

Einführungskurs PostScript

PostScript: Seitenbeschreibungssprache, entw. von Adobe Systems. Weite Verbreitung im Druckgrafik-Bereich.

Erlaubt Kombination von Textelementen und Vektorgrafiken; volle Mächtigkeit einer Programmiersprache.

Einlesen durch PostScript-Interpreter, dort Umsetzung in Rastergrafik für die Ausgabe (PS-Interpreter in vielen Druckern bereits fest implementiert).

PS-Viewer für den Bildschirm:

ghostscript (MS-Windows, OS-2 und Unix; Aladdin), *ghostview* (komfortabler Viewer unter MS-Win. und OS-2; Ghostgum) - Shareware; *xpsview* (Unix).

Download: <http://www.cs.wisc.edu/~ghost/>
<http://www.ghostscript.com>

PostScript-Programme: lesbare ASCII-Dateien.

- Beschränkung auf 128 Zeichen Standard-ASCII (keine Umlaute und ß verwenden!)
- Unterscheidung von Groß- und Kleinschreibung
- () [] { } / % besitzen besondere Bedeutung
- in Zahlenangaben Dezimalpunkt statt Komma
- Kommentare beginnen mit %, zeilenweise

Grundprinzip:

- konsequent stackorientiert

Operanden werden nacheinander auf einen Stack abgelegt, Befehle werden nach den Operanden eingegeben und beziehen sich immer auf die obersten Stack-Elemente.

- Postfix-Notation für alle Operatoren und Funktionsaufrufe.

Funktionsargumente werden vom Stack geholt.

Beispiele:

17 4 add % Addition 17+ 4

			4		
17	17	4	17	add	21

1 add % Inkrementierung d. obersten Stackelements um 1
42 100 moveto % Bewegung des virtuellen Stiftes an den Punkt (42; 100)
5 { 3 4 rlineto } repeat % Schleife (Argumente: Iterationszahl, iterierte
% Prozedur)

Falls nicht genügend Einträge auf dem Stack vorhanden sein sollten wird der Druckauftrag mit der Fehlermeldung „stackunderflow“ abgebrochen.

Datentypen:

Integer übliche Schreibweise, erweitert (optional) durch Basis#Zahl (z.B. 16#3f)

Real übliche Schreibweise, incl. E-Schreibweise (z.B. 1.3e3 für $1,3 \times 10^3$)

Boolean true, false

String (Hallo Welt!)

Array [3 42 0]

Sonderzeichen in Strings: \b, \f, \n, \r, \t, \\, \(, \), \xxx (Oktalzahl) (vgl. Sprache C)

Definition (incl. Deklaration und Initialisierung) einer Variablen:

/VarName Wert def

Prozedur-Definition:

/ProzedurName

{

... Anweisungen ...

} def

Abstrakte Zeichenfläche, abstraktes (2D-) Koordinatensystem ("Weltkoordinaten").

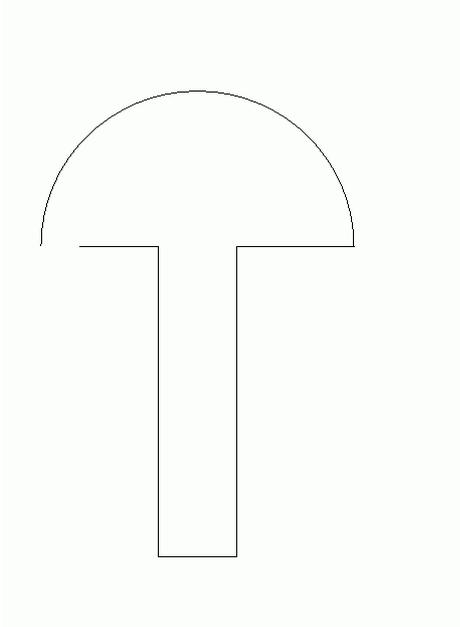
Voreinstellung: Ursprung (0; 0) = linke untere Ecke

Einheitslänge: 1 *Big Point* (dot) = 1/72 Zoll = 0,35277... mm

Ausmaße des voreingestellten A4-Blattes: (596,59; 843,75) = rechte obere Ecke.

Beispiele für erste Grafikbefehle:

```
100 500 moveto          % Startpunkt anfahren
200 500 lineto          % Speichern einer Linie, noch kein Zeichnen
200 100 lineto
300 100 lineto
0 400 rlineto
250 500 200 0 180 arc  % Kreisbogen: Mittelpunkt Radius
                        % Startwinkel Endwinkel
stroke                  % Zeichnen des aktuellen Pfades,
                        % dieser wird gelöscht
showpage               % Ausgabe und Löschen der Seite,
                        % Zurücksetzen der Grafik-Parameter
```

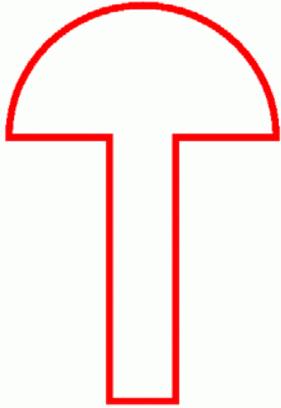


Ergebnis:

Schließen eines Polygonzuges (der Kreisbögen enthalten kann):

```
closepath
10 setlinewidth        % Setzen der Liniendicke für nächstes stroke
1 0 0 setrgbcolor     % Setzen der Zeichenfarbe auf
                       % Rot (RGB-Modell: (1; 0; 0) )
```

Ergebnis nach Einfügen vor `stroke` im letzten Beispiel:



Zum Füllen von Polygonen ist es notwendig, den aktuellen grafischen Zustand zu "retten" und später wiederherzustellen (Stack-Operationen!):

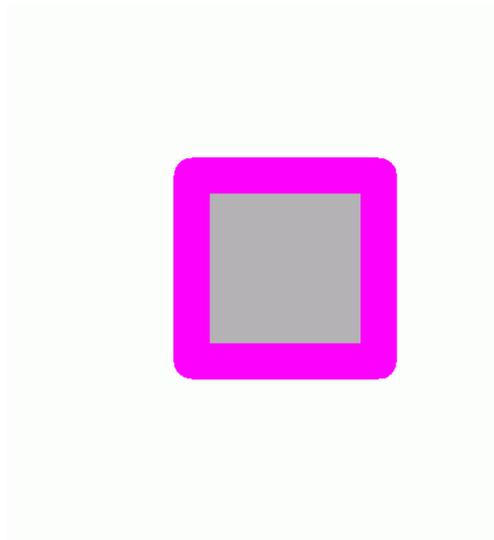
Aktuellen grafischen Zustand (Pfad u.a. Parameter) retten:

gsave

Wiederherstellen:

grestore

Beispiel:



wurde erzeugt mit folgendem PostScript-Programm:

```

80 80 moveto 1 setlinejoin    % Abrunden der Ecken. 0 = spitz,
                                % 2 = abgekantet.
80 0 rlineto
0 80 rlineto
-80 0 rlineto
closepath
0.7 setgray                    % Grauwert setzen, 0 = schwarz, 1 = weiss.
gsave fill                      % Fuellen d. akt. Pfades mit Grauton 0.7,
                                % Pfad wird geloesch...
grestore                        % und wiederhergestellt
0 1 0 0 setcmykcolor           % Zeichenfarbe Magenta
                                % (im CMYK-System)
15 setlinewidth
stroke
showpage

```

Ausgabe von Text:

```

/Arial findfont 24 scalefont setfont
    % obligatorisch für Text- Ausgabe: Auswahl, Skalieren u. Setzen
    % eines Fonts
10 10 moveto
(Hallo Welt!) show
showpage

```

Hallo Welt!

Erzeugen eines neuen, leeren Strings auf dem Stack:

Länge string

Arithmetik in PostScript:

```

3 4 mul                        % 3·4
4 5 6 mul add                  % 4 + 5·6
1 2 add 3 4 sub mul 5 6 add mul % (1+2)·(3-4)·(5+6)

```

Beispiel:

Erste 20 Fibonacci-Zahlen (1 1 2 3 5 8... a_i a_{i+1} (a_i+a_{i+1})...) ausgeben:

```
/Times findfont 16 scalefont setfont
10 10 moveto % obligatorisch: Startpunkt
/schreibe { % Schreibprozedur für obersten Stackeintrag
dup % ob. Stackeintrag duplizieren
6 string cvs % leeren String d. Länge 6 erz.,
% in diesen kommt d. ob. Stackeintr.
show % Anzeige des Strings
10 0 rmoveto % nach rechts weiterrücken
} def
0 1
20 {
schreibe % Textausgabe des ob. Stackelem.
2 copy % die obersten 2 Stackel. werden
% dupliziert und die Kopien werden...
add % durch ihre Summe ersetzt
} repeat % wiederhole dies 20 mal
showpage
```

Ergebnis:

1 1 2 3 5 8 13 21 34 55 89 144 233 377 610 987 1597 2584 4181 6765
--

Aufgabe (noch ohne Wertung):

Schreiben Sie ein PostScript-Programm, das die ersten 20 Quadratzahlen (beginnend mit 1) untereinander aufschreibt. Dabei sollen die geraden Quadratzahlen rot, die ungeraden schwarz erscheinen.

Hinweis: Es gibt mehrere Lösungswege. Einige davon könnten die folgenden Befehle verwenden:

pop entfernt oberstes Stackelement

dup dupliziert den obersten Wert auf dem Stack

exch vertauscht die beiden obersten Stack-Einträge

String **stringwidth** legt Breite und Höhe (in pt) des Strings als Wert auf dem Stack ab

Boolean Prozedur **if** Prozedur, notiert in {...}, wird ausgeführt, wenn der boolesche Ausdruck **true** liefert

Boolean Prozedur1 Prozedur2 **ifelse**

Wert1 Wert2 **eq** Test auf Gleichheit
Wert1 Wert2 **ne** Test auf Ungleichheit
Wert **neg** bewirkt Vorzeichenwechsel des Wertes
Startwert Inkrement Endwert Prozedur **for** for-Schleife

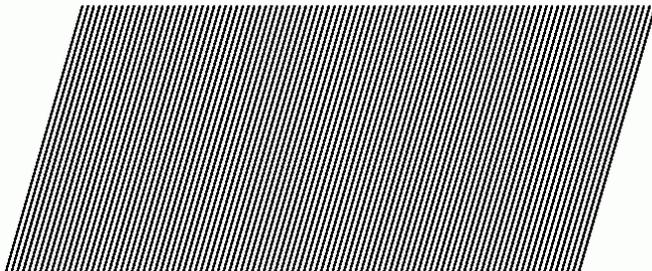
Verlagerung des Koordinatenursprungs um (*tx*, *ty*):

tx ty **translate**

Beispiel:

0 1 100

```
{  
  10 10 moveto  
  50 150 lineto  
  stroke  
  3 0 translate  
} for
```



showpage

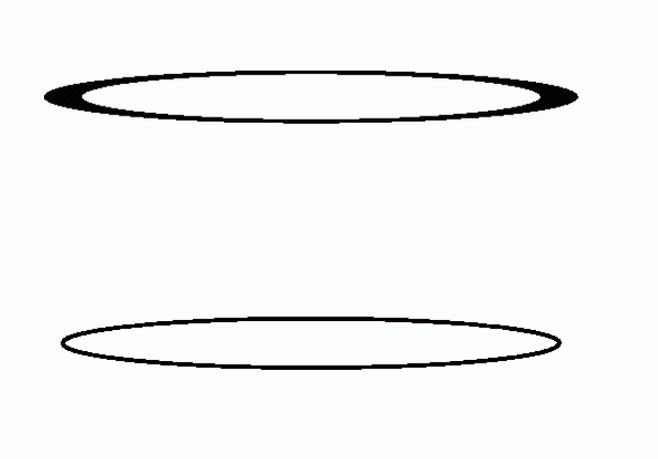
Neuskalierung des Koordinatensystems mit Faktoren *sx*, *sy*:

sx sy **scale**

Beispiel: Ellipsen

```
3 setlinewidth
10 1 scale
25 300 20 0 360 arc
closepath
stroke % Liniendicke wird mitskaliert
25 100 20 0 360 arc
closepath
0.1 1 scale
stroke % Liniendicke wird nicht mitskaliert
showpage
```

Ergebnis:



Rotation des Koordinatensystems um gegebenen Winkel:

Winkel rotate

Beispiel:

```
/Helvetica findfont 18 scalefont setfont
100 100 moveto
(Gerade ) show
45 rotate
3 1 scale
(Quer) show
1 3 div 1 scale
-45 rotate
( Gerade) show
showpage
```



Gerade **QUER** Gerade

Weitere Befehle:

`x y mod` Modulus-Operation (Rest bei Division)

`x y gt` Test auf größer

`x y ge` Test auf größer oder gleich

`x y lt` Test auf kleiner

`x y le` Test auf kleiner oder gleich

`x sin` Sinus

`x cos` Cosinus

`x atan` Arcus tangens

`x sqrt` Quadratwurzel

`x exp` Exponentialfunktion

`x ln` Logarithmus (Basis e)

`x log` Logarithmus (Basis 10)

`x floor` größtes Ganzes kleiner-gleich x

`x ceiling` kleinstes Ganzes größer-gleich x

`rand` Zufallszahl

`x y and` logisches Und

`x y or` logisches Oder

`x y xor` logisches Entweder-oder

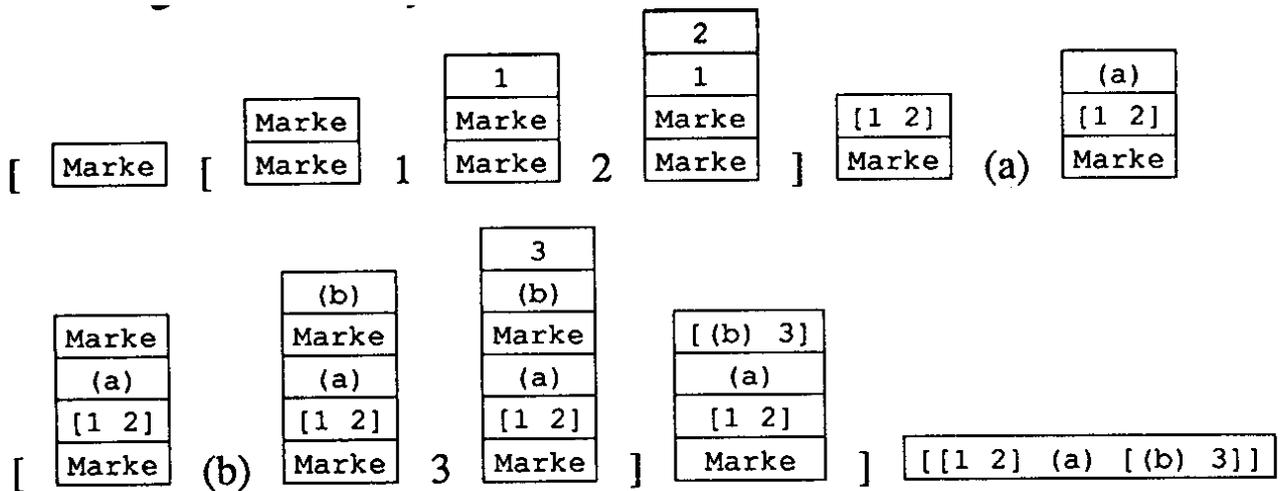
`x not` logische Verneinung

Arrays:

Definition eines Feldes beginnt mit »[« und endet mit »]«

Beispiel 1: [1 2 3]

Beispiel 2: `[[1 2] (a) [(b) 3]]`
 Stackdiagramm:



Beispiel:

```

/X 10 def % Der Name X wird mit der Zahl 10 verbunden.
/A [X 1 add] def % Durch die sofortige Auswertung wird die
% Variable X durch seinen Wert 10 ersetzt und
% der Befehl »add« ausgeführt.
/B {X 1 add} def % Dem Namen B wird eine Prozedur zugewiesen.
/X 20 def % X = 20
A % Unter dem Namen A findet sich das Array [11]!
B % Hier wird die unter B abgelegte Prozedur
% ausgeführt. Das Ergebnis ist die Zahl 21,
% weil X geändert wurde
  
```

Array Prozedur forall wendet die Prozedur auf jeden Wert aus dem Array an
Array num get holt den $(num+1)$ -ten Wert aus dem Array und legt ihn auf den Stack

Array num Wert put ersetzt den $(num+1)$ -ten Wert des Arrays durch *Wert*
Anzahl_der_Elemente array erzeugt ein leeres Array

Array aload entpackt Array komplett auf den Stack

$w_1 \dots w_n$ *Array astore* packt die Werte w_1, \dots, w_n zu einem Array zusammen

Stack:

clear alle Stackeinträge löschen

count legt Anzahl der Stackeinträge auf den Stack

num *roll* Rotation d. obersten num Einträge des Stacks um j Einträge

Aufgabe (noch ohne Wertung):

Zeichnen Sie mit PostScript ein Wabenmuster (regelmäßiges Sechseck-Gitter)!