==========

# "PROGMATH" - Programmieren in Mathematik
# (c) Herbert Paukert

==========



==========

"PROGMATH" ist ein Texteditor mit dessen Hilfe so genannte
GEOMATH-Skripts erstellt, ausgeführt und gespeichert werden.
Im Texteditor ist dazu ein Mathematik-Interpreter eingebaut.
"PROGMATH" ist der Kern des Programmes "GEOMATH".

Menü-Eintrag <Bibliothek> öffnet/schließt eine Bibliothek,
welche fertige Skripts und auch pdf-Dateien zur gesamten
Schulmathematik enthält!

<Datei-Demo-1> öffnet eine Sammlung von zehn Demoskripts
mit Berechnungen und Zeichnungen (ideal zur Einführung).
<Datei-Demo-2> öffnet eine Sammlung von zehn Demoskripts
mit interaktiven Antworteingaben und deren Bewertungen.
<Datei-Demo-3> öffnet Demoskripts aus der Integralrechnung.

Menü-Eintrag <Skript ausführen> führt Skripts im Editor aus.
Dazu muss der Cursor auf den Befehl "begin" platziert werden.
Ein Skript kann entweder offen oder geschlossen ablaufen,
d.h. entweder mit geöffnetem oder mit verdecktem Texteditor.
Für bestimmte Skripts kann der Editor auch gesperrt werden.
Jedes Skript kann jederzeit mit <Esc> abgebrochen werden.

---
HINWEIS: Skriptdateien können beim <Speichern> auch verschlüsselt
werden. Dabei wird automatisch vor dem Dateinamen ein Underline
"_" gestellt. Beim <Öffnen> werden solche verschlüsselte Dateien
intern wieder entschlüsselt. Der Texteditor ist dabei gesperrt.
---

"Drag and Drop" verschiebt markierte Textteile.
<Strg><Pos1> springt immer an den Textanfang.
<Strg><Ende> springt immer an das Textende.
Ein Zeilenvorschub erfolgt mittels Taste <Enter>.
Ein Seitenvorschub erfolgt mittels <Strg><Enter>.
Das Zeichen "PI" symbolisiert einen Seitenvorschub.

---
HINWEIS: Bei entsprechender Verwendung der GEOMATH-Befehle können
in einem Skript interaktiv Antworten abgefragt und eingegeben und
schließlich bewertet werden. Dafür wird der INVAR-Befehl verwendet
(siehe Seite 09). Auf Seite 25 befindet sich dazu ein Demoskript.

In den Eingabezellen des INVAR-Befehls kann mit einem Doppelklick
ein Taschenrechner aufgerufen werden. Beim Schließen des Taschen-
rechners werden alle Rechenergebnisse automatisch in die Eingabe-
zellen übertragen. Mit einem Steuerzeichen kann der Taschenrechner
auch gesperrt werden (siehe Seite 03).
---

Das Menü <Bearbeiten> enthält wichtige Funktionen.
<TrimAllLines> entfernt alle mehrfachen Leer-Zeilen
und auch alle mehrfachen Leer-Zeichen (Blanks).
<KompressLines> wirkt ähnlich wie <TrimAllLines>,
sollte aber nur bei GEOMATH-Skripts angewendet werden.
Der Schalter <Grafik> blendet eine Grafik ein und aus.
Ein rechter Mausklick in die Grafik zeigt die Koordinaten.

Tastaturbelegungen:

<F1> ... Mathematik-Operation aus einer Zeile durchführen.
<F11> ... Mathematik-Skript aus mehreren Zeilen durchführen.
<F11> entspricht dem Menü-Eintrag <Skript ausführen>.
Mit <Shift F11> wird die Auflistung aller Befehle
eines Bibliotheks-Skripts im Editor ein-/ausgeschaltet.

<F2> ... alle Zeichencodes der aktuellen Schrift anzeigen.
<F4> ... eine sichtbare Grafik drucken oder speichern,
(hängt davon ab, ob mit <F6> eine Umleitung erfolgte).
<F5> ... eine gespeicherte Grafik laden.
<F6> ... Umleitung von <F4> "Grafik drucken" auf "Grafik speichern".
(Ein-/Aus-Schalter)
<F7> ... den Taschenrechner einblenden.
<F9> ... statistische Auswertung von Daten, die untereinander stehen
oder durch Kommas getrennt nebeneinander stehen. Sie müssen
mit der Maus genau markiert werden. Dezimalzahlen werden mit
einem Dezimalpunkt geschrieben.

<F10> ... Schrift in Grafiken normalisieren.
<Shift><F10> ... Schrift in Grafiken vergrößern.
<Strg><F10> ... Schrift in Grafiken verkleinern.
<F12> ... Fettschrift in Grafiken ein-/ausschalten.
<Strg><F7>, <Shift><F7>, <Strg><F8>, <Shift><F8>, <Strg><F9>
und <Shift><F9> sind mit internen Spezialfunktionen belegt.
Diese Funktionen sind nur im sichtbaren Editor ausführbar.

## (1) Die GEOMATH-Befehle

Es kann in einer Textzeile eine mathematische Rechenformel mit beliebigen Zahlen und den konventionellen Operatoren und Funktionen eingegeben werden. Außerdem können die 26 Variablen a,b,c,... y,z zur Speicherung verwendet werden. Eine mathematische Berechnung wird ausgeführt, indem man den Cursor auf die Zeile stellt und auf <F1> drückt. Folgende Möglichkeiten bestehen:

| | | |
|---|---|---|
| a = Zahl | <F1> | speichert die Zahl auf a. |
| a = Formel(a,b,...) | <F1> | wertet die Formel aus und speichert den Wert auf a. |
| a = | <F1> | zeigt den Wert von a an. |
| Formel(a,b,...) = | <F1> | wertet die Formel aus und zeigt den Formelwert an. |

OPERATOREN: (,),+,-,*,/,^.
FUNKTIONEN: abs,acos,asin,atan,cos,deg,exp,fak,log,ln,
rad,round,sin,sqr,sqrt,tan,trunc,pi.

Hinweis: Dezimalzahlen immer mit Dezimalpunkt schreiben.
Die Anzahl der Dezimalstellen ist standardmäßig 2. Sie
kann mit <Bearbeiten-Dezimalstellen> eingestellt werden.

**Beginnt eine Zeile mit //, dann wird sie als Kommentarzeile interpretiert und nicht ausgeführt.**

**---**
**HINWEIS: Wenn in der zweiten Zeile eines Skripts genau die Steuerzeichen //* ... stehen, dann wird im Programm NUR das Skript dargestellt - alle anderen Funktionen sind gesperrt. Bei den Steuerzeichen //*+ ... wird zusätzlich auch noch der Taschenrechner gesperrt. //*! ... sperrt den Drucker.**
**---**

**Ein Punkt "." am Anfang einer Zeile wird immer ignoriert. Wird ein Punkt vor Wertzuweisungen (.a =) geschrieben, dann können diese Zeilen mit <Bearbeiten-Komprimieren> oder dem Befehl "COMPRESS" nicht entfernt werden. Dieser Befehl ist notwendig um alte Wertzuweisungen aus den Textzeilen zu entfernen, aber neue Wertangaben zu erhalten.**

**Neben den einfachen Wertzuweisungen (siehe oben) stehen 140 arithmetische und geometrische Spezialbefehle zur Verfügung, welche unten aufgelistet sind. In den Funktionsformeln F(x) ist x das Argument und es können auch die anderen Variablen a,b,... y,z für Zahlen verwendet werden, z.B. sqrt(a^2-x^2).**

**In vielen geometrischen Befehlen sind die Parameter in Klammern symbolische Bezeichner für Punkte A,B,C,... Y,Z. Die Punkte müssen vorher gespeichert (gezeichnet) werden, was durch Niederschreiben der Koordinaten und Drücken der Taste <F1> erfolgt, z.B. A(-2,3.5) oder B(4.72,-8.57). Die Koordinaten werden dabei durch Beistriche getrennt. Zwischen Klein- und Großschreibung wird nicht unterschieden. Ein "=" am Anfang vor einer Punktvariablen wird ignoriert.**

**Viele GEOMATH-Befehle liefern als Ergebnis Zahlen oder Punkte, welche dann in den nachfolgenden Zeilen automatisch in den Text eingefügt werden. Diese Ergebniszeilen enthalten immer das Zeichen "=" und müssen am Skriptanfang mit dem Befehl "COMPRESS" aus dem Text entfernt werden. Soll eine Zeile, welche das Zeichen "=" enthält nicht gelöscht werden, dann muss am Zeilenanfang ein Punkt "." geschrieben werden.**

**Während eines Skriptdurchlaufs sind für die Ergebnisse bestimmte Variable reserviert. Diese sollten vorher nicht verwendet werden, weil ihre Werte dann überschrieben werden. Wird mit ihnen weitergerechnet, ist eine Umspeicherung sinnvoll. Beispielsweise durch eine Zahlenzuweisung .a = k, wo der Wert der Variablen k auf die Variable a gespeichert wird; oder durch eine Punktezuweisung UMS(A,S), wo der Punkt S mit dem neuen Namen A gezeichnet und intern abgespeichert wird.**

**Sieben reservierte Zahlenvariable d, k, p, q, x, y, z:**
**d,k ... Abschnitt und Anstieg von berechneten Geraden**
**p,q ... Zusatzergebnis bei speziellen Befehlen (z.B. VGG)**
**x,y,z ... Koordinaten von Punkten**
**z ... Ergebnis bei Längen, Winkeln und Flächen**

**Acht reservierte Punktvariable N, E, W, P, Q, S, T, Z:**
**N,E,W ... Nullstellen, Extremstellen, Wendestellen**
**P,Q ... Ergebnispunkte bei speziellen Befehlen (z.B. VGG)**
**S ... Schnittpunkt bei allen Schnittaufgaben**
**T ... Berührpunkt von Tangenten**
**Z ... Ergebnis bei Punkten, Vektoren oder komplexen Zahlen**

## (2) Die GEOMATH-Skripts

Ein GEOMATH-Skript besteht aus einer Folge von Textzeilen in denen die Mathematik-Befehle stehen. In der ersten Zeile muss immer "begin" stehen und in der letzten Zeile immer "end.". Wenn man den Mauscursor in die erste Zeile stellt und die Taste <F11> betätigt oder den Menü-Eintrag <Skript ausführen> anklickt. Dann werden die nachfolgenden Befehle so lange durchlaufen bis der Befehl "end." auftritt. Auf diese Weise sind im Texteditor ein Parser und ein Interpreter integriert, welche das Erkennen und das Ausführen der Befehle bewerkstelligen.

BEISPIEL: Eine Gerade g geht durch die Punkte A(-2,1) und B(7,5). Ermittle das Lot vom Punkt C(2,8) auf die Gerade g, den Lotfußpunkt F und den Punktabstand e von der Geraden. (Die eingerückten Zeilen sind die automatisch generierten Ergebniszeilen der Befehle).

```
begin
COMPRESS                    // Zeilenkomprimierung
KOR(10,1)                   // Festlegung des Koordinatensystems
A(-2,1)                     // Speicherung und Zeichnung von Punkt A
B(7,5)                      // Speicherung und Zeichnung von Punkt B
C(2,8)                      // Speicherung und Zeichnung von Punkt C
GER(A,B)                    // Gerade durch Punkte A und B ermitteln
  y = 0.44 * x + 1.89       // Erste Ergebniszeile  (Gleichung)
  k = 0.44                  // Zweite Ergebniszeile (Anstieg)
  d = 1.89                  // Dritte Ergebniszeile (Abschnitt)
LOT(C,A,B)                  // Das Lot von C auf die Gerade g(A,B)
  y = -2.25 * x + 12.50     // Erste Ergebniszeile  (Gleichung)
  k = -2.25                 // Zweite Ergebniszeile (Anstieg)
  d = 12.50                 // Dritte Ergebniszeile (Abschnitt)
  = S(3.94,3.64)            // Vierte Ergebniszeile (Schnittpunkt)
  z = 4.77                  // Fünfte Ergebniszeile (Abstand)
.e = z                      // Umspeicherung von z auf e
UMS(F,S)                    // Umspeicherung von S auf F
  = F(3.94,3.64)            // Erste Ergebniszeile  (Punkt F)
end.
```

## (3) Liste aller GEOMATH-Befehle

**COMPRESS** komprimiert alle Befehlszeilen und löscht dabei alle Zeilen, welche eine Wertzuweisung "=" enthalten. Soll eine Zeile, die das Zeichen "=" enthält, nicht gelöscht werden, dann muss am Anfang der Zeile unbedingt ein Punkt "." geschrieben werden. Dadurch werden auch Wertzuweisungen zu Variablen nachhaltig fixiert.

Der Befehl sollte immer am Skriptanfang NACH "begin" stehen. Dadurch werden die Ergebnisse eines vorangegangenen Skriptdurchlaufs gelöscht. Der Befehl funktioniert nicht im Einzelschritt-Modus mit <F1>. Im Einzelschritt-Modus muss zum Komprimieren der Skript-Zeilen der Menüpunkt <Bearbeiten-Komprimieren> gewählt werden.

```
CLR(n)           Grafik (n=0) und/oder Variable (n=0,n=1) löschen.
DEZ(d)           Anzahl der angezeigten Dezimalstellen d (maximal 14).
ROU(a,d)         Rundet die Zahl a intern auf d Dezimalstellen.
CUT(a,d)         Beschneidet die Zahl a intern auf d Dezimalstellen.
ZZF(a,n)         liefert von der Zahl a die Ziffer an der Stelle n.
HZF(a)           liefert den höchsten Stellenwert (10^n) der Zahl a.
MOD(a,b)         Ganzzahliger Rest bei Division a/b.
KOM(n,k)         Kombinationen (k aus n).
GGT(a,b)         größter gemeinsamer Teiler der Zahlen a,b.
KGV(a,b)         kleinstes gemeinsames Vielfache von a,b.
PFT(a)           prüft ob a eine Primzahl ist und liefert 0 oder 1.
PFZ(a)           führt eine Primfaktorenzerlegung von a durch.
ZUF(a,b)         erzeugt ganze Zufallszahlen z mit a <= z <= b.
                 ZUF(0,1)  erzeugt reelle Zahlen zwischen 0 und 1.
                 ZUF(-1,1) erzeugt zufällig -1 oder +1.
IND(0)           liefert den Listenindex bei "zufun0" (auf Variable z)
MIN(a,b,c,...)   bestimmt das Minimum der Werte a,b,c,...
MAX(a,b,c,...)   bestimmt das Maximum der Werte a,b,c,...
KAL(Term)        berechnet den Zahlenwert des mathematischen Terms.

KOR(b,r)         löscht die Grafik und setzt die Halbbreite b
                 des Koordinatensystems. Bei r = 1 werden
                 die beiden Koordinatenachsen gezeichnet,
                 bei r = 0 hingegen nicht. Bei r = 2 wird
                 ein vollständiger Raster eingezeichnet.
RAS(b,r)         wie KOR(b,r), aber ohne Löschung der Grafik.
PEN(d,f)         setzt die Stiftdicke d (1=dünn,2=mittel,3-100=dick)
                 und Zeichenfarbe f (-1=keine,0=weiß,1=schwarz,2=rot,
                 3=grün,4=blau,5=gelb,6=magenta,7=cyan,8=weiß).
PFA(f)           setzt Punkt-/Textfarbe f (-1,0,1,2,3,4,5,6,7,8).
FIL(x,y,f,r,g,b) füllt einen mit Randfarbe f (0,1,2,3,4,5,6,7,8)
                 begrenzten Bereich, ausgehend von dem
                 Punkt (x,y), mit der Füllfarbe von Rot r,
                 Grün g und Blau b (0 <= r,g,b <= 255).
PNT(P)           Anzeigen des Punktes P.
UMS(A,S)         speichert den Punkt S auf den Punkt A um,
                 d.h. er erhält einen neuen Namen.
KXY(P)           liefert die Koordinaten x und y des Punktes P,
                 bzw. x, y und z im dreidimensionalen Fall.

TXN(Schrift)     Angabe des Schriftnamens für die Grafik, z.B.
                 "Arial", "Courier_New", "Symbol", "MS_LineDraw".
TXR(p)           Text ohne (p=0) oder mit (p=1) weißem Rechteck.
TXG(p)           relative Schriftgröße (von -5 bis +5, 0 = Standard).
TXF(p)           setzt Fettschrift aus (p = 0) oder ein (p = 1).
TXT(x,y,Text)    gibt an den Koordinaten (x,y) den Text aus.
                 Blanks werden mit Underlines markiert.
                 Ein " vor einem Buchstaben bewirkt Großschreibung.
TEX(x,y,Text)    gibt den Wert von Variablen <a> oder Punkten [P] aus.
TXZ(n,Text)      gibt in der n-ten Zeile den Text original aus
                 (d.h. Leerstellen, Großbuchstaben und Beistriche).
                 Mit maximal 25 Zeilen und 65 Zeichen pro Zeile.
TXV(n,Text)      gibt in der n-ten Zeile den Text original aus
                 (d.h. Leerstellen, Großbuchstaben und Beistriche).
                 Mit maximal 25 Zeilen und 65 Zeichen pro Zeile.
                 Zusätzlich können im Text auch Variablenwerte und
                 Punktkoordinaten ausgegeben werden. (Siehe dazu
                 die Steuerungsanweisung "pause" auf Seite [09]).
LIB(A,B,Text)    beschriftet die Strecke AB mit einem Text.
WIB(A,B,C,r,Text) beschriftet den Winkel w(ABC) mit einem
                 Text im Abstand r vom Winkelscheitel B.
```

```
LNG(A,B)          ermittelt die Länge der Strecke AB.
WIN(A,B,C)        ermittelt den Winkel w(ABC) mit Scheitel B.
FLA(A,B,C)        ermittelt die Fläche des Dreiecks ABC.

HAP(A,B)          ermittelt den Halbierungspunkt von AB.
SWP(A,B,C)        ermittelt den Schwerpunkt von Dreieck ABC.
REC(A,C)          zeichnet ein Rechteck mit Diagonale AC.
GER(A,B)          zeichnet die Gerade y = k*x + d durch A und B.
                  Bei Senkrechten werden x, k=10^10, d=10^10 ausgegeben!
GWG(A,B,w)        ermittelt jene Gerade durch Punkt A, die
                  mit der Strecke AB den Winkel w bildet.
STW(A,B)          Steigungswinkel der Strecke AB.
SWI(k)            Steigungswinkel zum Anstieg k.
STP(A,B,r)        trägt vom Punkt A auf der Geraden g(A,B).
                  in Richtung von B die Streckenlänge r ab.
LOT(A,B,C)        zeichnet die normale Gerade von Punkt A
                  auf eine Gerade durch die Punkte B und C,
                  und ermittelt den Lotfußpunkt F und auch
                  den Abstand des Punktes A von der Geraden.
SSM(A,B)          zeichnet die Symmetrale der Strecke AB.
SYM(A,B,C)        zeichnet die Symmetrale des Winkels mit
                  Scheitel B und Schenkeln BA und BC.
LIN(A,B,C...)     zeichnet einen Streckenzug durch A,B,C...

SCH(P,a,b)        Schiebung des Punktes P um den Vektor (a,b).
DRE(P,Z,w)        Drehung von P mit Zentrum Z und Winkel w.
SPI(P,A,B)        Spiegelung von P an der Achse durch A und B.
STR(P,Z,k)        Streckung von P mit Zentrum Z und Faktor k.
NSC(a,b)          Verschiebt alle Punkte eines Streckenzuges (N-Eck),
                  der mit LIN(A,B,C,...) gezeichnet wurde, um den
                  Vektor (a,b).
NDR(Z,w)          Dreht alle Punkte eines Streckenzuges (N-Eck),
                  der mit LIN(A,B,C,...) gezeichnet wurde, um das
                  Zentrum Z mit Winkel w.
NSP(U,V)          Spiegelt alle Punkte eines Streckenzuges (N-Eck),
                  der mit LIN(A,B,C,...) gezeichnet wurde, an der
                  Achse durch U und V.
NST(Z,k)          Streckt alle Punkte eines Streckenzuges (N-Eck),
                  der mit LIN(A,B,C,...) gezeichnet wurde, mit dem
                  Zentrum Z und mit dem Streckungsfaktor k.

KRS(M,r)          Kreis mit dem Mittelpunkt M und Radius r.
KRU(A,B,C)        Umkreis des Dreiecks durch die Punkte A,B,C.
KRI(A,B,C)        Inkreis des Dreiecks durch die Punkte A,B,C.
KSE(M,r,v,w)      Kreissektor mit Mittelpunkt M, Radius r,
                  Anfangswinkel v und Endwinkel w.
KBO(M,r,v,w)      Kreisbogen (wie Kreissektor).
EBO(M,a,b,v,w)    Ellipsenbogen mit Mittelpunkt M, Halbachsen a,b
                  und Anfangswinkel v und Endwinkel w.
BOG(M,A,B)        zieht einen Kreisbogen mit Mittelpunkt M durch
                  die beiden Kreispunkte A und B.
RVE(n,r)          zeichnet ein regelmäßiges n-Eck mit n <= 256
                  und dem Umkreis mit M(0,0) und Radius r.
ELL(a,b)          Ellipse mit den Halbachsen a und b.
HYP(a,b)          Hyperbel mit den Halbachsen a und b.
PAR(p)            Parabel mit dem Parameter p.
EL2(A,B)          Ellipse durch die Punkte A und B.
HY2(A,B)          Hyperbel durch die Punkte A und B.
PA1(A)            Parabel durch den Punkt A.

PFL(A,B,d,f)      Pfeil von A nach B mit Dicke d und Farbe f.
                  (Ein neuerliches Zeichnen löscht den Pfeil).
```

| | |
|---|---|
| **FUN(F(x))** | **zeichnet eine beliebige Funktion F(x).** |
| **FUN(F(x),a,b)** | **zeichnet die Funktion F(x) im Intervall [a,b].** |
| **FNA(F(x))** | **zeichnet die erste Ableitung einer Funktion.** |
| **FNT(F(x),n)** | **zeichnet n gleitende Tangenten entlang der Funktionskurve F(x) mit n = 10,11,...1000.** |
| **AFF(k,F(x))** | **Achsenaffinität der Funktion F(x) mit Faktor k.** |
| **BEW(u,v,w,F(x))** | **führt mit Funktion F(x) eine Bewegung aus, zuerst eine Drehung mit dem Winkel w um den Ursprung und dann eine Schiebung um (u/v).** |
| **GAU(z)** | **liefert Werte der Verteilungsfunktion F[0;z] der Standard-Normalverteilung (auf Variable f).** |
| **NUL(F(x),n,a,b)** | **ermittelt eine reelle Nullstelle der n-ten Ableitung der Funktion F(x) auf dem Intervall [a,b] mit n = 0,1 oder 2. Wird dort keine Nullstelle gefunden, so kann mit <Esc> abgebrochen werden.** |
| **DIF(F(x),n,a)** | **n-te Ableitung der Funktion F(x) an der Stelle x = a mit n = 0,1 oder 2.** |
| **INT(F(x),a,b)** | **bestimmtes Integral der Funktion F(x) auf dem Intervall [a,b] mit a < b. Die Funktion muss dort stetig sein und es darf kein Vorzeichenwechsel auftreten.** |
| **TKP(F(x),a)** | **ermittelt die Tangente an die Kurve F(x) durch den Kurvenpunkt P(a,F(a)).** |
| **TNG(F(x),a,b,c,d)** | **ermittelt die Tangente an die Kurve F(x) durch den Nicht-Kurvenpunkt P(a,b), wo der Berührpunkt im Intervall [c,d] liegen muss. Wird dort kein Berührpunkt gefunden, so kann mit <Esc> abgebrochen werden.** |
| **SGG(k1,d1,k2,d2)** | **ermittelt den Schnittpunkt der beiden Geraden y = k1*x+d1 und y = k2*x+d2.** |
| **SKG(M,r,k,d)** | **Schnittpunkte von Gerade y = k*x+d und Kreis mit Mittelpunkt M und Radius r.** |
| **SKK(M1,r1,M2,r2)** | **Schnittpunkte zweier Kreise mit den Mittelpunkten M1, M2 und den Radien r1, r2.** |
| **SFF(F1(x),F2(x),a,b)** | **ermittelt den Schnittpunkt der beiden Kurven F1(x) und F2(x) auf Intervall [a,b]. Wird dort kein Schnittpunkt gefunden, so kann mit <Esc> abgebrochen werden.** |
| **DET(A,B,C)** | **Determinante mit den Zeilenvektoren A,B,C; auch zweidimensional mit DET(A,B) möglich.** |
| **LGS(A,B,C,D)** | **Lineares Gleichungssystem mit den linksseitigen Zeilenvektoren A,B,C und dem rechtsseitigen Spaltenvektor D. Auch mit LGS(A,B,C) möglich.** |
| **LG4(0)** | **Lineares Gleichungssystem in 4 Variablen (w,x,y,z) mit den 20 Koeffizienten (a,b,..,s,t). Analog dazu sind auch LG3(0) und LG2(0) möglich.** |
| **QUA(a,b,c)** | **Lösungen der Gleichung $ax^2 + bx + c = 0$.** |
| **DZN(a,n)** | **Umwandlung der Zahl a vom 10- ins n-System.** |
| **NZD(a,n)** | **Umgekehrte Umwandlung mit 2 <= n <= 36.** |
| **SRD(v,w)** | **definiert für die Schrägrißdarstellung den Verzerrungsfaktor (v) und den Winkel (w). PFA(-1) unterbindet die Darstellung der Achsen.** |
| **SRK(P)** | **stellt einen dreidimensionalen Punkt P im Schrägriß als zweidimensionalen Punkt dar.** |

```
Mit Punkten als KOMPLEXE ZAHLEN kann auch gerechnet werden:
KPX(Term(A,B,C,.....))  ermittelt den Wert des Terms mit
                        den komplexen Zahlen A,B,C,......
                        Zulässige Operatoren: +,-,*,/,^,(,).
                        Nach dem Potenzoperator ^ muss ein
                        reeller Rechenausdruck folgen, z.B.
                        KPX(A+B^(1/2)).
ADD(A,B)        Addition zweier komplexer Zahlen.
SUB(A,B)        Subtraktion zweier komplexer Zahlen.
MUL(A,B)        Multiplikation zweier komplexer Zahlen.
DIV(A,B)        Division zweier komplexer Zahlen.
POT(A,n)        n-te Potenz der komplexen Zahl A.
WUR(A,n)        n-te Wurzel der komplexen Zahl A.
POL(A)          liefert die Polarkoordinaten (r,w) von A.
BIN(r,w)        Kartesische Koordinaten (x,y) von (r,w).

Mit Punkten als VEKTOREN kann ebenfalls gerechnet werden.
Dabei werden zwei oder drei Vektor-Koordinaten eingegeben.
Die Anzahl der Koordinaten wird automatisch erkannt.
VEK(Term(A,B,C,.....))  ermittelt den Wert des Terms mit
                        den Vektoren A,B,C,..... Dabei
                        zulässige Operatoren: +,-,*,(,).
                        Der Operator * bedeutet die Multi-
                        plikation eines Vektors mit einem
                        reellen Ausdruck, z.B. VEK(A+B*(1/2)).
VSU(A,B)        Vektorsumme A+B.
VDI(A,B)        Vektordifferenz A-B.
VFA(A,k)        Vektorvielfaches k*A.
VPR(A,B)        Vektorprodukt von A und B.
VSK(A,B)        Skalarprodukt von A und B.
VLE(A)          Länge eines Vektors A.
VWI(A,B)        Winkel zwischen A und B.
VFL(A,B)        Fläche zwischen A und B (Parallelogramm).
VOL(A,B,C)      Volumen zwischen A, B, C (Parallelepiped).
VEV(A)          Normierter Vektor von A (Einheitsvektor).
VNV(A)          Normale Vektoren auf A.
VNA(P,A,N)      Abstand des Punktes P von der Geraden (2-dim) oder
                Ebene (3-dim) durch Punkt A mit dem Normalvektor N.
                Außerdem wird der Lotfußpunkt F ermittelt.
VGG(A,B)        Geradengleichungen im Raum durch die Punkte A,B.
                Ausgabe: <funtex1> und <funtex2> und die beiden
                Normalvektoren P und Q und die Konstanten p und q.
VEG(A,B,C)      Ebenengleichung im Raum durch die Punkte A,B,C.
                Ausgabe: <funtex1> und Normalvektor Z und Konstante z.
VNG(A,N)        Gleichung der Geraden (2-dim) oder Ebene (3-dim)
                durch den Punkt A mit dem Normalvektor N.
                Ausgabe: <funtex1> und Normalvektor Z und Konstante z.

SCS(0)          Zwischenspeicherung der aktuellen Grafik.
SCR(0)          Wiederherstellung der zwischengespeicherten Grafik.
LPC(Name,g)     Lädt eine JPEG-Datei "Name" in das Koordinatenfenster,
                mit originaler Größe (g=0) oder angepasst (g=1).
LSN(Name)       Spielt eine Sounddatei ab (WAV, WMA oder MP3).
                Wenn Name = '0' ist, dann wird der Sound abgeschaltet.
PAG(0)          Markiert im einen Seiten-Anfang. Dann kann man beim
                "pause"-Befehl mit <F2><Enter> eine Seite zurückgehen.

SXY(n)          Speichert die Variablen x und y auf die internen
                Listenelemente DatX[n] und DatY[n]. Die Elemente
                DatX[0] und DatY[0] enthalten die Datenanzahl (<=100).
WXY(0)          Schreibt die interne Liste in ein CSV-Textfile.
RXY(0)          Liest ein CSV-Texfile in die interne Liste
LXY(n)          Lädt die internen Listenelemente DatX[n] und DatY[n]
                auf die Variablen x und y.
DXY(0)          Löscht alle internen Listenelemente.
```

```
================================================================
(4) STEUERUNGS-Anweisungen
================================================================

Neben diesen 140 Mathematikbefehlen gibt es noch 35 Anweisungen
zur Steuerung des Skriptablaufes. Eine Steuerungsanweisung
ist bereits oben genau beschrieben worden, nämlich "compress".

begin               Erste Zeile eines GEOMATH-Skripts
end.                Letzte Zeile eines GEOMATH-Skripts
wait(t)             Wartet t Millisekunden im Skriptablauf.
pause(Text)         Erzeugt eine Pause und zeigt den Text an.

Wenn im Pausentext eine Variable zwischen spitzen Klammern < >
steht, dann wird dort ihr Zahlenwert angezeigt. Wenn im Text
ein Punktname zwischen eckigen Klammern [ ] steht, dann werden
dort die Punktkoordinaten angezeigt, dreidimensional mit { }.
Das Zeichen & im Pausentext bewirkt einen Zeilenumbruch.

Mit "pause(<funtex1>)" wird der Wert der internen Textvariablen
"funtex1" angezeigt. Hingegen wird mit "pause(<funtex2>)" der Wert
der zweiten Textvariablen "funtex2" angezeigt.

Beim Ausgabebefehl "pause" und bei den Eingabebefehlen "invar",
"input2", "input3", "intex1" und "intex2" kann mit Taste <Esc>
das Skript jederzeit abgebrochen werden. Beim Befehl "pause" kann
auch die aktuelle Grafik gedruckt werden (mit Funktionstaste <F4>).

select(Text: Var,N)  Erzeugt eine Auswahlbox mit N Schaltern
                     von 1 bis N. Der Auswahlschalter 0 ist
                     immer vorhanden. Die ausgewählte Nummer
                     wird auf die Variable Var gespeichert.
                     Zusätzlich wird der Text angezeigt.

invar(Text: Variablenliste)  Eingabe von skalaren Variablen-
                     werten. Dabei wird der Text angezeigt.
                     Wenn dieser Text ein Fragezeichen "?"
                     enthält, dann wird mit einem Doppelklick
                     in das Eingabefeld ein Taschenrechner
                     aufgerufen. Wird er geschlossen, dann
                     wird das Rechenergebnis in das Eingabe-
                     feld übertragen.

input2(Punktname)    Eingabe der Koordinaten x,y einer Punkt-
                     bzw. Vektor-Variablen (zweidimensional).
input3(Punktname)    Eingabe der Koordinaten x,y,z einer Punkt-
                     bzw. Vektor-Variablen (dreidimensional).

ifle(Var,W,Ziel)    Vergleicht die Variable Var mit dem Wert W.
                    Dabei kann W auch eine Variable sein.
                    Wenn Var <= W ist, erfolgt ein Sprung zu
                    jener Befehlszeile, wo der Ziel-Text steht.
                    Dort muss diesem Ziel-Text ein Underline "_"
                    vorangestellt werden (symbolische Adresse).
                    Wenn aber Var > W ist, wird weiter gegangen.

ifls(Var,W,Ziel)    Wie "ifle", aber nur wenn Var < W ist.
ifge(Var,W,Ziel)    Wie "ifle", aber nur wenn Var >= W ist.
ifgr(Var,W,Ziel)    Wie "ifle", aber nur wenn Var > W ist.
ifeq(Var,W,Ziel)    Wie "ifle", aber nur wenn Var = W ist.
ifne(Var,W,Ziel)    Wie "ifle", aber nur wenn Var <> W ist.
```

```
iferror(Ziel)       Wie "ifeq", aber nur bei internem Fehler.
seterror            Setzt die interne Fehlervariable auf 1.
delerror            Setzt die interne Fehlervariable auf 0.

Wenn bei einem mathematischen Befehl ein Fehler auftritt, dann
wird die interne Fehlervariable, welche normalerweise den Wert
Null hat, automatisch auf 1 gesetzt. Sie kann dann mit "iferror"
abgefragt und mit "delerror" wieder auf Null gesetzt werden.

goto(Ziel)          Springt unbedingt zu jener Befehlszeile, die
                    durch das "Ziel" symbolisch adressiert ist.
                    Im Listing muss vor den Namen des Sprungziels
                    ein Underline gestellt werden ("_Ziel").
exit                Bricht das Programm unbedingt ab.

Neben den Steuerungsanweisungen für den Verlauf des GEOMATH-Skripts
geibt es noch so genannte Stringbefehle. Sie beziehen sich auf die
Zuweisung, Eingabe, Speicherung und Verarbeitung von zwei internen
Stringvariablen "funtex1" und "funtex2". Diese können mit gültigen
mathematischen Ausdrücken oder Funktionstermen belegt werden und
sie versorgen dann folgende Befehle als Eingabeparameter:

NZD,KAL,KPX,VEK,TXT,TXZ,TXV,DIF,INT,FUN,FNA,BEW,NUL,TKP,TNG,SFF.
Zur Datenausgabe werden sie bei folgenden Befehlen verwendet:
DZN,PFZ,QUA,VNG,VGG,VEG.

setfun1(Formel)     Weist der Textvariablen "funtex1" eine Formel zu.
setfun2(Formel)     Weist der Textvariablen "funtex2" eine Formel zu.
                    Hier können auch die 4 Namen "funtex1", "-funtex1",
                    "funtex2" und "-funtex2" eingegeben werden.

intex1(Text)        Eingabe einer mathematischen Formel und
                    Speicherung auf der Textvariablen "funtex1".
                    Dabei wird der Text angezeigt.
intex2(Text)        Eingabe einer mathematischen Formel und
                    Speicherung auf der Textvariablen "funtex2".
                    Dabei wird der Text angezeigt.

stofun1             Speichert intern zusätzlich den Text von "funtex1".
getfun1             Holt den mit "stofun1" gespeicherten Text zurück.
stofun2             Speichert intern zusätzlich den Text von "funtex2".
getfun2             Holt den mit "stofun2" gespeicherten Text zurück.

fun1q               Bildet f(x)^2 mit der Formel f(x) in "funtex1".
fun1x               Bildet x*f(x) mit der Formel f(x) in "funtex1".
fun1m               Bildet x*f(x)^2 mit der Formel f(x) in "funtex1".

                    Die Ergebnisse sind auf der Variablen z gespeichert.

IBO(l,r)            Bildet das Integral von sqrt(1+f'(x)^2) mit f(x)
                    in "funtex1" zwischen den Grenzen l und r. Das wird
                    zur Berechnung von Bogenlängen verwendet.

IBX(l,r)            Bildet das Integral von x*sqrt(1+f'(x)^2) mit f(x)
                    in "funtex1" zwischen den Grenzen l und r. Das wird
                    zur Berechnung von Bogenschwerpunkten verwendet.

IBY(l,r)            Bildet das Integral von f(x)*sqrt(1+f'(x)^2) mit f(x)
                    in "funtex1" zwischen den Grenzen l und r. Das wird
                    zur Berechnung der Mantelfläche von Drehkörpern
                    verwendet.

                    Die Ergebnisse sind auf der Variablen z gespeichert.
```

```
 zufun1              Wählt aus einer Liste von mit Kommas getrennten
                     Texten (Formeln), welche mittels "setfun1(Liste)"
                     angelegt wird, zufällig einen Text (Formel) aus
                     und speichert ihn auf die Variable "funtex1".
 zufun2              Funktioniert wie "zufun1" - nur mit "funtex2".

 zufun0              Wählt aus zwei Listen "setfun1(Liste1)" und
                     "setfun2(Liste2)" schrittweise zwei Texte (Formeln)
                     aus, die in beiden Listen an der gleichen Position
                     stehen. Diese synchronen Texte werden dann auf
                     "funtex1" und "funtex2" gespeichert. Die Auswahl
                     beginnt mit dem ersten Listeneintrag und durch-
                     läuft die ganze Textliste. Am Ende beginnt der
                     Durchlauf wieder am Listenanfang.
                     Hinweis: Mit dem Befehl IND(0) wird die aktuelle
                     Listenposition der Variablen "z" zugewiesen.

 Das nachfolgende Skript demonstriert die oben beschriebenen String-
 Befehle. Um es auszuführen, kann das Skript im Hilfetext markiert
 und dann mit <Strg C> und <Strg V> in den Editor kopiert werden.

 begin
// Demo-Skript zum Befehl "zufun0"
 _anf
 clr(0)
 txr(1)

 _start
 kor(10,1)
 compress
 setfun1(2*x+3,0.5*x^2,0.3*x^3-x^2-3*x+4,1/x,sqrt(x),sqrt(25-x^2),3*sin(deg(x)))
 setfun2(2,x,0.9*x^2-2*x-3,-1/x^2,0.5/sqrt(x),-x/sqrt(25-x^2),3*cos(deg(x)))
 zufun0
 ind(0)
 .n = z
 pfa(4)
 txg(2)
 txz(1,Sieben Funktionen und ihre Ableitungen)
 txg(0)
 pfa(1)
 txv(23,Funktion: f(x) = <funtex1>          )
 pen(2,1)
 fun(funtex1)
 pause(Weiter)
 pfa(2)
 txv(24,Ableitung: f'(x) = <funtex2>        )
 pen(2,2)
 fun(funtex2)
 pause(<n>. Funktion)
 ifeq(n,7,aus)
 goto(start)

 _aus
 pause(Wiederholen oder Abbruch mit "Esc")
 goto(anf)
 end.

 ----------------------------------------------------------------
 Mit diesen Steuerungsanweisungen können Wertevergleiche, unbedingte
 und bedingte Sprünge, Wiederholungsschleifen, Dateneingaben und
 Datenausgaben und verschiedene Zusatzfunktionen realisiert werden.
 ----------------------------------------------------------------
```

```
================================================================
Zehn Programmbeispiele
================================================================

begin
// Programm 01
// mit einem unbedingten Sprung "goto(start)"
// Sprungziele beginnen immer mit einem Underline
_start
compress
clr(0)
dez(2)
kor(10,0)
txv(1,Einfache Rechnungen I)
.a = 3
.b = 2
invar(Zwei Zahlen a und b eingeben: a,b)
txv(2,a = <a>, b = <b>)
.c = a + b = 5.00
.d = a - b = 1.00
.e = a * b = 6.00
.f = a / b = 1.50
txv(4,<a> + <b> = <c>)
txv(5,<a> - <b> = <d>)
txv(6,<a> * <b> = <e>)
txv(7,<a> / <b> = <f>)
pause(Wiederholen oder Beenden mit "Esc")
goto(start)
end.

================================================================

begin
// Programm 02
// mit zwei unbedingten Sprüngen "goto(start)" und "goto(weiter)"
// und mit einem bedingten Sprung "ifeq(b,0,fehler)"
// Sprungziele beginnen immer mit einem Underline
_start
compress
clr(0)
dez(2)
kor(10,0)
txv(1,Einfache Rechnungen II)
.a = 3
.b = 0
invar(Zwei Zahlen a und b eingeben: a,b)
txv(2,a = <a>, b = <b>)
.c = a + b = 3.00
.d = a - b = 3.00
.e = a * b = 0.00
txv(4,<a> + <b> = <c>)
txv(5,<a> - <b> = <d>)
txv(6,<a> * <b> = <e>)
ifeq(b,0,fehler)
.f = a / b
txv(7,<a> / <b> = <f>)
goto(weiter)
_fehler
txv(7,<a> / <b> = ?, Division durch Null ist nicht möglich !)
_weiter
pause(Wiederholen oder Beenden mit "Esc")
goto(start)
end.
```

```
begin
// Program 03
// Koordinatensystem und Punkte eingeben
// Linienzug zeichnen
// Streckenlänge berechnen
_start
compress
clr(0)
dez(2)
.g = 10
kor(g,1)
txv(1,Punkte und Strecken im Koordinatensystem)
invar(Halblänge des Koordinatensystems: g)
kor(g,1)
txv(1,Punkte und Strecken im Koordinatensystem)
pfa(-1)
A(2,2)
B(8,7)
pfa(1)
input2(A)
input2(B)
lin(A,B)
lng(A,B)
     z = 7.81
.a = z = 7.81
txv(25,Länge von AB = <a>)
pause(Wiederholen oder Beenden mit "Esc")
goto(start)
end.
```

==========

```
begin
// Program 04
// Koordinatensystem und Punkte eingeben
// Linienzug zeichnen
// Streckenlängen berechnen
_start
compress
clr(0)
dez(2)
.g = 10
kor(g,1)
txv(1,Dreiecke im Koordinatensystem I)
invar(Halblänge des Koordinatensystems: g)
kor(g,1)
txv(1,Dreiecke im Koordinatensystem I)
pfa(-1)
A(1,3)
B(4,0)
C(8,8)
pfa(1)
input2(A)
input2(B)
input2(C)
lin(A,B,C,A)
lng(A,B)
     z = 4.24
.c = z = 4.24
lng(A,C)
     z = 8.60
.b = z = 8.60
```

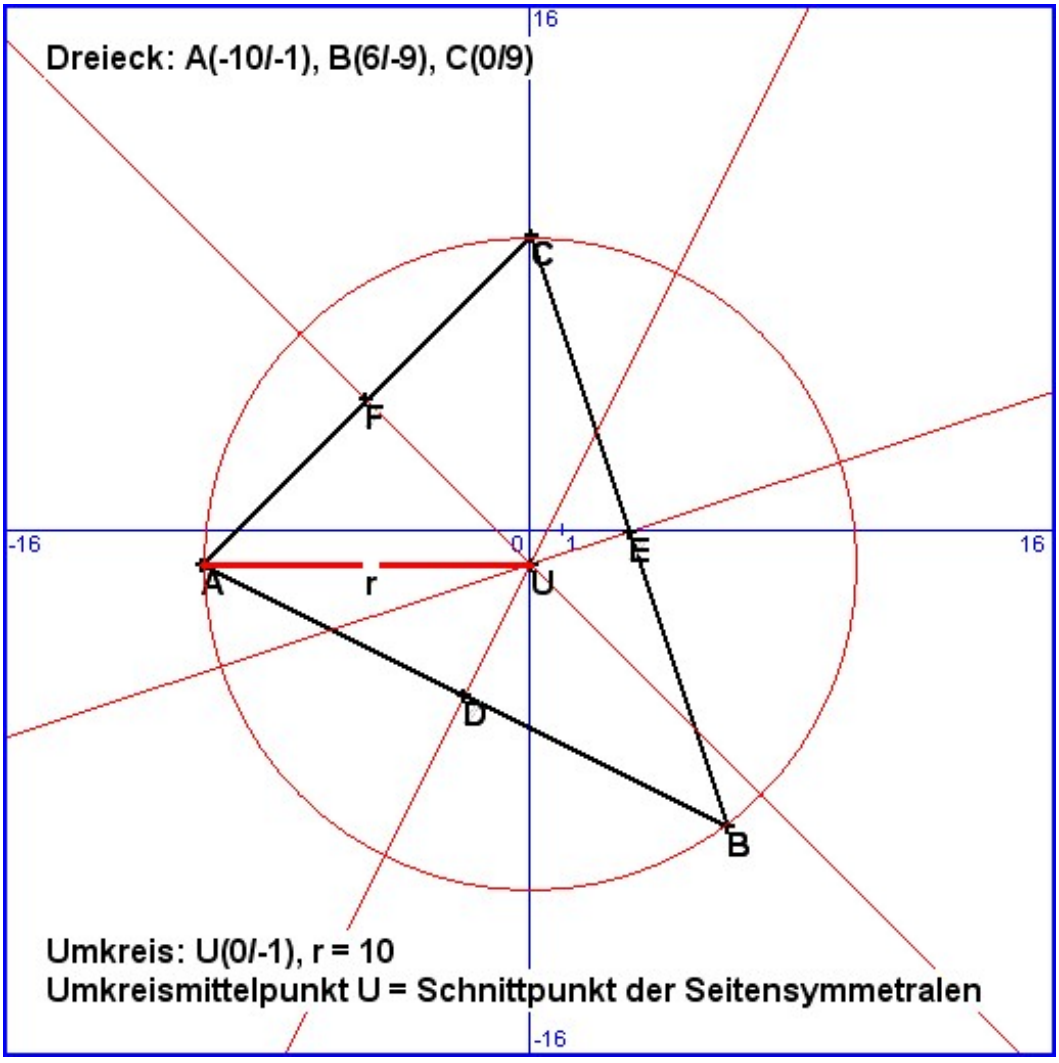
```
lng(B,C)
     z = 8.94
.a = z = 8.94
.u = a + b + c = 21.78
txv(24,c = AB = <c>, b = AC = <b>, a = BC = <a>)
txv(25,Umfang = a + b + c = <u>)
pause(Wiederholen oder Beenden mit "Esc")
goto(start)
end.

================================================================

begin
// Program 05
// Koordinatensystem und Punkte eingeben
// Linienzug zeichnen
// Streckenlängen berechnen
// Winkelgrößen berechnen
_start
compress
clr(0)
dez(2)
.g = 10
kor(g,1)
txv(1,Dreiecke im Koordinatensystem II)
invar(Halblänge des Koordinatensystems: g)
kor(g,1)
txv(1,Dreiecke im Koordinatensystem II)
pfa(-1)
A(1,3)
B(4,0)
C(8,8)
pfa(1)
input2(A)
input2(B)
input2(C)
lin(A,B,C,A)
lng(A,B)
     z = 4.24
.c = z = 4.24
lng(A,C)
     z = 8.60
.b = z = 8.60
lng(B,C)
     z = 8.94
.a = z = 8.94
.u = a + b + c = 21.78
win(B,A,C)
     z = 80.54
.i = z = 80.54
win(A,B,C)
     z = 71.57
.j = z = 71.57
win(A,C,B)
     z = 27.90
.k = z = 27.90
.s = i + j + k = 180.01
rou(s,0)
txv(21,c = AB = <c>, b = AC = <b>, a = BC = <a>)
txv(22,Umfang = a + b + c = <u>)
txv(23,w(B,A,C) = <i>°, w(A,B,C) = <j>°, w(A,C,B) = <k>°)
txv(24,Winkelsumme = <s>°)
pause(Wiederholen oder Beenden mit "Esc")
goto(start)
end.
```

```
begin
// Program 06
// Koordinatensystem und Punkte eingeben
// Linienzug zeichnen
// Streckenlängen berechnen
// Winkelgrößen berechnen
// Umkreis ermitteln
_start
compress
clr(0)
dez(2)
.g = 10
kor(g,1)
txv(1,Dreieck und Umkreis)
invar(Halblänge des Koordinatensystems: g)
kor(g,1)
txv(1,Dreieck und Umkreis)
pfa(-1)
A(1,3)
B(4,0)
C(8,8)
pfa(1)
input2(A)
input2(B)
input2(C)
lin(A,B,C,A)
lng(A,B)
     z = 4.24
.c = z = 4.24
lng(A,C)
     z = 8.60
.b = z = 8.60
lng(B,C)
     z = 8.94
.a = z = 8.94
.u = a + b + c = 21.78
win(B,A,C)
     z = 80.54
.i = z = 80.54
win(A,B,C)
     z = 71.57
.j = z = 71.57
win(A,C,B)
     z = 27.90
.k = z = 27.90
.s = i + j + k = 180.01
rou(s,0)
kru(A,B,C)
    = U(5.33,4.33)
    r = 4.53
pen(2,2)
lin(A,U)
pen(1,1)
txv(21,c = AB = <c>, b = AC = <b>, a = BC = <a>)
txv(22,Umfang = a + b + c = <u>)
txv(23,w(B,A,C) = <i>°, w(A,B,C) = <j>°, w(A,C,B) = <k>°)
txv(24,Winkelsumme = <s>°)
pfa(2)
txv(25,Umkreis: [U], r = <r>)
pfa(1)
pause(Wiederholen oder Beenden mit "Esc")
goto(start)
end.
```

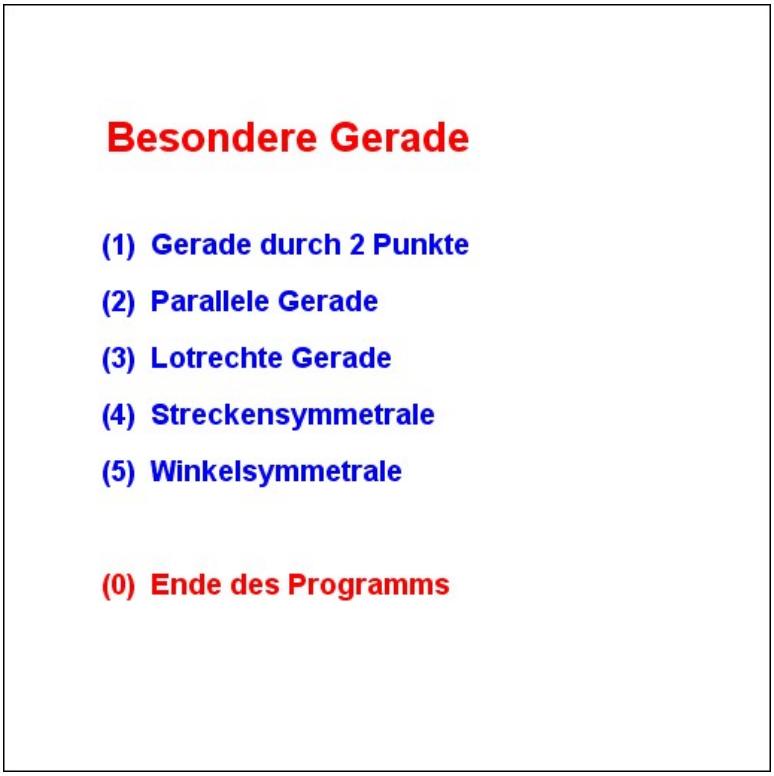
```
begin
// Programm 07
// Der Umkreis des Dreiecks

_start
compress
clr(0)
dez(2)
kor(10,0)
pfa(4)
txg(2)
txz(1,Der Umkreis des Dreiecks)
txg(0)
pfa(-1)
A(-10,-1)
B(6,-9)
C(0,9)
pfa(1)
.g = 16
invar(Halbbreite des Koordinatensystems: g)
kor(g,1)
txz(1,Der Umkreis des Dreiecks)
pause(Dreieck ABC eingeben)

pen(2,1)
input2(A)
input2(B)
input2(C)
lin(A,B,C,A)
pfa(-1)
swp(A,B,C)
    = Z(-1.33,-0.33)
kxy(Z)
    x = -1.33
    y = -0.33
pfa(1)
fil(x,y,1,220,240,220)
ras(g,1)
pen(1,1)
win(C,A,B)
     z = 71.57
.i = z = 71.57
win(A,B,C)
     z = 45.00
.j = z = 45.00
.z = 180 - j = 135.00
win(A,C,B)
     z = 63.43
.k = z = 63.43
lng(A,B)
     z = 17.89
.c = z = 17.89
lng(A,C)
     z = 14.14
.b = z = 14.14
lng(B,C)
     z = 18.97
.a = z = 18.97
.u = a + b + c = 51.00
.s = u/2 = 25.50
.d = s*(s-a)*(s-b)*(s-c) = 14395.16
ifgr(d,0,weiter)
pause(Die Punkte bilden kein Dreieck)
goto(start)
```

```
_weiter
.f = sqrt(d) = 119.98
.l = g/10 = 1.60
txr(1)
wib(B,A,C,l,a)
wib(A,B,C,l,b)
wib(A,C,B,l,g)
lib(A,B,c)
lib(A,C,b)
lib(B,C,a)
txv(2,Dreieck: [A], [B], [C])
txv(22,Seiten: a = <a>, b = <b>, c = <c>)
txv(23,Winkel: <i>°, <j>°, <k>°)
txv(24,Umfang = <u>, Fläche = <f>.)
pause(Weiter)

kor(g,1)
pen(2,1)
ums(A,A)
    = A(-10.00,-1.00)
ums(B,B)
    = B(6.00,-9.00)
ums(C,C)
    = C(0.00,9.00)
lin(A,B,C,A)
pause(Seitensymmetralen zeichnen und schneiden)

pen(1,2)
ssm(A,B)
y = 2.00 * x + -1.00 = 31.12
    k = 2.00
    d = -1.00
    = Z(-2.00,-5.00)
ums(D,Z)
    = D(-2.00,-5.00)
pause(Weiter)

ssm(B,C)
y = 0.33 * x + -1.00 = 4.30
    k = 0.33
    d = -1.00
    = Z(3.00,0.00)
ums(E,Z)
    = E(3.00,0.00)
pause(Weiter)

ssm(A,C)
y = -1.00 * x + -1.00 = -17.06
    k = -1.00
    d = -1.00
    = Z(-5.00,4.00)
ums(F,Z)
    = F(-5.00,4.00)
pause(Weiter zum Ergebnis)
```

```
kru(A,B,C)
    = U(0.00,-1.00)
    r = 10.00
.u = r = 10.00
pen(3,2)
lin(A,U)
pfa(-1)
hap(A,U)
    = Z(-5.00,-1.00)
kxy(Z)
    x = -5.00
    y = -1.00
txt(x,y,r)
pfa(1)
txv(1,Dreieck: [A], [B], [C])
txv(24,Umkreis: [U], r = <u>)
txz(25,Umkreismittelpunkt U = Schnittpunkt der Seitensymmetralen)
pause(Wiederholen oder Beenden mit "Esc")
goto(start)

end.
```

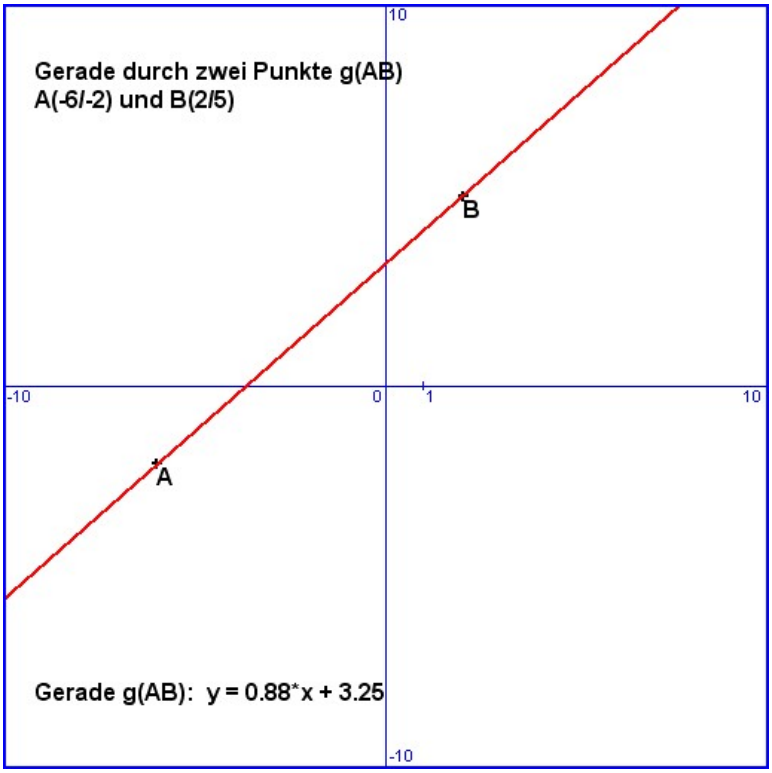



***Grafik zum Programm 07***

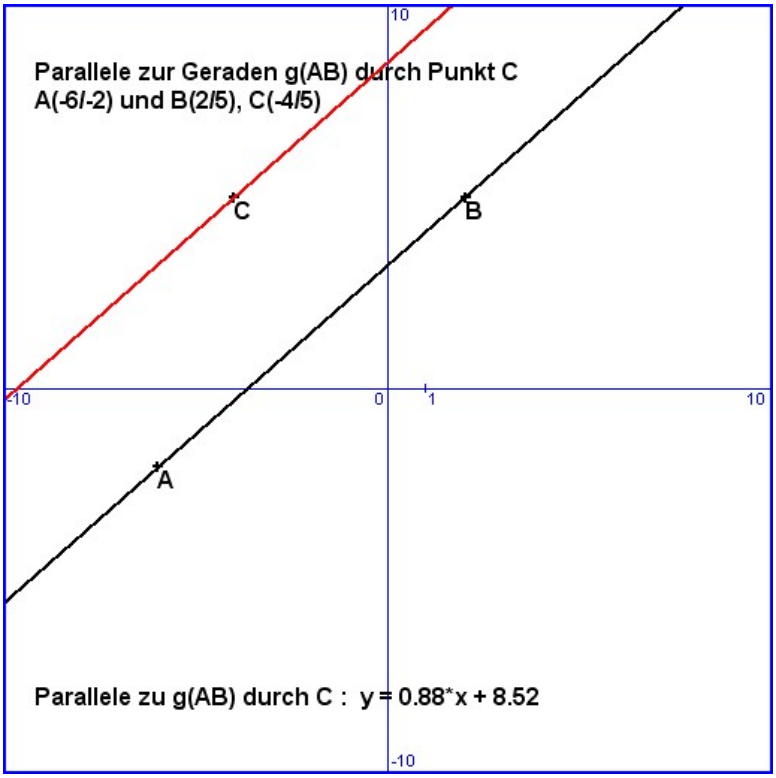



```
begin
// Program 08
// mit selektiver Menüauswahl
clr(0)
dez(2)
.m = 10000000000

_start
kor(10,0)
pfa(-1)
A(-6,-2)
B(2,5)
C(-4,5)
pfa(2)
txg(8)
txz(04,      Besondere Gerade)
txg(2)
pfa(4)
txz(08,         (1)  Gerade durch 2 Punkte)
txz(10,         (2)  Parallele Gerade)
txz(12,         (3)  Lotrechte Gerade)
txz(14,         (4)  Streckensymmetrale)
txz(16,         (5)  Winkelsymmetrale)
pfa(2)
txz(20,         (0)  Ende des Programms)
txg(0)
pfa(1)

.s = 0
select(Bitte auswählen: s,5)
ifeq(s,0,jmp0)
ifeq(s,1,jmp1)
ifeq(s,2,jmp2)
ifeq(s,3,jmp3)
ifeq(s,4,jmp4)
ifeq(s,5,jmp5)
```

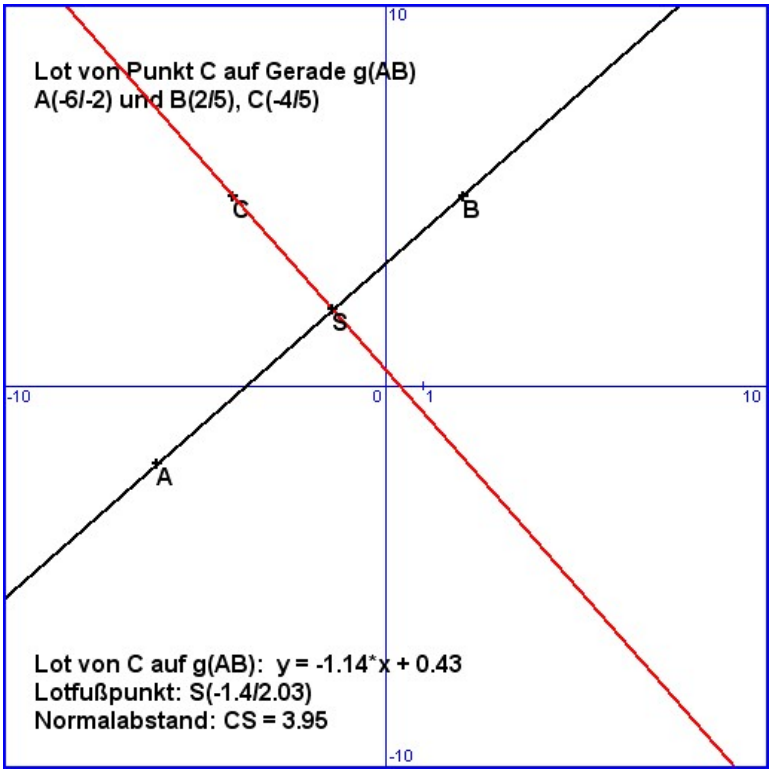
```
_jmp0
exit

_jmp1
compress
.g = 10
invar(Halbbreite des Koordinatensystems: g)
kor(g,1)
txz(2,Gerade durch zwei Punkte g(AB))
input2(A)
input2(B)
txv(3,[A] und [B])
pen(2,2)
ger(A,B)
y = 0.88 * x + 3.25 = 12.08
    k = 0.88
    d = 3.25

ifls(k,m,go11)
txv(24,Gerade g(AB):  x = <x>)
goto(go12)
_go11
txv(24,Gerade g(AB):  y = <k>*x + <d>)
_go12
pause(Zurück)
goto(start)
```



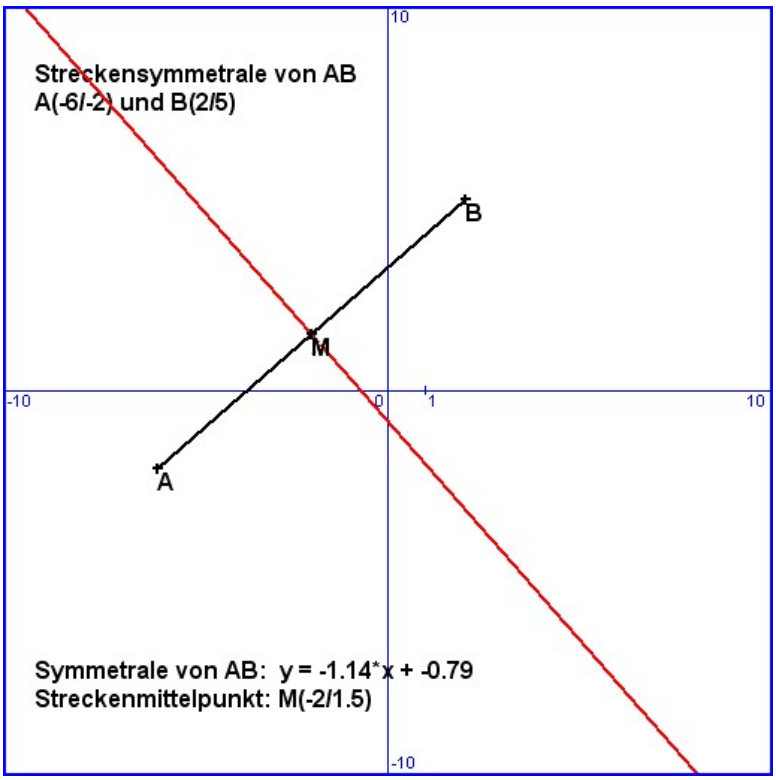
```
_jmp2
compress
.g = 10
invar(Halbbreite des Koordinatensystems: g)
kor(g,1)
txz(2,Parallele zur Geraden g(AB) durch Punkt C)
input2(A)
input2(B)
pen(2,1)
ger(A,B)
y = 0.88 * x + 3.25 = 12.08
    k = 0.88
    d = 3.25

input2(C)
txv(3,[A] und [B], [C])
pause(Weiter)
kxy(C)
    x = -4.00
    y = 5.00

.e = x = -4.00
.f = y - k*e = 8.52
pen(2,2)
fun(k*x+f)
ifls(k,m,go21)
txv(24,Parallele durch C zu g(AB):  x = <e>)
.pfa(-1)
D(e,0)
ger(C,D)
goto(go22)
_go21
txv(24,Parallele zu g(AB) durch C :  y = <k>*x + <f>)
_go22
pause(Zurück)
goto(start)
```



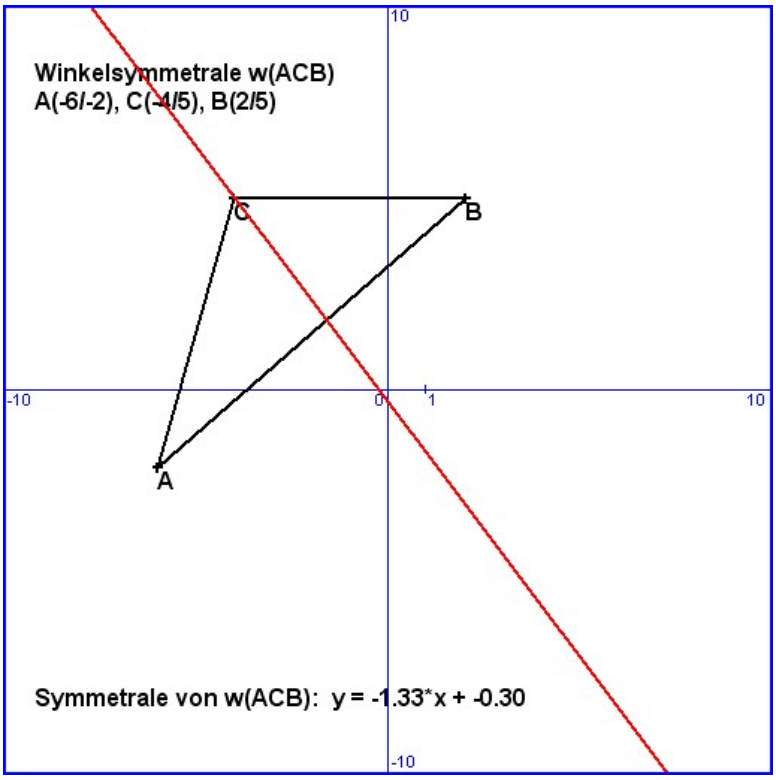
```
_jmp3
compress
.g = 10
invar(Halbbreite des Koordinatensystems: g)
kor(g,1)
txz(2,Lot von Punkt C auf Gerade g(AB))
input2(A)
input2(B)
pen(2,1)
ger(A,B)
y = 0.88 * x + 3.25 = 12.08
    k = 0.88
    d = 3.25

input2(C)
txv(3,[A] und [B], [C])
pause(Weiter)
pen(2,2)
lot(C,A,B)
y = -1.14 * x + 0.43 = -11.01
    k = -1.14
    d = 0.43
    = S(-1.40,2.03)
    z = 3.95

ifls(k,m,go31)
txv(23,Lot von C auf g(AB):  x = <x>)
txv(24,Lotfußpunkt: [S])
txv(25,Normalabstand: CS = <z>)
goto(go32)
_go31
txv(23,Lot von C auf g(AB):  y = <k>*x + <d>)
txv(24,Lotfußpunkt: [S])
txv(25,Normalabstand: CS = <z>)
_go32
pause(Zurück)
goto(start)
```



```
_jmp4
compress
.g = 10
invar(Halbbreite des Koordinatensystems: g)
kor(g,1)
txz(2,Streckensymmetrale von AB)
input2(A)
input2(B)
txv(3,[A] und [B])
pen(2,1)
lin(A,B)
pen(2,2)
ssm(A,B)
y = -1.14 * x + -0.79 = -12.23
    k = -1.14
    d = -0.79
    = Z(-2.00,1.50)

ums(M,Z)
    = M(-2.00,1.50)

ifls(k,m,go41)
txv(23,Symmetrale von AB:  x = <x>)
txv(24,Streckenmittelpunkt: [M])
goto(go42)
_go41
txv(23,Symmetrale von AB:  y = <k>*x + <d>)
txv(24,Streckenmittelpunkt: [M])
_go42
pause(Zurück)
goto(start)
```

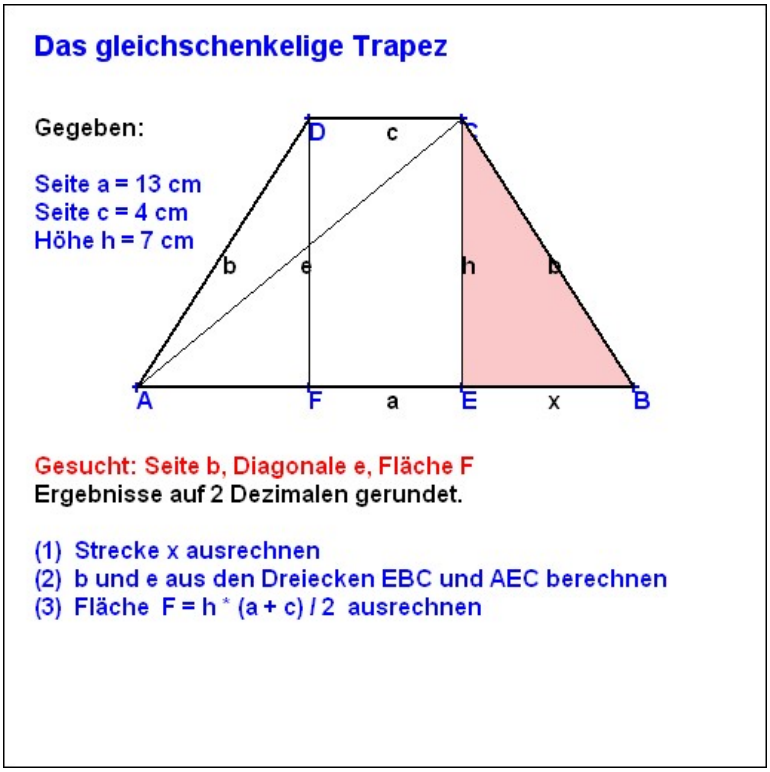


```
_jmp5
compress
.g = 10
invar(Halbbreite des Koordinatensystems: g)
kor(g,1)
txz(2,Winkelsymmetrale w(ACB))
input2(A)
input2(B)
input2(C)
txv(3,[A], [C], [B])
pen(2,1)
lin(A,B,C,A)
pause(Weiter)
pen(2,2)
sym(A,C,B)
y = -1.33 * x + -0.30 = -13.65
    k = -1.33
    d = -0.30

ifls(k,m,go51)
txv(24,Symmetrale von w(ACB):  x = <x>)
goto(go52)
_go51
txv(24,Symmetrale von w(ACB):  y = <k>*x + <d>)
_go52
pause(Zurück)
goto(start)

end.
```



# Das kleine Ein-Mal-Eins

```
begin
//*+! Sperrung von Editor, Taschenrechner und Drucker
// Program 09
// mit interaktiven Antworten und Bewertungen
clr(0)
.m = 10
.n = 0
.r = 0

_start
compress
dez(0)
.n = n + 1 = 1
kor(10,0)
txr(1)
pfa(1)
txg(2)
txv(2,<n>.-ter Versuch von maximal <m> Versuchen)
txg(8)
pfa(4)
txv(10,   Ein-Mal-Eins)
zuf(1,10)
     z = 5
.a = z = 5
zuf(1,10)
     z = 8
.b = z = 8
pfa(2)
txv(13,   <a> * <b> = ?)
.c = a * b = 40
.x = 0
invar(Produkt = ?: x)
pfa(1)
txv(15,   <a> * <b> = <c>)
pfa(1)
txg(2)
ifeq(x,c,richtig)
goto(falsch)

_richtig
.r = r + 1 = 1
txv(24,<r> von <n> Antworten richtig)
pause(<x> = richtige Antwort)
ifeq(n,m,aus)
goto(start)

_falsch
txv(24,<r> von <n> Antworten richtig)
pause(<x> = falsche Antwort)
ifeq(n,m,aus)
goto(start)

_aus
pfa(2)
txg(2)
.p = 100 / n * r
rou(p,0)
txv(24,<n> Versuche, <r> richtig . . . Ergebnis: <p>%)
pause(Ende - Drucken mit "F4")

end.
```




```
begin
//* Sperrung des Editors
// Program 10
// mit interaktiven Antworten und Bewertungen
clr(0)
.n = 0
.m = 0
_start
compress
dez(2)
kor(10,0)
.n = n + 1 = 2.00
.g = 10
.j = g/2 = 5.00
.i = 3*g/2 = 15.00
zuf(j,i)
     z = 13
.a = z = 13.00
.i = 3*a/5 = 7.80
.j = a/5 = 2.60
zuf(j,i)
     z = 4
.c = z = 4.00
.j = g/5 = 2.00
.i = 3*g/4 = 7.50
zuf(j,i)
     z = 7
.h = z = 7.00
kor(g,0)
txr(0)
pfa(4)
txg(2)
txz(1,Das gleichschenkelige Trapez)
txg(0)
```

```
.s = (a - c) / 2 = 4.50
.t = a - s = 8.50
.r = a + c = 17.00
.b = sqrt(h*h + s*s) = 8.32
.e = sqrt(h*h + t*t) = 11.01
.f = r*h/2 = 59.50
pen(2,1)
.u = a/2 = 6.50
.v = c/2 = 2.00
A(-u,0)
B(u,0)
C(v,h)
D(-v,h)
E(v,0)
F(-v,0)
lin(A,B,C,D,A)
pen(1,1)
lin(E,C)
lin(A,C)
lin(F,D)
pfa(-1)
swp(E,B,C)
    = Z(3.50,2.33)
kxy(Z)
    x = 3.50
    y = 2.33
fil(x,y,1,250,200,200)
pfa(2)
txr(0)
lib(A,B,a)
lib(C,D,c)
lib(B,C,b)
lib(A,D,b)
lib(A,C,e)
lib(E,B,x)
lib(E,C,h)
pfa(1)
txv(4,Gegeben:)
pfa(4)
txv(6,Seite a = <a> cm)
txv(7,Seite c = <c> cm)
txv(8,Höhe h = <h> cm)
pfa(1)
pfa(2)
txz(16,Gesucht: Seite b, Diagonale e, Fläche F)
pfa(1)
txz(17,Ergebnisse auf 2 Dezimalen gerundet.)
pfa(4)
txv(19,(1)  Strecke x ausrechnen)
txv(20,(2)  b und e aus den Dreiecken EBC und AEC berechnen)
txz(21,(3)  Fläche  F = h * (a + c) / 2  ausrechnen)
.p = 0
.q = 0
.w = 0
invar(Seite b, Diagonale e, Fläche F = ?: p,q,w)
pfa(2)
.x = s = 4.50
txv(23,b = <b> cm, e = <e> cm, F = <f> cm²)
pfa(1)
```

```
rou(p,2)
rou(q,2)
rou(w,2)
rou(b,2)
rou(e,2)
rou(f,2)
.i = p * q * w = 0.00
.j = b * e * f = 5450.39
ifeq(i,j,richtig)
goto(falsch)

_richtig
.m = m + 1
.z = m * 100 / n
rou(z,0)
txv(25,<n> Versuche, <m> richtig . . . Ergebnis: <z>%)
pause((<p>,<q>,<w>) = richtige Antwort &Abbruch mit "Esc")
goto(start)

_falsch
.z = m * 100 / n = 0.00
rou(z,0)
txv(25,<n> Versuche, <m> richtig . . . Ergebnis: <z>%)
pause((<p>,<q>,<w>) = falsche Antwort &Abbruch mit "Esc")
goto(start)

end.
```

```
================================================================
 GEOMATH-Skripts und PDF-Dateien zur gesamten Schulmathematik
================================================================

 #001; Die natürlichen Zahlen (pdf)
 #002: Die Grundrechenarten (Theorie)
 #003: Die Grundrechenarten (Praxis)
 #004: Die Dezimalzahlen
 #005: Rechnen mit Dezimalzahlen
 #006: Die ganzen Zahlen
 #007: Rechnen mit ganzen Zahlen
 #008: Teilbarkeit der Zahlen
 #009; Die Bruchzahlen (pdf)
 #010: Rechnen mit Bruchzahlen
 #011; Gleichungen (pdf)
 #012: Einfache Gleichungen
 #013; Prozentrechnungen (pdf)
 #014: Schlussrechnungen
 #015: Prozentrechnungen
 #016: Mischungsaufgaben
 #017: Bewegungsaufgaben
 #018: Leistungsaufgaben
 #019; 25 Übungsaufgaben (pdf)
 #020; Potenzen und Wurzeln (pdf)

 #031; Die reellen Zahlen (pdf)
 #032: Koordinatensysteme
 #033; Die komplexen Zahlen (pdf)
 #034: Rechnen mit komplexen Zahlen
 #035: Quadratische Gleichungen
 #036; Fundamentalsatz der Algebra (pdf)
 #037: Reelle Nullstellen
 #038; Logik und Mengenlehre (pdf)

 #041: Umfang & Fläche des Rechtecks
 #042; Grundlagen des Messens (pdf)
 #043: Einfache Rechtecksflächen
 #044: Parallelwinkel & Normalwinkel
 #045: Umfang & Fläche des Dreiecks
 #046: Dreieckskonstruktion I
 #047: Dreieckskonstruktion II
 #048: Winkelsumme im Dreieck
 #049: Vier merkwürdige Punkte
 #050: Kongruente Dreiecke
 #051: Die Strahlensätze
 #052: Ähnliche Dreiecke
 #053: Der Sehnensatz
 #054: Der Randwinkelsatz
 #055: Die Flächenformel von Heron
 #056: Umkreis- und Inkreisradius

 #061: Umfang & Fläche des Kreises
 #062: Kreisteile
 #063: Zusammengesetzte Flächen
 #064: Lehrsatz von Pythagoras
 #065: Rechtwinkelige Dreiecke
 #066: Gleichschenkelige Dreiecke
 #067: Diagonalen des Rechtecks
 #068: Rechteck und Kreis
 #069: Parallelogramm
 #070: Deltoid
 #071: Trapez
```

```
#072: Räumliche Körper
#073: Quader
#074: Prisma
#075: Pyramide
#076: Tetraeder
#077: Rhombendodekaeder
#078: Zylinder
#079: Kegel
#080: Kugel
#081: Kugelteile

#091: Koordinatensysteme
#092: Eine Gerade in der Ebene
#093: Zwei Gerade in der Ebene
#094: Besondere Gerade
#095: Determinanten
#096: Lin. Systeme mit 2 Variablen
#097: Lin. Systeme mit 3 Variablen
#098; 40 Textgleichungen (pdf)

#111: Abbildungen
#112: Schiebungen
#113: Drehungen
#114: Spiegelungen
#115: Erster Spiegelungssatz
#116: Zweiter Spiegelungssatz
#117: Streckungen
#118: Verkettungen

#121: Winkelfunktionen
#122: Sinussatz und Cosinussatz
#123: Parallelogramme I
#124: Parallelogramme II
#125: Parallelogramme III
#126: Dreiecke (SSS)
#127: Dreiecke (SWS)
#128: Dreiecke (SsW)
#129: Dreiecke (SWW)
#130; Zusammengesetzte Winkel (pdf)

#131: Höhenmessungen I (Theorie)
#132: Höhenmessungen I (Praxis)
#133: Höhenmessungen II (Theorie)
#134: Höhenmessungen II (Praxis)
#135: Vorwärtseinschneiden (Theorie)
#136: Vorwärtseinschneiden (Praxis)
#137: Rückwärtseinschneiden (Theorie)
#138: Rückwärtseinschneiden (Praxis)

#141: Vektorrechnung I
#142: Vektorrechnung II
#143: Vektorrechnung III

#144: Skalarprodukte in der Ebene
#145: Abstand zweier Punkte
#146: Gerade in der Ebene I
#147: Gerade in der Ebene II
#148: Abstand von Punkt und Gerade
#149: Schnittpunkt von Geraden
#150: Streckenteilungen
#151: Strecken- und Winkelsymmetralen
#152: Dreiecksberechnungen
#153: Umkreis des Dreiecks
#154: Inkreis des Dreiecks
#155: Vier merkwürdige Punkte
```

```
#156: Skalarprodukte im Raum
#157: Abstand zweier Punkte
#158: Ebenen im Raum I
#159: Ebenen im Raum II
#160: Abstand von Punkt und Ebene
#161: Gerade im Raum
#162: Abstand von Punkt und Gerade
#163: Schnitt von Ebene und Gerade
#164: Zwei Gerade im Raum
#165: Schnittwinkel zweier Ebenen
#166; Lineare Systeme (pdf)
#167; Analytische Geometrie (pdf)

#171: Kegelschnittslinien (KSL)
#172: Ellipse
#173: Hyperbel
#174: Parabel
#175: Wichtige Formeln
#176; Evoluten und Tangenten (pdf)
#177: Gerade und KSL
#178: Kreis-Tangenten IN
#179: Kreis-Tangenten VON
#180: KSL-Tangenten IN
#181: KSL-Tangenten VON
#182: Schnitt von zwei KSL
#183: Schmiegkreise der KSL
#184: Scheitelgleichungen der KSL

#191: Funktionen
#192: Darstellung von Funktionen
#193: Schwingungen I
#194: Schwingungen II
#195: Exponenten & Logarithmen I
#196: Exponenten & Logarithmen II
#197: Ungebremstes Wachstum
#198: Gebremstes Wachstum I
#199: Gebremstes Wachstum II

#201: Folgen I    (Definitionen)
#202: Folgen II   (Grenzwerte)
#203: Folgen III  (Rechenregeln)
#204: Folgen IV   (Beispiele)
#205: Die Ludolfsche Zahl "pi"
#206: Zinseszinsen
#207: Die Eulersche Zahl "e"
#208: Geometrische Reihen

#209: Die Stetigkeit
#210: Differenzieren (Theorie)
#211: Differenzieren (Praxis)
#212; Ableitungsregeln (pdf)

#217: Kurvendiskussion (Theorie)
#218: Kurvendiskussion (Praxis)
#219: Asymptoten von Kurven
#220: Extremwertaufgaben (Theorie)
#221: Extremwertaufgaben (Praxis)
#222: Schnitt zweier Kurven
#223: Kurventangente IN
#224: Kurventangente VON
#225: Gleitende Tangenten
#226; Wirtschaftsmathematik (pdf)
#227; Reihenentwicklungen (pdf)
#228; Differenzialgleichungen (pdf)
```

```
#231: Stammfunktionen
#232: Das Riemannsche Integral
#233: Der Hauptsatz
#234: Die Fläche unter der Kurve
#235; Integralrechnung (pdf)
#236; Höhere Integrationsmethoden (pdf)
#237: Die Fläche unter den KSL

#239: Drehkörper-Volumen (Theorie)
#240: Drehkörper-Volumen (Praxis)
#241: Flächen-Schwerpunkt (Theorie)
#242: Flächen-Schwerpunkt (Praxis)
#243: Wichtige Hinweise
#244: Mantelfläche von Drehkörpern
#245: Bogenlänge von Kurven
#246: Schwerpunkt von Kurven
#247: Schwerpunkt von Drehkörpern
#248: Der Torus
#249: Besondere Volumsberechnungen

#251: Beschreibende Statistik I
#252: Beschreibende Statistik II
#253; Statistische Formeln (pdf)
#254: Statistische Datenauswertung
#255: Häufigkeit, Wahrscheinlichkeit
#256; Normalverteilung (pdf)
#257: Normalverteilung (Praxis)
#258; Beurteilende Statistik (pdf)
#261; Regressionen (pdf)
#262: Lineare Regression
#263: Quadratische Regression
#264: Kubische Regression
#265: Exponentielle Regression

#271; Kombinatorik (pdf)
#272: Kombinatorik (Praxis)
#273; Binomialverteilung
#274: Drei Zufallsexperimente
#275; Wahrscheinlichkeitstheorie (pdf)
#276; 23 Übungsaufgaben (pdf)

#281: Abbildungsgleichungen
#282: Bewegungen von KSL
#283: Hauptachsentransformationen

#299: Eine Multimedia-Show


=====================================================================
ENDE
=====================================================================
```