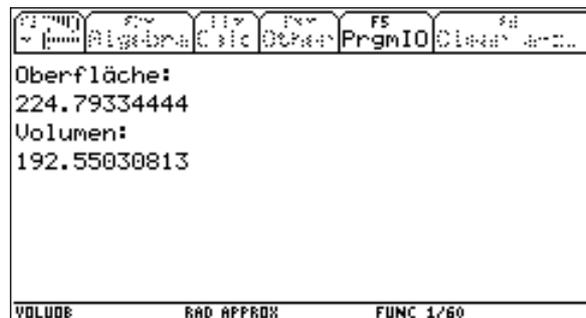
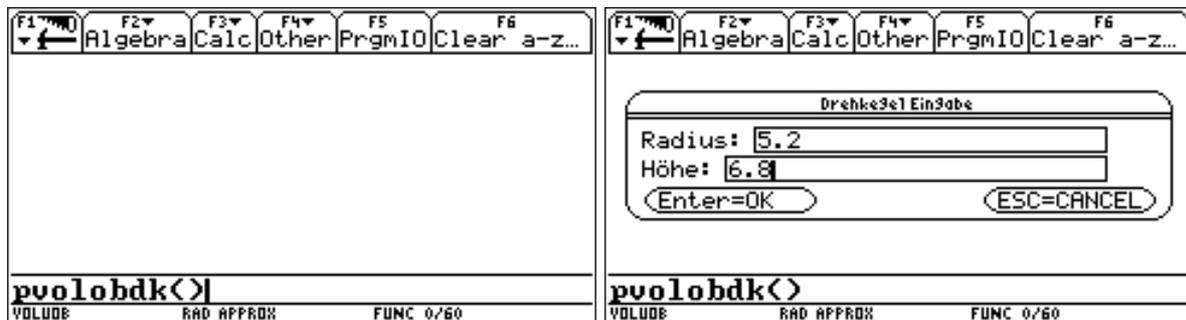
<pre> F1 Control F2 I/O F3 Var F4 Find... F5 Mode F6 1:Text 2:Request 3:PopUp 4:DropDown 5:Dialog...EndDlog 6:ToolBar...EndTBar 7:Title 8:Item :r^2*π+r*π*s→o :r^2*π*h/3→v :ClrIO :Disp "Oberfläche: " VOLUDB RAD APPROX FUNC </pre>	<pre> F1 Control F2 I/O F3 Var F4 Find... F5 Mode F6 :pvlobdk() :Prgm :Local s,o,v,r,h :ClrIO :Dialog :Title "Drehkegel Eingabe" :EndDlog :√(r^2+h^2)→s :r^2*π+r*π*s→o :r^2*π*h/3→v :ClrIO :Disp "Oberfläche: " VOLUDB RAD APPROX FUNC </pre>
--	---

Der Befehl, der eine Eingabe fordert, lautet Request, ihm folgt ein Text und die Variable, unter der die Eingabe gespeichert wird. Dieser Befehl könnte wieder über F3 geholt werden, es ist aber natürlich auch möglich, ihn über die Tastatur einzugeben. Wichtig ist nur diesen Befehl noch vor EndDlog einzugeben.

Da die über Request eingelesenen Werte als Text abgespeichert werden, müssen diese nach dem Dialog-Ende in eine Zahl verwandelt werden. Dazu steht uns der Befehl expr zur Verfügung.

<pre> F1 Control F2 I/O F3 Var F4 Find... F5 Mode F6 :pvlobdk() :Prgm :Local s,o,v,r,h :Dialog :Title "Drehkegel Eingabe" :Request "Radius",r :Request "Höhe",h :EndDlog :√(r^2+h^2)→s :r^2*π+r*π*s→o :r^2*π*h/3→v :ClrIO VOLUDB RAD APPROX FUNC </pre>	<pre> F1 Control F2 I/O F3 Var F4 Find... F5 Mode F6 :pvlobdk() :Prgm :Local s,o,v,r,h :Dialog :Title "Drehkegel Eingabe" :Request "Radius",r :Request "Höhe",h :EndDlog :expr(r)→r :expr(h)→h :√(r^2+h^2)→s :r^2*π+r*π*s→o VOLUDB RAD APPROX FUNC </pre>
---	---

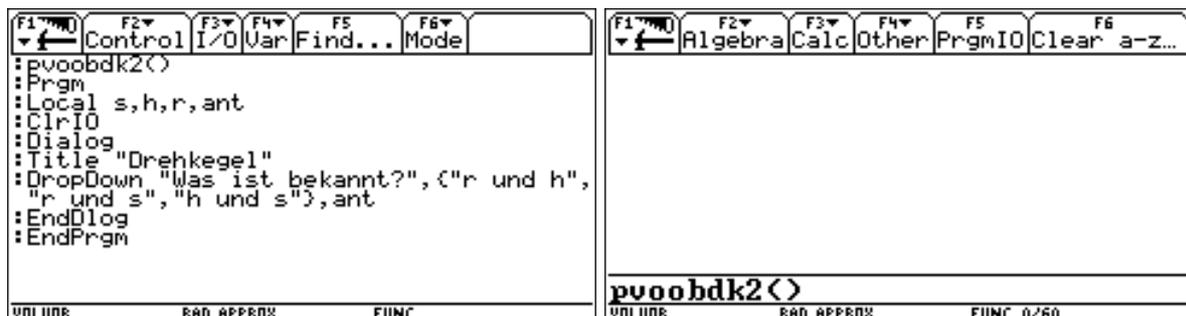
Nun können wir unser Programm aus dem Homebildschirm wieder starten. Es erscheint zunächst eine Dialogbox, in der wir die Werte für r und h eingeben, anschließend werden die beiden Ergebnisse wieder in bereits bekannter Art ausgegeben.

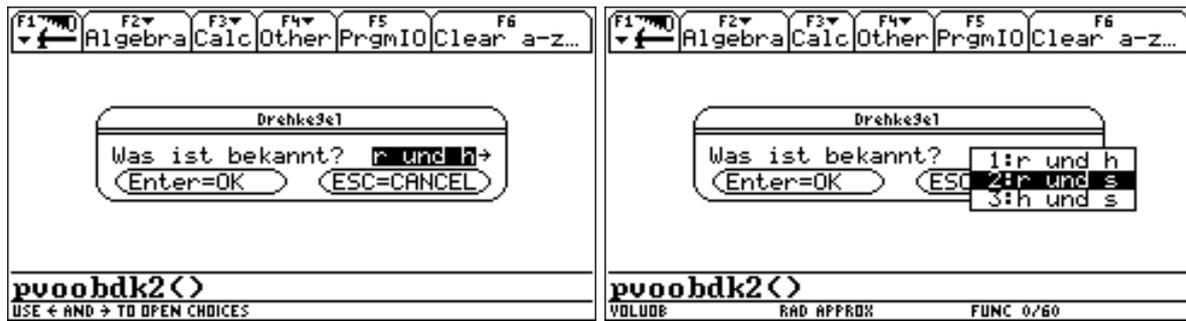


Programm zur Auswahl der gegebenen Größen und Berechnung von Oberfläche und Volumen

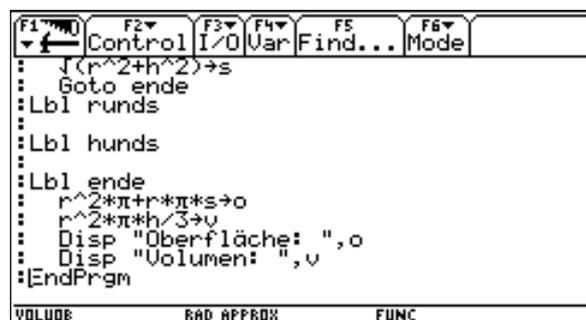
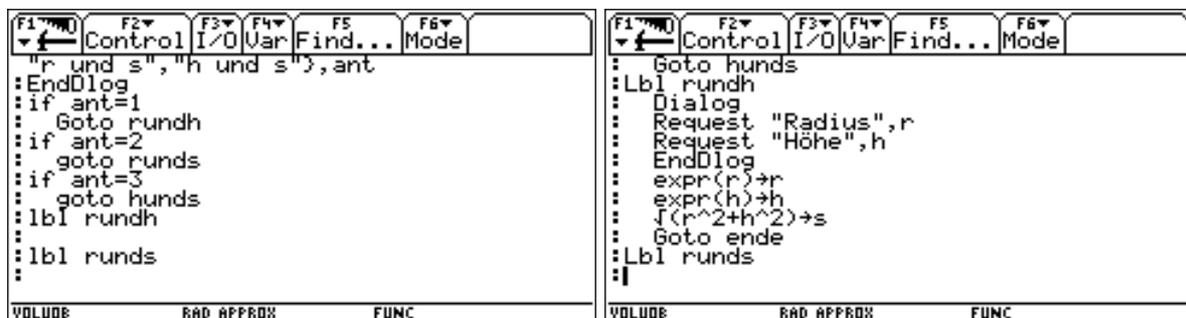
Da nicht immer r und h gegeben sind, kann nun noch ein etwas komplizierteres Programm mit einer Abfrage nach den gegebenen Größen und davon abhängiger Berechnung erstellt werden. Im folgenden werden nur mehr die noch nicht bekannten Befehle erläutert.

Mit Hilfe des Befehls DropDown lässt sich in einem Dialog eine Menüliste erzeugen. Wird ein Menüpunkt ausgewählt, so wird die Nummer dieses Menüpunktes in der Variablen `ant` gespeichert.





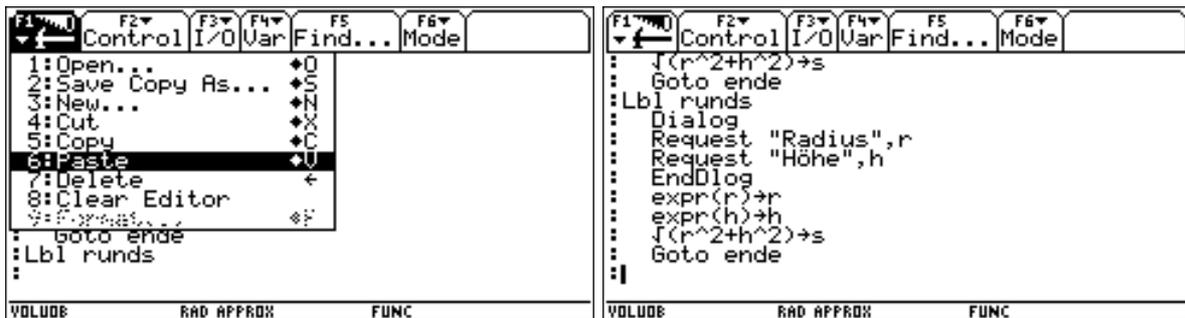
Anschließend wird mit Hilfe von mehreren If-Abfragen entsprechend der Antwort mittels Goto-Befehl zu einem entsprechenden Label, der weiter unten erstellt wird, verwiesen. Nach den einzelnen Labels sind nun die entsprechenden Befehle für Berechnung und Ausgabe einzufügen. Wir erstellen zunächst die Abfrage von Radius und Höhe im Label rundh, wandeln die Eingabe wieder in Zahlen um, berechnen s und Verzweigen zum Label ende. Dort erfolgt dann die Berechnung und Ausgabe von Volumen und Oberfläche für alle drei Fälle gleich.



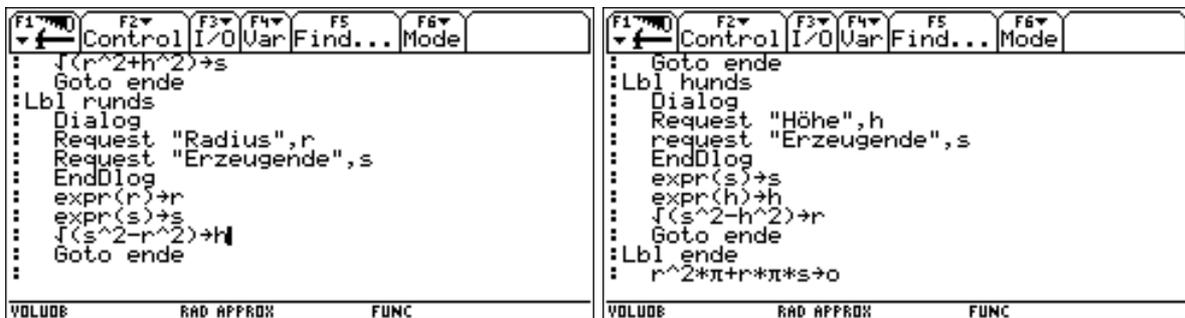
Nun müssen nur mehr die beiden Labels runds und hunds fertiggestellt werden. Da diese aber sehr ähnlich zum Label rundh sind, werden wir die Einträge aus rundh zunächst nach runds kopieren und dann die entsprechenden Änderungen vornehmen. Zunächst muss der zu kopierende Teil markiert werden. Wir stellen den Cursor an den Anfang des zu markierenden Blocks, drücken die SHIFT-Taste und bewegen den Cursor nun zum Ende des zu markierenden Blocks. Mittels F1/5 wird der Block in die Zwischenablage kopiert.



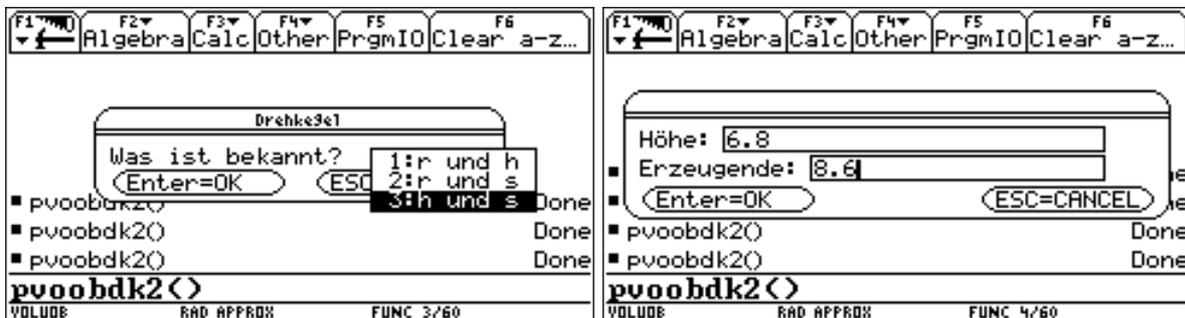
Stellen wir den Cursor nun an die Stelle, an der der Block eingefügt werden soll, und fügen ihn mittels F1/6 wieder ein.

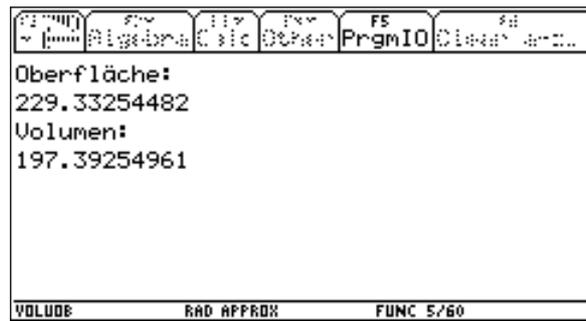


Nun sind nur noch die Änderungen vorzunehmen und der Vorgang für den Label hunds zu wiederholen.



Nun kann das Programm endgültig gestartet werden





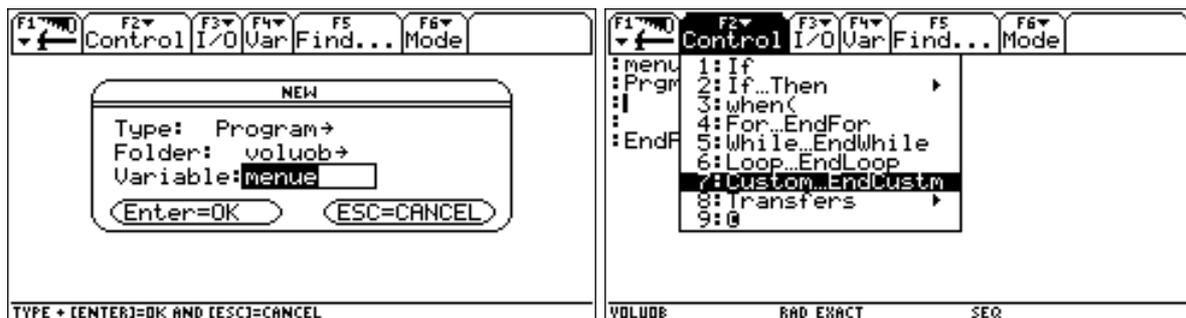
Im Anschluß befindet sich nun noch einmal das gesamte Programmlisting:

```
( )
Prgm
Local s,h,r,ant,o,v
ClrIO
Dialog
Title "Drehkegel"
DropDown "Was ist bekannt?",{ "r und h", "r und s", "h und s"},ant
EndDlog
If ant=1
  Goto rundh
If ant=2
  Goto runds
If ant=3
  Goto hunds
Lbl rundh
  Dialog
  Request "Radius",r
  Request "HÖhe",h
  EndDlog
  expr(r)»r
  expr(h)»h
  §(r^2+h^2)»s
  Goto ende
Lbl runds
  Dialog
  Request "Radius",r
  Request "Erzeugende",s
  EndDlog
  expr(r)»r
  expr(s)»s
  §(s^2-r^2)»h
  Goto ende
Lbl hunds
  Dialog
  Request "HÖhe",h
  Request "Erzeugende",s
  EndDlog
  expr(s)»s
  expr(h)»h
  §(s^2-h^2)»r
  Goto ende
Lbl ende
  r^2*E+r*E*s»o
  r^2*E*h/3»v
  Disp "Oberfläche: ",o
  Disp "Volumen: ",v
EndPrgm
```

Erstellen eines eigenen Menüs

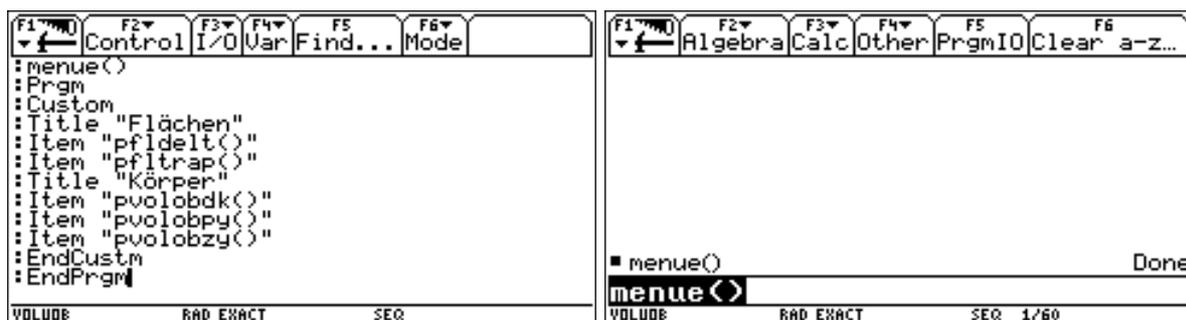
Abschließend soll nun noch gezeigt werden, wie eine zusätzliche Menüleiste erstellt werden kann. Über diese kann man dann eigene Programme und Funktionen aufrufen.

Angenommen es wurden ebenso wie für den Drehkegel Programme für andere Körper (pvolobpy für Pyramide, pvolobzy für Zylinder usw.) und auch Programme zur Flächenberechnung (z.B. pfdelt für Deltoid, pfltrap für Trapez usw.) erstellt. Die Verwirklichung eines Menüs, das all diese Programme aufrufen kann sieht dann folgendermaßen aus: Wir wählen im Programmreditor wieder New, bleiben im Folder voluob und geben als Name menue ein. In der nun erscheinenden Programmstruktur fügen wir den Befehl Custom – EndCustom für den Aufruf eines benutzerdefinierten Menüs ein

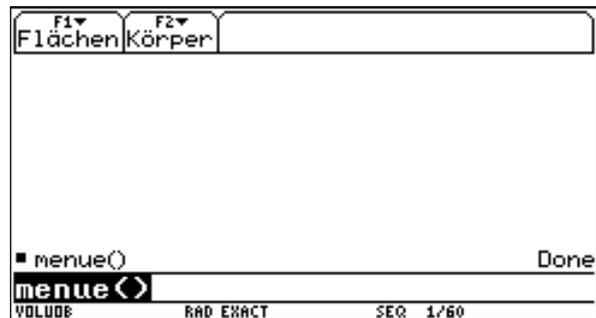


Mit dem Befehl Title wird die Überschrift des Pulldownmenüs, mit dem Befehl Item werden die einzelnen zu dieser Überschrift gehörenden Befehle festgelegt. Wir legen eine Menüleiste mit zwei Pulldownmenüs an, eines für die Flächen und eines für die Körper. Bei den Items sollte beachtet werden, dass es nur dann sinnvoll ist, die Klammern bereits wieder zu schließen, wenn kein Parameter eingegeben werden sollen. Wird zum Beispiel die Funktion fobdk als Item eingefügt, so wäre es sinnvoller Item "fobdk(" zu schreiben. Dadurch ist der Cursor dann beim Aufruf direkt an der Stelle, an der die Parameter eingegeben sind.

Wenn man das Programm anschließend startet, so läuft es ab und endet mit der Meldung done, man sieht aber zunächst keine Änderung:



Über 2nd Custom können wir aber unsere Menüleiste aufrufen. Über 2nd Custom erhalten wir auch wieder die originale Menüleiste.



Die Menüleiste lässt sich nun wie gewohnt bedienen. Über Enter gelangt der gewünschte Befehl in die Eingabezeile.



Nach ein weiteren Enter startet das Programm bereits wie gewohnt

