TigerJython: Übersicht, Zusammenfassung, Beispiele

VERGLEICH MIT TI-BASIC

	TigerJython	TI-Basic (TI-89)
Eingabe	`x = input("Zahl")` `z = inputInt("ganze Zahl")`	`input "Zahl",x`
Ausgabe	`print "hallo"`	`disp "hallo"`
	`print 5*z`	`disp 5*z`
	`print "Die Zahl ist ",z`	`disp "Die Zahl ist ",z`
	`msgDlg("Zahl ist " + str(z))`	`text "Zahl ist "&string(z)`
Kommentar	`# Kommentar`	◕ Kommentar
Rechnen	`+ - * / ()`	`+ - * / ()`
Potenz	`a**3`	`a^3`
Grosse Zahlen	`c = 2.998e8`	
ganzzahlige Div.	`a//b`	`intdiv(a,b)`
Rest	`a%b`	`mod(a,b)`
Mathematische Funktionen	`from math import *` `sin(sqrt(z*pi))`	 `sin(√(z*π))`

Trigonometrie	`sin, cos, tan`
	`asin, acos, atan`
	`# immer im Bogenmass`
	`radians(30), degrees(pi)`
		`sin, cos, tan`
		$\sin^{-1}$, $\cos^{-1}$, $\tan^{-1}$
		◕ Winkelmasseinheit je
		◕ nach Einstellung

Undefinierte Werte	`from math import *` `isnan(x)`	
Runden	`round(z,2)`	`round(z,2)`

Zufallszahl	`from random import *`
	`randint(1,6)`
	`random()`
		
		`rand(6)`
		`rand()`

Wert zuweisen	`a, b = 2, 5`
	`c = a+b`
		`2 STO a`
		`5 STO b`
		`a+b STO c`

Wert erhöhen	`x += 1`	`x+1 STO x`
	`s += k**2`	`s+k^2 STO s`
	`+=, -=, *=, /=, //=, %=`	
logische Op.	`And or not`	`and or not`
Test: gleich?	`A == b`	`a=b`
Test: ungleich?	`A != b`	`a/=b`
	`<, >, <=, >=`	`<, >, <=, >=`

Logische Werte	`True, False`	`true, false`

Bedingung	`if x<0:`
	`  print "Zahl ist negativ"`
	`elif x>0:`
	`  print "Zahl ist positiv"`
	`else:`
	`  print "Zahl ist 0"`
		`if x<0 then`
		`  disp "Zahl ist negativ"`
		`elseif x>0 then`
		`  disp "Zahl ist positiv"`
		`else`
		`  disp "Zahl ist 0"`
		`endif`

Schleifen	`while a<b:`
	`  a += 1`
		`while a<b`
		`  a+1 STO a`
		`endwhile`

	`for i in range(5,n):`
	`  summe += i`
	`# range(0,10,2): 2er-Schritte`
		`for i,5,n`
		`  summe + i STO summe`
		`endfor`

Repeat gibt's nur in TigerJython!	`repeat 10:` `  s += 1`	`loop` `endloop`
Abbruch einer Schleife	`break`	`exit`

Funktion	`def potenz(base, expo)`
	`  return base**expo`
		`potenz(base,expo)`
		`func`
		`return base^expo`
		`endfunc`

Globale Variablen	`# innerhalb einer Funktion`
	`  global phi`
	`  phi = (5**0.5 + 1)/2`
		◕ Nicht speziell dekla-
		◕ rierte Variablen sind
		◕ immer lokal

Lokale Variablen	`# Nicht speziell deklarierte`
	`# Variablen in einer Fu.`
	`# sind immer lokal`
		◕ innerhalb einer Funktion
		`local phi`
		`(5^0.5 + 1)/2 STO phi`

Befehl über mehrere Zeilen schreiben	`# eigentlich unnötig, aber \` `mit "\" trotzdem möglich`	◕ automatisch

VARIABLENARTEN IN TIGERJYTHON

Ganze Zahl (Integer)	`z = inputInt("ganze Zahl")` `int(pi)`
Grosse Ganzzahl (Long)	`# Anzeige mit "L" am Schluss`

Dezimalzahl (Float)	`x = inputFloat("Dezimalzahl")`
	`float(5)`
	`c = 2.9979e8`

Zeichenkette (String)	`t = inputString("Text eingeben")`
Gross-/ Kleinbuchstaben erzeugen	`t.upper(), t.lower()`
Länge abfragen	`len(t)`
Erster Buchstabe	`t[0]`
Letzte vier Buchstaben	`t[len(t)-5: len(t)-1]`
ASCII-Code (Unicode)	`ord("x"), chr(65)`
Zeilenumbruch im Text	`\n`
Stelle in Text finden	`t.index("bla")`
Text aufspalten (ergibt Liste)	`t.split("x")`
Text ersetzen	`t.replace("x", "xox")`
Daten einfügen	`print "{0} {1} {2}".format("Hallo", "du", "da")`
Liste (Array)	`day = ["mo", "di", "mi", "do", "fr", "sa", "so"]`
Liste aufeinanderfolgender Zahlen	`ziffern = range(10) # oder range(0, 10)`
Arithmetische Folge	`even = range(0, 10, 2)`
Listenelement abrufen	`day[2] # gibt drittes Element`
oder	`day[-2] # gibt zweitletztes Element`
Listenelement ändern	`day[1] = "tue"`
Listenelement löschen	`del day[6]`
Listenelement hinten hinzufügen	`day.append("sun")`
ganze Liste löschen	`del ziffern`
Test, ob El. in Liste ist	`"sun" in day`
Elementnummer finden	`day.index("sun")`
aufsteigend sortieren	`day.sort()`
absteigend sortieren	`day.reverse()`
grösstes Element	`max(ziffern)`
kleinstes Element	`min(ziffern)`
Länge der Liste	`len(day)`
Summe der Elemente	`sum(ziffern)`
Matrix erzeugen	`unitmatrix = [ [1, 0, 0], [0, 1, 0], [0, 0, 1] ]`
Element a_{23} abrufen	`unitmatrix[1][2]`

Siehe auch http://docs.python.org/3/tutorial/, Kapitel 3 und 5

TURTLE-GRAFIK IN TIGERJYTHON

Turtle bzw. Fenster erstellen (-400 ≤ x ≤ 400, -300 ≤ y ≤ 300)	`from gturtle import *` `makeTurtle()`
Linie und Punkt	`forward(s), dot(d)`
Drehen	`right(w), left(w)`
	`penUp(), penDown()`
	`hideTurtle(), showTurtle()`
	`setLineWidth(w), penWidth(w)`
Turtle ausrichten (0 = Nord)	`heading(winkel)`
Geschwindigkeit (min … max.)	`speed(0), speed(20), speed(100), speed(-1)`
Bekannte Farbe verwenden	`setPenColor(« blue »)`
Fläche füllen	`setFillColor(„blue“), fill()`
	`fillToPoint(), fillOff()`
Ganzes Fenster füllen	`clear(„blue“)`
Farbe mischen	`makeColor(r,g,b)`
Farbe verwenden	`setPenColor( makeColor(r,g,b) )`
Linie mittels Koordinaten	`setPos(x, y), moveTo(x, y)`
Punkt mittels Koordinaten	`setPos(x, y), dot(d)`
Koordinaten abfragen	`getX(), getY()`

Aktion bei Mausklick (Text schreiben)	`def click(x, y):`
	`  setPos(x, y)`
	`  label("Hallo du!")`
	`makeTurtle(mouseHit = click)`

Aktion bei Mausklick erst, wenn die vorher laufende Aktion beendet ist	`makeTurtle(mouseHitX = click)`

Aktion beim Ziehen der Maus	`def drag(e):`
	`  x = toTurtleX(e.getX())`
	`  y = toTurtleY(e.getY())`
	`  moveTo(x, y)`
	`makeTurtle(mouseDragged = drag)`

Verschiedene Aktionen mit Maus	`makeTurtle(mousePressed = press,`
	`  mouseReleased = release,`
	`  mouseClicked = click`
	`  mouseMoved = move)`

Cursor-Form verändern	`setCursor(Cursor.DEFAULT_CURSOR)`
	`setCursor(Cursor.CROSSHAIR_CURSOR)`
	`setCursor(Cursor.HAND_CURSOR)`
	`setCursor(Cursor.TEXT_CURSOR)`

Bessere (schnellere) Grafik mit dem TigerJython-Modul **gpanel** von Aegidius Plüss
Tutorial auf www.tigerjython.ch, Kapitel „Grafik & Bilder“

SPEZIELLE MODULE IN TIGERJYTHON

Importarten	`from math import *`
	`sin(x)`
	`# oder`
	`import math`
	`math.sin(x)`

Ostern mit tjaddons	`from tjaddons import *` `easterday(2014)`
Regenbogenfarben mit tjaddons	`from tjaddons import *` `makeRainbowColor(50, 400)`
Kurze Pause mit time	`from time import *` `sleep(0.5)`

Laufzeit messen mit time	`from time import *`
	`startzeit = time()`
	`# auszuführende Befehle`
	`endzeit = time()`
	`laufzeit = endzeit - startzeit`

TIGERJYTHON: EINE EIGENE FUNKTIONEN-BIBLIOTHEK ANLEGEN

Erstelle eine übliche python-Datei, in welcher alle Funktionen, die in anderen Programmen verwendet werden sollen, definiert sind.

Diese Datei muss in jenem Ordner abgelegt werden, in dem sich auch das Programm tigerjython (bzw. tigerjython.jar) befindet.

Beachte, dass einige tigerjython-spezifische Befehle (z.B. `repeat`) in dieser Bibliotheks-Datei nicht vorkommen dürfen! (?)

Als Alternative zum einfachen `repeat` bietet sich die `for`-Schleife an (siehe Zeile `5`).

```
# Diese Datei heisst meinebibliothek.py
from gturtle import *
def vieleck(anz, seite):
    for i in range(anz):
        forward(seite)
        left(360/anz)
def ...
```

Die Bibliotheks-Datei muss in am Anfang jedes tigerjython-Programms importiert werden, wenn man darauf zugreifen will.

Das tigerjython-Programm, in welchem man die Bibliothek verwendet, muss selber auch im gleichen Ordner abgespeichert sein.

```
from gturtle import *
from meinebibliothek import *
makeTurtle()
setPenColor("red")
vieleck(10, 25)
forward(sl/4)
...
```

BEISPIEL 1 DEMO_CLICK_FIGURE

```
from gturtle import *
from random import *

def vieleck(anz,seite,dicke,farbe):
      penWidth(dicke)
      setPenColor(farbe)
      for i in range(anz):
            forward(seite)
            left(360/anz)

def spirale(anz,seite,richtung,dicke,farbe):
      penWidth(dicke)
      setPenColor(farbe)
      if richtung == 1:
            for i in range(16*anz):
                  forward(i*seite/10)
                  left(22.5)
      else:
            for i in range(16*anz):
                  forward(i*seite/10)
                  right(22.5)

def onClick(x,y):
      setPos(x,y)
      a = randint(1,2)
      if a == 1:
            n = randint(3,15)
            l = randint(10,50-2*n)
            d = randint(2,6)
            f = makeColor(random(), random(), random())
            vieleck(n,l,d,f)
      else:
            n = randint(2,4)
            l = randint(2,10)
            r = randint(1,2)
            d = randint(2,6)
            f = makeColor(random(), random(), random())
            spirale(n,l,r,d,f)

makeTurtle(mouseHit = onClick)
hideTurtle()
```